IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 July 2024, accepted 7 August 2024, date of publication 22 August 2024, date of current version 30 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3442300

RESEARCH ARTICLE

USING Sensor: User-Smartphone Decoupling Detection as a Mobile-Based Virtual Sensor

WOOJIN PARK, HYEYOUNG AN, AND SOOCHANG PARK, (Member, IEEE)
Department of Computer Engineering, Chungbuk National University, Seowon, Cheongju, Chungbuk 28644, Republic of Korea

Corresponding author: Soochang Park (cewinter@chungbuk.ac.kr)

This work was supported by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea under Grant NRF-2021S1A3A2A01090919.

ABSTRACT In many disaster situations, the location of victims is the most important information for saving them. To provide such information, most studies simply use indoor positioning technologies of the victim's mobile device. Nevertheless, their schemes have a large lack to point out the exact location of a victim because people sometimes drop off their mobile device or put it down some places, such as a desk and a table. In other words, it merely provides the location of mobile devices and not of victims. Hence, this paper proposes a novel virtual sensing mechanism, named the USING sensor (User-Smartphone-Decoupling Detection). The USING sensor relies on the user activity sensing and co-existence monitoring of users and smart mobile devices. It could track the continuous coupling state of the user-mobile device. However, continuous user activity detection is not trivial, since some actions, such as sitting, dropping, and so on, intermittently occur while some actions like walking are frequent and continuous. So, data acquisition methods from a mobile device for the detection of user actions should be designed and managed differently. In addition, for high sensing reliability, the USING sensor is based on machine learning (ML) analytics with multiple algorithms and the ensemble technique. Thus, it could achieve 95.0% of user activity recognition accuracy and 86.8% of sensing accuracy.

INDEX TERMS Virtual sensor, mobile sensing, user activity monitoring.

I. INTRODUCTION

As smartphone usage increases day by day, smartphones have become one of the most common tools in our lives. Sensors in smartphones have also been increasing. The performance of these sensors improves over time, and the range of fields in which they are used is also increasing [1], [2], [3]. In particular, services that utilize the user's location have become one of the most popular services, such as navigation [4], [5], rescue [6], [7], and tracking [8], [9]. Therefore, the method of measuring the user's location has become important. Research on positioning has been continuously studied, such as the use of GPS [10], wireless signals [11], [12], and light [13], [14]. However, the location mentioned in the above research does not indicate the user's actual location but rather the location of the mobile device the user is believed to have. To use the smartphone's location as the user's location, it is necessary to assume that the user has a smartphone. But the way to ensure that is lacking. If the system using the user's location cannot check a decoupling status between the user and the smartphone, it may provide incorrect information. Especially, this mismatching about the location can be a more critical issue in a disaster situation where the rescue teams find the victim using the smartphone's signal such as GPS. In such scenarios, it's difficult for rescue teams to determine whether the smartphone is with the victim or not. Consequently, if the smartphone's location diverges from the user's actual location, it can result in the excessive consumption of human and time resources.

The decoupling status between the user and the mobile device can be used to check the location matching. To acquire the status, it is essential to define the situation in which the user decouples with the smartphone. We define it as when the user is in physical contact with the smartphone. In this case, it can be said that the smartphone's location reliably is considered the user's location. For instance, there is a situation where the user is walking with the smartphone in his hand. Conversely, a situation in which the user and the smartphone are not in physical contact is defined as a situation in which the user decouples with the mobile device.

The associate editor coordinating the review of this manuscript and approving it for publication was Adamu Murtala Zungeru.

In this case, the mobile device's location cannot be used as the user's location. For example, there is a situation where the user drops the smartphone or puts it down on a desk. It is also said that the activity measured by the mobile device, such as walking, put down, and etc.., can be used to detect decoupling.

For the decoupling detection based on the user's activities, the smartphone must be able to recognize various activities. In the research [15], authors thought that decoupling between the user and smartphone could affect the reliability of smart device-based localization. In other words, in a system that utilizes the user's location measured through a smartphone, decoupling detection is necessary for system reliability. To detect activity for decoupling detection, it must consider how to obtain sensor data representing specific activities at random times

The sensor data representing the activity is categorized into two types. One type is the activity that occurs continuously, such as walking and running. Another type is intermittently, such as drop and put down. When classifying activities using sensor data, continuous activities make it easy to extract feature data, which well represents the activity, from the data flow. However, in the case of intermittent activity, it should be considered to detect feature data occurring at random times. The feature data must well mirror the characteristic of the intermittent activity to classify with another activity.

Considering the above issues, this paper proposes a novel decoupling detection system. The system determines the decoupling between the user and the smartphone using activity recognition that uses the only accelerometer sensor of the smartphone. The system expertly detects both continuous and intermittent activities. Therefore, we introduce new methods specialized for detecting both types of activities. These methods are monitor mode and detector mode. Monitor and detector modes detect continuous and intermittent conditions respectively. Depending on activity recognition, the proposed decoupling algorithm detects the decoupling status. To measure the performance of the system, we actually collect the activity data. Additionally, we created an application that implemented monitor mode and detector mode, and used it to conduct experiments assuming decoupling situations in various environments.

The main contributions of the paper are the following.

1) The system uses only the accelerometer sensor to recognize human activity. It reduces the usage of smartphone resources compared to using two or more sensors. Additionally, it only uses the smartphone without any additional devices.
2) To extract the feature data of human activity, we propose monitor and detector modes. Using two modes, the continuous and intermittent activity feature data can be detected and extracted. It improves the quality of the data and the performance of machine-learning models that are data-driven approach.
3) Because the decoupling decision in the system depends on human activity recognition, increasing the accuracy of the recognition is important. By using machine-learning models, preprocessing, and ensemble techniques, we increase the accuracy of human activity recognition. Also, we experiment with the decoupling decision using an application that has monitor and detector modes.

II. RELATED WORKS

A. USER LOCALIZATION

Research in the field of measuring a user's location is one of the most popular. There are various methods for measuring a user's location. One of the common approaches involves using wireless signal [11], [16], [17]. In [17], information about the user's orientation is used to enhance localization performance through received signal strength (RSS). It uses the mobile behavior similarity of a user to measure the user's location with a time-dependent Markov prediction model [18]. The location is found using data obtained from the smartphone in these methods. Therefore, it is difficult to say that the user's location has been found. In other words, this location represents only the smartphone's location, not the user's location. To ensure that this location represents the user's whereabouts, we use the decoupling status between the user and the mobile device

B. ACTIVITY DETECTION AND RECOGNITION

TABLE 1. Comparison researches for activity detection, recognition, and decoupling detection.

Type ofsensor data	Activity detection	Activity Recognition	Decoupling detection	Ref.
Inertial	X	O	X	in [23]
Kinetic	X	O	X	in [19]
Kinetic	O	O	X	in [20]
CSI	O	O	X	in [21], [22]
Inertial Image	O	O	X	in [24]
Inertial Orientation Magnetic	X	O	O	in [15]
Accelerometer	O	O	O	USING sensor

Human activity can be used to determine the decoupling status. Fortunately, the research of activity recognition is one of the popular fields today. They recognize the activity using a variety of data and methods. Skeleton data is commonly used to recognize human activity. In [19], authors use a directed acyclic graph (DAG) based on the kinematic dependency between the joints and bones to represent the skeleton data. Based on DAG and directed graph neural networks, the activities are classified. In [23], inertial sensors and long short term memory (LSTM) are used to classify the activities. However, they do not consider the challenge of detecting feature data that accurately represents activities occurring at random times. While test and train data might be well-represented based on collection methods, the additional method that detects the feature data should be required in a continuous data stream. Implementing activity detection capable of accurately capturing the characteristics of activities occurring randomly becomes essential.

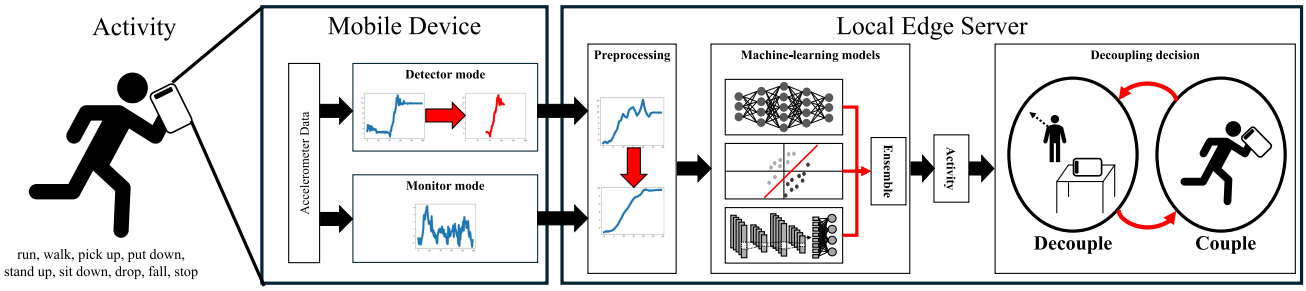

FIGURE 1. USING sensor system overview.

Activity recognition studies that consider activity detection have proceeded. In [20], the authors employ a segmentation process to detect human activity. Also, the maximum entropy markov model is used to classify the activity depending on likelihood probabilities. In [21], channel state information (CSI), which is a wireless channel property of communication links and changes due to human activity, is used to detect activity and classify it with a faster region-based convolutional neural network (RCNN). In [22], the start and end points of the activity are detected using CSI. Then, they use gaussian mixture model-hidden Markov model (GMM-HMM) to recognize the activity. In [24], authors use the image and the inertial data to detect and classify the activity. Although skeleton, CSI, and image data are common sources that detect activity, they are difficult to use in daily life. This is because CSI requires additional equipment to obtain data, and skeletons and images have the problem of violating the user's privacy.

In [15], the authors discuss that decoupling between users and mobile devices could affect the reliability of smart device-based localization. They use human activity recognition to make decoupling status. Although this is a similar approach to our system, there are no considerations for detecting activity as mentioned earlier. Table. 1 summarize the activity recognition research compared with USING sensor.

We propose USING sensor for determining decoupling status that ensures that the mobile device's location can be used as the user's location. To determine decoupling status, we rely on human activity using the smartphone without any additional equipment. Furthermore, methods to detect the activity are designed to detect two types of activity that occur continuously or intermittently.

III. SYSTEM MODEL

In this section, we describe a USING sensor system that detects decoupling between the user and the mobile device using the accelerometer in the mobile device.

The system's main purpose is to detect decoupling between the user and the mobile device using the activities that are measured by the accelerometer sensor data. Fig. 1 shows an overview of the system. The system comprises users, mobile devices, and the local edge server. In the system, users perform various activities such as running, walking, picking up mobile devices, and sitting. Mobile devices collect accelerometer sensor data and send it to the local edge server. Before data is sent to the server, mobile devices use two types of data management methods: monitor mode and detector mode. Monitor mode is specialized for detecting continuous activities. Because the continuous activity continuously occurs, the monitor mode regularly sends the data to the server without any process. On the other hand, the detector mode is specialized for detecting intermittent activities. So, the detector mode detects the intermittent activity as well as extracts the feature data, which well represents the intermittent activity.

In the local edge server, the received data from the mobile device is used to measure the activity. The data is preprocessed to improve the accuracy of an activity classification. Machine-learning models and ensemble technique uses the preprocessed data to measure the activity. Depending on the measured activity, the system determines the decoupling status between the user and the mobile device.

Classifying the activity into continuous or intermittent is an important point in the system. Because the system detects and classifies the activity based on the types of activity, two scan types are proposed. Table. 2 provides information about scan type according to the activity type. Scan type indicates the methods to detect activities that are categorized into continuous and intermittent. The monitor mode is suitable for classifying continuous activity, including persistent activities such as stopping, walking, and running. Characteristics of continuous activity are that they occur continuously and repeatably. In contrast, the detector mode is designed to classify intermittent activities. These intermittent activities include 'fall down', 'stand up', 'sit down', 'pick up', and 'drop'. Characteristics of intermittent activity are that they occur in a short and random time. Therefore, a detector algorithm to detect and extract feature data from them is required. Detailed explanations of the detector and monitor mode are described in the framework subsection.

A. FRAMEWORK

Fig. 2 illustrates the framework that shows the process of detecting activity and decoupling status between the user and the mobile device. The framework consists of 3 modules: data aggregation, ML-aided activity decision, and decoupling detection.

TABLE 2. User activity classification table.

Scan Type	Activity Type	Activity
Monitor	Continuous	Stop
		Walk
		Run
Detector	Intermittent	Fall down
		Stand Up
		Sit Down
		Pick Up
		Put down
		Drop

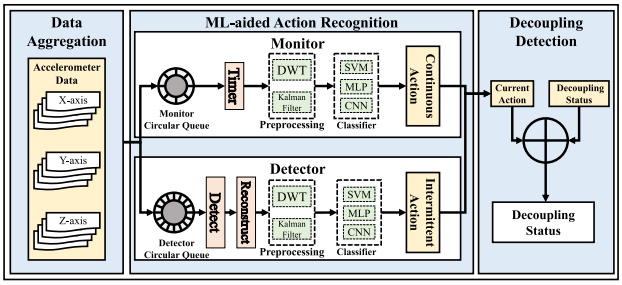

FIGURE 2. Framework for decoupling detection using activity recognition.

The data aggregation module is responsible for managing accelerometer sensor data. Subsequently, this data is forwarded to the ML-aided activity decision module.

The ML-aided activity decision module classifies various activities, which are categorized as continuous and intermittent activities. To classify continuous activity, the monitor mode is used. In this mode, data in the monitor circular queue is regularly classified using a timer. Before classification, the data may be applied to two preprocessings which are Kalman filter and Discrete Wavelet Transform (DWT). After preprocessing, activities are classified using classifiers such as multi-layer perceptron (MLP), 1D convolutional neural networks (1D-CNN), and support vector machine (SVM). On the other hand, the detector mode has distinct features compared to the monitor mode. The detector mode is specialized to capture a specific period in which intermittent activity occurs. The detector mode verifies whether data related to intermittent activities is present in the 'Detect' process. If data related to the intermittent activity is found in this process, the ''Reconstruct'' process is initiated. During the ''Reconstruct'' process, feature data that represents the activity is extracted in the detector circular queue. Following this, similar to monitor mode, activities are predicted through preprocessing and classifier.

The decoupling detection module determines the decoupling status between the user and the smartphone. The current activity, which is measured from the previous module, and the decoupling status are used to detect the status. When the user possesses the smartphone, this is called the couple state. In this status, the smartphone can detect the user's activities. When a specific activity, such as pick down, occurs, the system changes the state to decouple state from couple state. The decouple state indicates that the user does not possess the smartphone. There are also some activities that change to the couple state from the decouple state. Detailed methods for determining the status are discussed in section IV.

Algorithm 1 Decoupling Detection Algorithm

Input: Prior Decoupling Status (PDS),
Current Activity (CA)
Output: Decoupling status

```
if CA == PDS's Decoupling activity then
    return Decoupling
else if CA == PDS's Coupling activity then
    return Coupling
else if CA is PDS's Impossible activity then
    return No change
end if
```

TABLE 3. Decoupling decision based on prior and current activity.

Prior Status	Decoupling activity	Coupling activity	Impossible activity
Couple	Fall down Put down Drop	Walk Run Stand up Sit down Stop	Pick up
Decouple	Stop	Pick up Stand up	Drop Fall down Put down Sit down Walk Run

IV. DECOUPLING DETECTION ALGORITHM

In this section, the decoupling detection algorithm based on the measured activity is introduced. Algorithm. 1 shows a method to determine the decoupling status based on the prior status and the measured activity. For example, if the put down activity occurs in the coupling status, the status changes to the decoupling between the user and the mobile device. If the pick up activity occurs in the decoupling status, the status changes to the coupling from the decoupling. Table. 3 shows the detailed decoupling decision using prior status and measured activity. Decoupling activity in Table. 3 indicates the activity that changes the status to decoupling. Coupling activity means the activity that changes the status to coupling.

The algorithm has the ability to perform accurate decoupling detection even when there are errors that may occur in activity recognition. The decoupling, for example, occurs when the user put down the smartphone. Unfortunately, the activity recognition measures that the current activity is drop activity. In this case, even if the measured activity is incorrect, the results of decoupling detection will be determined by the decouple state. This is because the algorithm determines the decouple state when drop activity occurs in the couple state. In this way, the algorithm can flexibly respond to activity recognition errors in certain situations.

V. MONITOR MODE AND DETECTOR MODE

In this section, we introduce the monitor and detector modes that are used to detect continuous and intermittent activities.

Sensor data representing user activities can be categorized into two types: continuous activity and intermittent activity.

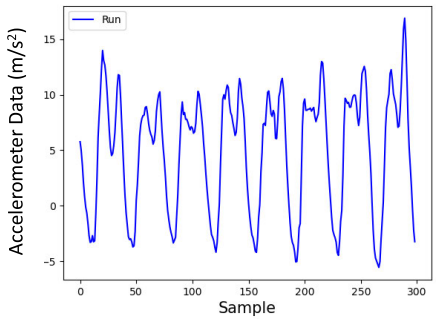

FIGURE 3. Accelerometer data samples about continuous activities.

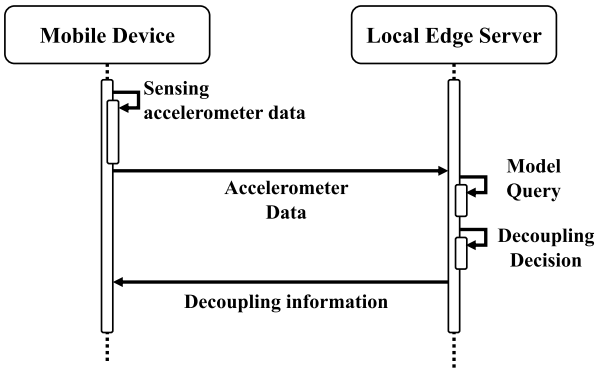

FIGURE 4. Sequence diagram about monitor mode operation.

Continuous activity occurs continuously, such as walking and running. Fig. 3 shows examples of accelerometer sensor data of continuous activities. We identify that data of continuous activity is generated repeatedly during the activity. In other words, a characteristic of continuous activity is the repetitive and similar trends. Consequently, there is no need to consider extracting a specific range as representative of the activity. The sequence diagram of monitor mode using the feature of continuous activity is shown in Fig. 4. The monitor mode periodically sends data to the server. Because of the persistence of the continuous activity, the monitor mode does not require the additional process to find the specific range that well represents the activity. If the server receives the data, machine learning models in the server and the data are used to classify the user activity corresponding to the data. After that, the server determines the decoupling status between the user and the mobile device based on classified activity.

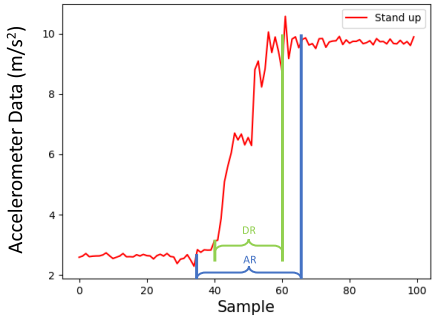

FIGURE 5. Accelerometer data samples about intermittent activities.

Fig. 5 shows an example of intermittent activity. Unlike continuous activity, intermittent activity only occurs once in a short time. To get the feature data of intermittent activity, the methods to recognize the intermittent activity and extract the range that well represents the activity must be considered. So, we propose another mode, called detector mode, for extracting feature data as well as detecting the activity. The detector mode is designed to detect feature data at specific times when intermittent activity occurs.

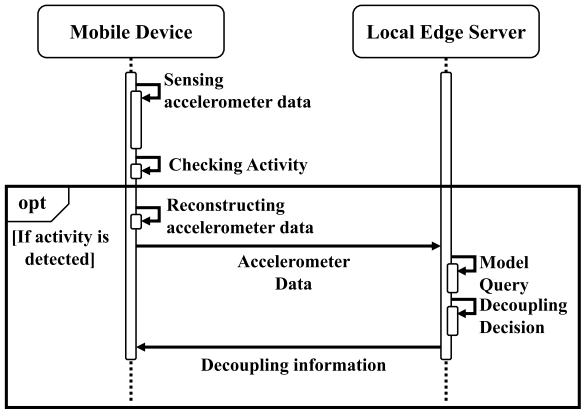

FIGURE 6. Sequence diagram about detector mode operation.

Fig. 6 shows the sequence diagram for the detector mode. It is worth noting that the sensing data is not always sent to the server. The sensing data is sent to the server only when activity is detected. After the server receives the data, the process to detect the decoupling status is the same as in the monitor mode.

The detailed process to detect the user activity using sensor data is shown in Algorithm. 2. Notations that are used in the algorithm are described in Table. 4. In the system. the mobile device uses the algorithm and determines whether the intermittent activity is detected or not. First, the sensor data, which is collected by the accelerometer in the mobile device, fills out the detector queue (DQ) until DQ is full. Then, the algorithm checks whether the sensor data corresponding to the intermittent data is in the detector queue. This process is shown in lines 7 through 19 of the algorithm. The change value of the sensor data is used to detect the intermittent activity. The change value is the difference between the [i]-th data in DQ and [i + detect range value (DR)]-th data. If the change value is over the threshold (Th) that is set by the system, i-th data is a starting point to occur the intermittent activity. In other words, intermittent activity is detected from the detector queue. The above process uses all data in DQ to find the index that has the greatest change value. The system extracts the data from the *index* to activity range value (AR). Also, we must consider cases where intermittent activity occurs at the end of DQ and the sensing data does not fully reflect it. So, if the index is lower than the length of DQ - 2 $\times$ DR, the system checks the DQ again after collecting the next 2 $\times$ DR pieces of data. If intermittent activity is not detected, The system checks DQ again after collecting data

Algorithm 2 Algorithm to Detect the Intermittent Activity

```
Start to sensing.
count = 0
while True do
    DQ_count = sensor data
    count += 1
    if count == Len(DQ) then
        Detect = False
        Index = 0
        maximum = 0
        for i = 0 to Len(DQ) - DR do
            Data = DQ_i
            if | Data - Next(DR)_Data | ≥ Th then
                if |Data - Next(DR)_Data| ≥ maximum then
                    Detect = True
                    maximum = |Data - Next(DR)_Data|
                    Index = i
                end if
            end if
        end for
        if Detect == True then
            if Index ≤ Len(DQ) - 2 × DR then
                count = 0
                return DQ_index:index+AR
            else
                count = Len(DQ) - 2 × DR
            end if
        else
            count = Len(DQ) × 1/2
        end if
    end if
end while
```

TABLE 4. Terms for detector algorithm.

Term	Definition
DQ	It is detector queue
DQ_i	It is i-th sensor data in the detector queue.
$DQ_{a:b}$	It is the data in the range a to b in DQ.
Th	It is the threshold to detect intermittent activity
DR	It is the range to check the change in DQ
AR	It is the range to represent the intermittent activity in DQ
$Next(k)_{data}$	It is the next k-th data after $data$

half the length of DQ. The reason for collecting data that is only half of the length of DQ is to increase the system response speed.

Setting the algorithm's parameters is an important issue in the system. Because the detector mode detects intermittent activities and extracts the feature data from them, the feature data can be different according to the parameters. Before setting the parameters, we collect the accelerometer value for each intermittent activity. As a result, 30 accelerometer values are typically collected during each intermittent activity. So AR, which is the feature data of the activity, is set to 30. In the algorithm, DR is used to detect the activity. We confirm that approximately 20 of the 30 feature data are the ranges where sensor values changed most rapidly. Because the algorithm detects the activity using a change in the sensor value, DR is set to 20, which can represent the largest change. Figure. 5 shows the ranges of AR and DR. The algorithm compared the calculated change in the sensor value and Th. When the chance is over the Th, the algorithm recognizes the activity is detected. If Th is so small, the algorithm can detect all the activity. However, the algorithm may recognize that it has detected an intermittent activity even though it has not occurred. Contrary, If Th is so big, the algorithm cannot detect the activity. So, it is important to set the appropriate Th. we found that when Th is 3, there are cases where the detector does not recognize the intermittent activity. This means that the maximum value of Th that the algorithm can properly recognize intermittent activity is 3. So, Th is set to 3.

VI. ACTIVITY CLASSIFICATION

In this section, we explain methods to classify various activities with accelerometer sensor data. Machine-learning models, such as MLP, 1D-CNN, and SVM, are used as classification models. Additionally, preprocessing methods, such as Kalman filter and DWT, are used to reduce noise in the sensor data. We apply the ensemble technique to improve classification performance with preprocessing methods and classification models.

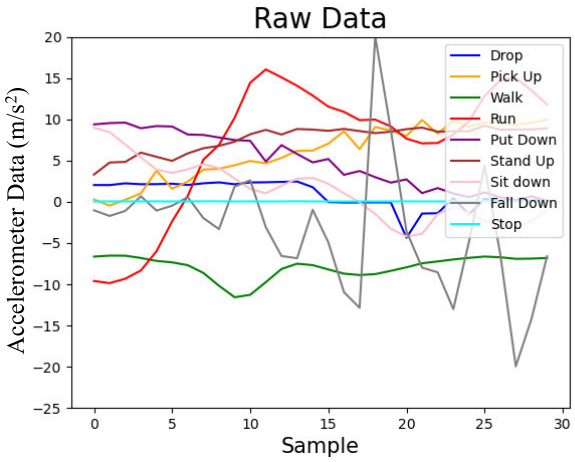

FIGURE 7. Raw sensor data.

A. PREPROCESSING METHODS

Preprocessing methods are applied to reduce noise in the data. Two common preprocessing methods utilized are the Kalman filter and Discrete Wavelet Transform (DWT). Kalman filter and DWT are often used for data prerprocessing [25], [26]. We implement the Kalman filter and DWT with Python and provide a description of how to implement them.

1) KALMAN FILTER

Kalman filter consists of two procedures. The first procedure is called a prediction. The prediction procedure calculates the predicted activity estimate and predicted estimate covariance.

This procedure is represented in equation (1).

$$\hat{-}_k^- = A\hat{-}_{k-1}$$
$$P_k^- = AP_{k-1}A^T + Q \tag{1}$$

In equation (1), $\hat{-}_k^-$, P_k^-, A, Q are the estimated value at time k, estimate error covariance, the activity transition matrix, and noise covariance matrix, respectively.

The second procedure is called an update. This procedure consists of calculating Kalman gain, $\hat{-}_k$, and error covariance. These parts are shown in equation (2), equation (3), and equation (4).

Calculate the Kalman gain (K_k):

$$K_k = P_K^- \hat{H}(HP_k^-\hat{H} + R)^{-1} \tag{2}$$

Calculate the predicted value ($\hat{-}_k$):

$$\hat{x_k} = \hat{x_k}^- + K_k(z_k - H\hat{x_k}^-) \tag{3}$$

Calculate the predicted error covariance (P_k):

$$P_k = P_k^- - K_k H P_k^- \tag{4}$$

In these equations, K_k, P_k, x_k, z_k, H, and R are Kalman gain, predicted error covariance, predicted value, measured value, observation matrix, and noise covariance matrix, respectively. Kalman filter is used to denoise data. we use the Kalman filter to remove noise in sensor data. Fig. 8 is shown data with Kalman filter.

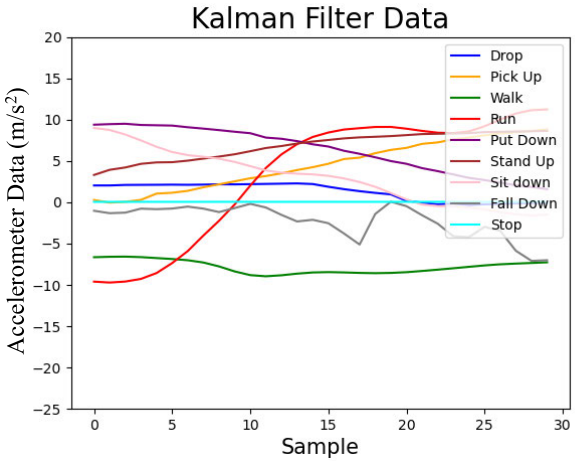

FIGURE 8. Sensor data with Kalman filter.

2) DISCRETE WAVELET TRANSFORM (DWT)

DWT, like the Kalman filter, is also used to remove noise from sensor data. First, DWT decomposes sensor data by using a low pass filter ($g[x]$) and a high pass filter ($h[x]$). As a result of the decomposition process, y_{low} that is passed by the low pass filter and y_{high} that is passed by the high pass filter are obtained. An equation of the decomposition procedure is shown in equation (5).

Decomposition Procudure:

$$y_{low}[n] = \sum_{k=-\infty}^{\infty} x[k]g[2n-k]$$

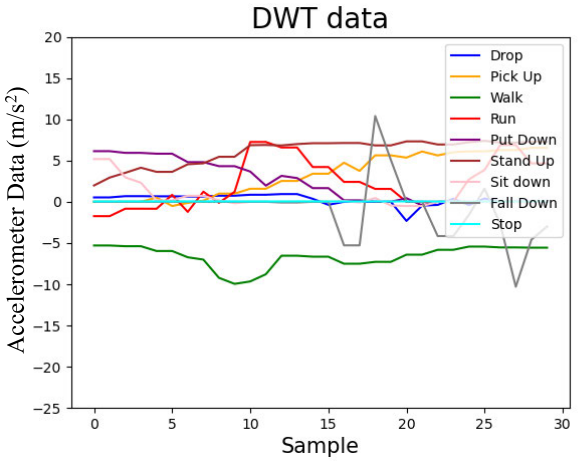

FIGURE 9. Sensor data with DWT.

$$y_{high}[n] = \sum_{k=-\infty}^{\infty} x[k]h[2n-k] \tag{5}$$

In the above equation, x is the original sensor data set. y_low and y_high are the values after x passes through the high pass filter and low pass filter.

After the decomposition procedure, it applies thresholds at y_{low} and y_{high} to remove noise in the data. A denoising procedure uses a standard deviation of y_{low} and y_{high}. The denoising procedure is shown in equation (6).

Denoising Procedure:

$$y_{low}[n] = \frac{y_{low}[n]}{|y_{low}[n]|} \times (Maximum((|y_{low}[n]| - Standard(y_{low}[n]), 0)))$$
$$y_{high}[n] = \frac{y_{high}[n]}{|y_{high}[n]|} \times (Maximum((|y_{high}[n]| - Standard(y_{high}[n]), 0))) \tag{6}$$

$Standard(x)$ means the standard deviation value of x. $Maximum(a, b)$ means the maximum of a and b. After denoising procedure, y_{low} and y_{high} are combined. The combined data becomes data from which noise is removed from the original data. Fig. 9 is shown data with DWT.

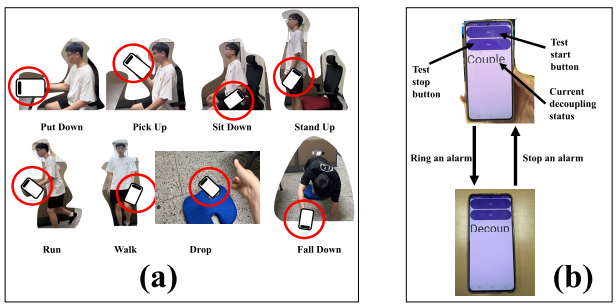

FIGURE 10. (a): The smartphone's position when collecting activity data. (b): Smartphone application for decoupling detection test.

B. CLASSIFICATION WITH ENSEMBLE TECHNIQUE

Machine-learning models are often used as classifiers in various fields such as distributed denial-of-service (DDoS) attack detection [29], gas classification [30], motion classification [31], sports judge [32], and positioning [33]. Epsectually, the human activity recognition field has used many types of machine-learning models, such as LSTM [34], CNN [35], and SVM [36]. Machine-learning models with a data-driven approach are very convenient methods to predict something with high-quality data. By using the feature data, which is high-quality data acquired using monitor and detector mode, we expect an improvement in the model's performance.

The ensemble technique is used to improve classification performance with preprocessing methods. The ensemble is a technique that combines the results of individual classification models to improve the overall performance and accuracy. It is often used to improve classification performance of various fields [27], [28]. In the proposed system, there are a total of 27 classification models, resulting from the combination of 3-axis data (x-axis, y-axis, z-axis), 3 preprocessing methods (no-preprocessing, Kalman filter, DWT), and 3 classification models (MLP, 1D-CNN, SVM). In the case of the MLP and 1D-CNN, the results have the probability of all activities. However, the result of SVM does not have a probability. So, we set the probability for the result of the SVM to 0.3 and the probabilities that are not measured as the result to 0. With the results of the classification models and soft voting, the final results are calculated.

VII. EXPERIMENTAL ENVIRONMENT

TABLE 5. Specifications of machine-learning models for training.

MLP Specification								
Layer type	Input layer	Hidden Layer	Hidden Layer	Hidden Layer	Output Layer	-	-	-
Layer size	30	64	32	16	9	-	-	-
Activation function	-	relu	relu	relu	softmax	-	-	-
Batch size	8							
Epoch size	100							

1D-CNN Specification								
Layer type	Input layer	Hidden Layer	Hidden Layer	Hidden Layer	Hidden Layer	Hidden Layer	Hidden Layer	Output Layer
Layer size	30	Kerenl: 4 Filter: 4	Max pooling	Kernel: 4 Filter: 3	Flatten	32	16	9
Activation function	-	relu	-	relu	-	relu	relu	softmax
Batch size	8							
Epoch size	100							

TABLE 6. Datasets used to train and evaluate each activity recognition model.

	Drop	Pick up	Put down	Stand up	Sit down	Fall down	Run	Walk	Stop
Train	100	120	120	110	119	100	100	100	100
Evaluation	80	80	80	80	80	80	80	80	80

In this section, we explain an experimental environment where the activity data is collected and the system is tested.

To measure activity recognition performance, acceleration sensor data for activity are collected. The activities to be classified are set as stop, put down, pick up, sit down, stand up, run, walk, drop, and fall down. We collect accelerometer data on each activity with the application we made. For collecting training data and testing, Galaxy 7 and Galaxy 21 are used. A sensing delay of mobile devices is set to 60ms.

Fig. 10 (a) shows what it looks like when data for each activity is collected. The red circles are points where the smartphone is located. When we train machine-learning models using intermittent activity data, the detector mode is utilized to extract the feature data from collected data. The settings for detector mode are *Th*: 3, *DQ* size: 300, *DR*: 20, and *AR*: 30 for collecting training data. The continuous activity data is collected using the monitor mode. These data are used to train machine-learning models such as MLP, 1D-CNN, and SVM. The number of training and evaluation data for each activity is shown in Table. 5. Half of the evaluation data is used as validation data and the other half is used as test data. As a result, the ratio between training, validating, and testing data is approximately 6:2:2.

When we conducted the decoupling detection test, a mobile application that implemented monitor and detector modes is used. The settings of detector mode in the application are *Th*: 3, *DQ* size: 100, *DR*: 20, and *AR*: 30. In the case of the monitor mode, every time 30 pieces of data are collected, they are sent to the server. The discrepancy in the size of *DQ* between the training and testing phases can be attributed to distinct objectives. During training, *DQ* is configured with a larger capacity to guarantee the inclusion of all relevant activities. Conversely, during testing, *DQ* is deliberately set to a smaller size relative to training to reduce the data collection process while maintaining efficiency. Fig. 10 (b) shows the application that is used to test the decoupling detection. The application rings an alarm when the decoupling status is detected. On the other hand, if the status changes to the coupling from the decoupling, the alarm stops. Because the *DQ* size is 100 and the data collection speed is 60ms, we wait up to 6 seconds and determine the test result by checking whether the alarm sounds or not.

To implement the machine-learning models and DWT, we utilized Ubuntu version 18.04, Python version 3.7.12, TensorFlow library version 2.10.0 which is used to implement MLP and CNN models, sklearn version 1.0.2 that is used to implement SVM model, and PyWavelet version 1.3.0 that used to implement DWT.

The structures and specifications of machine-learning models are shown in Table. 5. In the case of MLP models, the hidden layers and output layer use Dense layer provided by TensorFlow library. In the case of CNN models, the first and third hidden layers use Conv1D layer. The second hidden layer uses Maxpooling layer and the fourth hidden layer uses Flatten layer. The rest of hidden layer and the output layer use Dense layer. The layers used in CNN models are also provided by TensorFlow library. In the case of SVM models, a linear kernel is used for training.

VIII. PERFORMANCE EVALUATION

A. PERFORMANCE EVALUATION FOR THE DETECTOR MODE

Detector mode is used to extract a feature of intermittent activity, which occurs once at random times. So, the

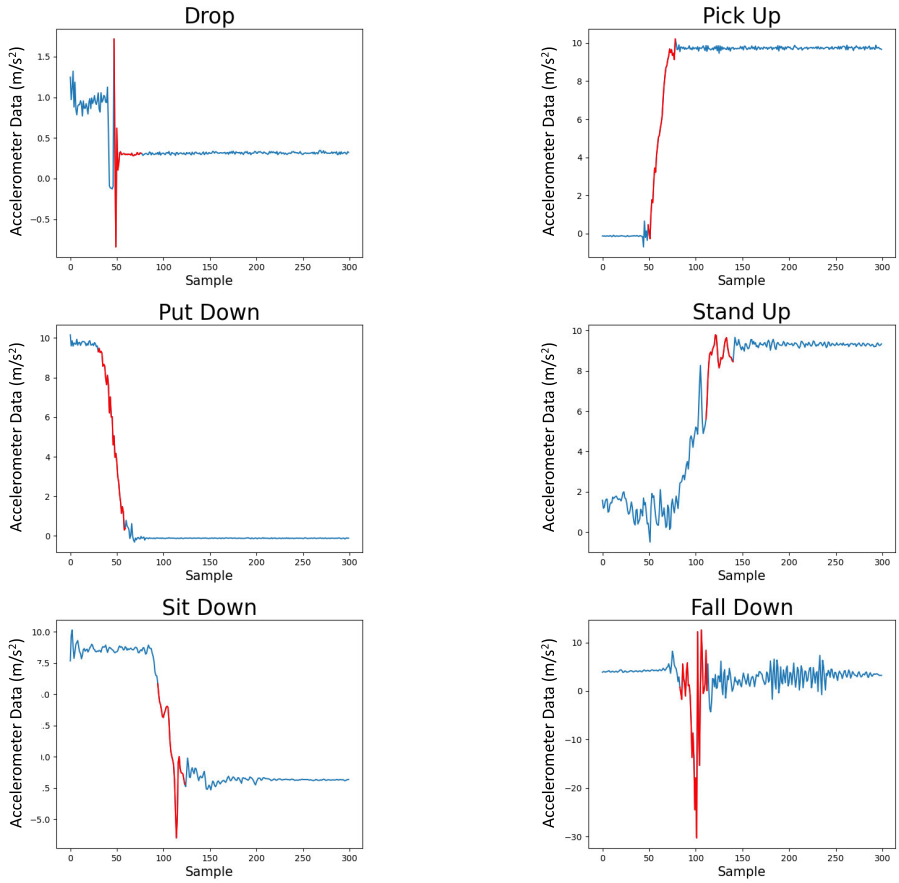

FIGURE 11. It shows the detected feature data using the detector mode from the intermittent data flow. The blue line represents the original data flow, while the red line signifies the feature data within the data flow.

performance of the detector mode is determined by how to extract the feature representing the activity. In other words, standards to estimate the performance of the detector mode are a recognition ability and a similarity extracted from the same activity data. The recognition ability is evaluated by how well one recognizes intermittent data in data flow. An experiment was conducted in detector mode to recognize six intermittent activities based on the X-axis, Y-axis, and Z-axis accelerometer data. The results of the experiment are shown in Table. 7. It shows that the detector algorithm using Z-axis data very well detects intermittent data. So, the detector mode uses the Z-axis data to detect intermittent activities in the experiments that follow. Fig. 11 shows the detected feature data of each intermittent activity data using the detector mode. Red lines indicate detected feature data that the detector mode detected. As a result, the detector mode algorithm can detect a specific time when intermittent activities occur.

TABLE 7. Probability of detecting intermediate activity.

	Drop	Pick Up	Put Down	Stand Up	Sit Down	Fall Down
X-axis	69%	40%	22%	60%	100%	83%
Y-axis	53%	100%	100%	100%	100%	83%
Z-axis	100%	100%	100%	92%	99%	100%

To evaluate the similarity extracted from the same activity data, we calculate the Euclidean distance among feature data from the same activity. Table. 8 shows the Euclidean distance among feature data from the same activity. Fig. 12 shows the 100 feature data extracted for each activity. It can be seen that similar features are extracted from the same activity.

TABLE 8. Euclidean distance of detected feature data.

	Drop	Pick Up	Put Down	Stand Up	Sit Down	Fall Down
X-axis	27.96	6.45	6.14	15.62	25.44	50.54
Y-axis	16.19	8.90	7.79	11.51	13.93	44.41
Z-axis	60.09	13.32	12.11	14.04	22.91	138.88

B. PERFORMANCE EVALUATION FOR ACTIVITY CLASSIFICATION

Machine-learning techniques, preprocessing methods, and ensemble techniques are used to improve classification accuracy. The classification models categorize nine activities, comprising six intermittent activities (drop, pick up, put down, stand up, sit down, and fall down) and three continuous activities (walk, run, and stop). For intermittent activities, the classification models are trained using feature data acquired by the detector mode algorithm based on Z-axis data.

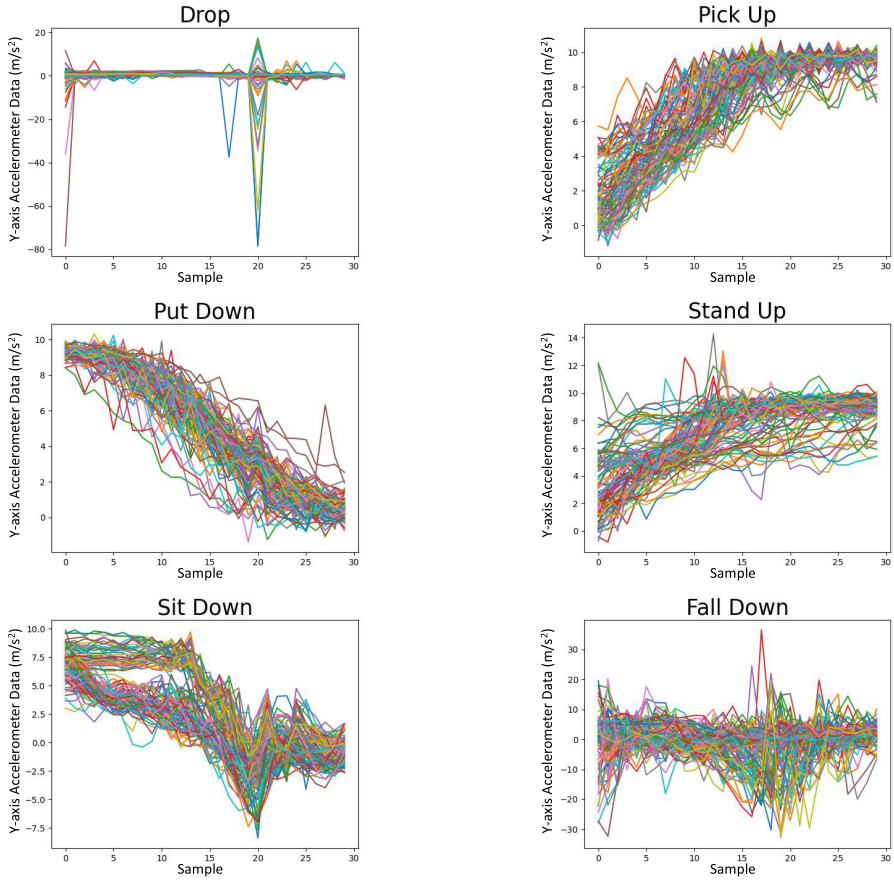

FIGURE 12. It is a set of Y-axis data extracted by the detector mode algorithm based on Z-axis data. There are 100 samples of each intermittent activity.

There are a total of 27 classification results, obtained through the combination of three-axis data (x-axis, y-axis, z-axis), three preprocessing methods (no-preprocessing, Kalman filter, DWT), and three classification models (MLP, 1D-CNN, SVM). Without any preprocessing, the highest accuracy is achieved 85.4%. With the Kalman filter, the highest accuracy is achieved at 87.1%.

With DWT, the accuracy is achieved at 84.2%. Consequently, using a single classification model, the accuracy can be achieved over 84.0%.

Additionally, we use the ensemble technique to improve accuracy. The ensemble technique uses the results of the above 27 classification models with soft voting. As a result, the accuracy of activity classification could be improved to 95% with the ensemble technique. It is confirmed that when the ensemble technique is used, accuracy improves by approximately 13.1% compared to when it is not used.

Fig. 13 shows the confusion matrix about activity classification using the ensemble technique. It shows an accuracy of over 88% for all behavior classifications. However, in the case of the stand up and the sit down activity, the accuracy is lower than other activities. We think this is because the data for these two activities was collected with the smartphone in the pocket, so there was irregular shaking compared to other activity data. Nevertheless, by using the ensemble technique, a high accuracy of over 88% was achieved.

C. PERFORMANCE EVALUATION FOR DECOUPLING DETECTION

	Drop	Pick up	Put down	Stand up	Sit down	Fall down	Run	Walk	Stop	Recall
Drop	73	0	0	0	0	7	0	0	0	91%
Pick down	0	79	0	2	0	0	0	0	0	98%
Put down	1	1	80	0	1	0	0	0	0	96%
Stand up	0	0	0	70	3	0	0	0	0	96%
Sit down	0	0	0	8	70	0	0	0	0	90%
Fall down	6	0	0	0	0	73	1	0	0	91%
Run	0	0	0	0	0	0	79	0	0	100%
Walk	0	0	0	0	6	0	0	80	0	93%
Stop	0	0	0	0	0	0	0	0	80	100%
Accuracy	91%	99%	100%	88%	88%	91%	99%	100%	100%	

FIGURE 13. Confusion matrix for activity classification using ensemble.

The decoupling status is determined based on the measured activity and prior status. To evaluate the performance of the decoupling detection algorithm, we recreate situations where the decoupling may occur. We conducted tests by actually performing situations that change into a decoupling, such as when the user drops the smartphone or places the smartphone

on a table, and situations that change from a decoupling to a coupling, such as picking up the smartphone. Of course, the situations in which the status remains, such as the user walking with a phone or sitting down on a chair while coupled with the smartphone, are also considered. Additionally, the test is conducted at various locations were used as if actual decoupling occurred, and it included not only cases where one activity occurred but also cases where several activities occurred sequentially.

TABLE 9. Confusion matrix for the results of decoupling detect test.

		Actual	
		Decouple	Couple
Measured	Decouple	TP = 38	FN = 6
	Couple	FP = 6	TN = 41

Table. 9 shows the confusion matrix for the results of the test. Accuracy of the decoupling detection is achieved at 86.8% with the system. Because the decoupling detection test uses the confusion matrix, True Positive (TP), False Negative (FN), False Positive (FP), and True Negative (TN) values can be obtained. Based on the obtained values, we calculate F1-score to evaluate the decoupling detection in more detail. The following are the formulas used to calculate the F1 score.

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

$$F1score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{9}$$

F1-score is calculated at 0.864. Although the decoupling detection uses the results of the activity recognition, the performance of decoupling detection is lower than the activity recognition. This is because the decoupling detection test is conducted in more diverse environments than the activity recognition test. For example, the smartphone may be picked up and walked around, and other times it may be left on desks of various heights. Therefore, even when the same activity is performed, slightly different data that makes wrong recognition can be obtained. We believe that results obtained in situations similar to reality are more valuable and reliable.

IX. CONCLUSION

In this paper, we propose a novel decoupling detection system with a decoupling detection algorithm with mobile-based virtual sensor. The algorithm uses activity recognition that uses only accelerometer sensor data to determine the decoupling status between the user and the smartphone. The activities are categorized into two types: continuous activity and intermittent activity. In order to extract feature data of each activity, a monitor and detector modes are used. In the case of the monitor mode, the features of continuous activity, which occurs periodically and continuously, are extracted at regular times. In the case of the detector mode, an action detect algorithm is used to extract features of intermittent activity, which occurs at random times. Based on the machine-learning models and extracted feature data from two modes, the system measures the activities to determine the decoupling status.

Because the decoupling detection uses action recognition, activity recognition accuracy is important in the system. Thus we employ preprocessing techniques and ensemble techniques to improve the accuracy. As a result, the ensemble model can achieve an accuracy of 95%. Also, the decoupling detection accuracy can achieve 86.8% using the ensemble model. For testing the decoupling detection, we use a hand-made application, in which the detector and monitor modes are implemented, in various scenarios that can occur in the real world.

Consequently, USING sensor can be a solution to determine the decoupling status between the user and the smartphone in the real world. This leads to the matching of the smartphone's location and the user's location. Depending on the matching, the reliability of traditional systems, which use the smartphone's location as the user's location, is improved. Because the USING sensor does not require the additional equipment or private information of the user, it can be easily added to existing mobile-based systems that measure the smartphone's location and use it as the user's location. In addition, through a detector algorithm that detects intermittent activity, various activities can be detected with just an acceleration sensor and feature data representing these activities can be extracted. Not only does machine learning performance improve, but the data needed for learning can be collected effectively.

REFERENCES

[1] J. G. Jagüey, J. F. Villa-Medina, A. López-Guzmán, and M. Á. Porta-Gándara, "Smartphone irrigation sensor," *IEEE Sensors J.*, vol. 15, no. 9, pp. 5122–5127, Sep. 2015.
[2] O. Walter, J. Schmalenstroeer, A. Engler, and R. Haeb-Umbach, "Smartphone-based sensor fusion for improved vehicular navigation," in *Proc. 10th Workshop Positioning, Navigat. Commun. (WPNC)*, Mar. 2013, pp. 1–6.
[3] W. Elloumi, A. Latoui, R. Canals, A. Chetouani, and S. Treuillet, "Indoor pedestrian localization with a smartphone: A comparison of inertial and vision-based methods," *IEEE Sensors J.*, vol. 16, no. 13, pp. 5376–5388, Jul. 2016.
[4] L. Árvai, "Mobile phone based indoor navigation system for blind and visually impaired people: VUK—Visionless supporting framework," in *Proc. 19th Int. Carpathian Control Conf. (ICCC)*, May 2018, pp. 383–388.
[5] M. Jain, R. C. P. Rahul, and S. Tolety, "A study on indoor navigation techniques using smartphones," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Aug. 2013, pp. 1113–1118.
[6] A. Al-Sadi, H. Al-Theiabat, and F. Awad, "Smartphone-assisted location identification algorithm for search and rescue services," in *Proc. 8th Int. Conf. Inf. Commun. Syst. (ICICS)*, Apr. 2017, pp. 276–281.
[7] X. Ma, Q.-Y. Huang, and X.-M. Shu, "A new localization algorithm of mobile phone for outdoor emergency rescue," in *Proc. 3rd Int. Conf. Intell. Syst. Design Eng. Appl.*, Jan. 2013, pp. 124–127.
[8] A. Puscasiu, A. Fanca, and H. Valean, "Tracking and localization system using Android mobile phones," in *Proc. IEEE Int. Conf. Autom., Quality Test., Robot. (AQTR)*, May 2016, pp. 1–6.
[9] M. Alzantot and M. Youssef, "UPTIME: Ubiquitous pedestrian tracking using mobile phones," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2012, pp. 3204–3209.
[10] C. Moun and C. Netramai, "Localization and building identification in outdoor environment for smartphone using integrated GPS and camera," in *Proc. 4th Int. Conf. Digit. Inf. Commun. Technol. Appl. (DICTAP)*, May 2014, pp. 327–332.

[11] Y. Li and K. Yan, "Indoor localization based on radio and sensor measurements," *IEEE Sensors J.*, vol. 21, no. 22, pp. 25090–25097, Nov. 2021.
[12] H. Seo, H. Kim, T. Kim, and D. Hong, "Accurate positioning using beamforming," in *Proc. IEEE 18th Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2021, pp. 1–4.
[13] K. Abe, T. Sato, H. Watanabe, H. Hashizume, and M. Sugimoto, "Smartphone positioning using an ambient light sensor and reflected visible light," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Nov. 2021, pp. 1–8.
[14] F. Yang, S. Li, H. Zhang, Y. Niu, C. Qian, and Z. Yang, "Visible light positioning via floor reflections," *IEEE Access*, vol. 7, pp. 97390–97400, 2019.
[15] T. Yang, S.-H. Lee, and S. Park, "AI-aided individual emergency detection system in edge-Internet of Things environments," *Electronics*, vol. 10, no. 19, p. 2374, Sep. 2021.
[16] R. Faragher and R. Harle, "Location fingerprinting with Bluetooth low energy beacons," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 11, pp. 2418–2428, Nov. 2015.
[17] S. Sun, S. Li, Y. Li, B. Moran, and W. S. T. Rowe, "Smartphone user tracking by incorporating user orientation using a double-layer HMM," *IEEE Trans. Veh. Technol.*, vol. 71, no. 7, pp. 7780–7790, Jul. 2022.
[18] Y. Li, L. Lei, and M. Yan, "Mobile user location prediction based on user classification and Markov model," in *Proc. Int. Joint Conf. Inf., Media Eng. (IJCIME)*, Dec. 2019, pp. 440–444.
[19] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Skeleton-based action recognition with directed graph neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7904–7913.
[20] G. Zhu, L. Zhang, P. Shen, and J. Song, "An online continuous human action recognition algorithm based on the Kinect sensor," *Sensors*, vol. 16, no. 2, p. 161, Jan. 2016.
[21] B. Sheng, F. Xiao, L. Gui, and Z. Guo, "Context-aware faster RCNN for CSI-based human action perception," *IEEE Trans. Human-Mach. Syst.*, vol. 53, no. 2, pp. 438–448, Apr. 2023.
[22] X. Cheng and B. Huang, "CSI-based human continuous activity recognition using GMM-HMM," *IEEE Sensors J.*, vol. 22, no. 19, pp. 18709–18717, Oct. 2022.
[23] Y. K. Han and Y. B. Choi, "Human action recognition based on LSTM model using smartphone sensor," in *Proc. 11th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Jul. 2019, pp. 748–750.
[24] N. Dawar and N. Kehtarnavaz, "Action detection and recognition in continuous action streams by deep learning-based sensing fusion," *IEEE Sensors J.*, vol. 18, no. 23, pp. 9660–9668, Dec. 2018.
[25] Á. Fehér, "Denoising ECG signals by applying discrete wavelet transform," in *Proc. Int. Conf. Optim. Electr. Electron. Equip. (OPTIM) Int. Aegean Conf. Electr. Mach. Power Electron. (ACEMP)*, May 2017, pp. 863–868.
[26] M. S. Nazemi, H. Hakimnejad, and Z. Azimifar, "PCG denoising using AR-based Kalman filter," in *Proc. 29th Iranian Conf. Electr. Eng. (ICEE)*, May 2021, pp. 902–906.
[27] W. Park, S. Park, D. Lee, T. Yang, and S.-H. Kim, "Inter-twin connectivity for digital twin networks in secure contactless delivery service scenarios," in *Proc. IEEE 97th Veh. Technol. Conf. (VTC-Spring)*, Jun. 2023, pp. 1–5.
[28] D. Lee, S. Park, W. Park, T. Yang, and S.-H. Kim, "Fine-grained passenger-vehicle coupling management for secure ride-sharing services," in *Proc. IEEE 97th Veh. Technol. Conf. (VTC-Spring)*, Jun. 2023, pp. 1–5.
[29] Y. Wei, J. Jang-Jaccard, F. Sabrina, A. Singh, W. Xu, and S. Camtepe, "AE-MLP: A hybrid deep learning approach for DDoS detection and classification," *IEEE Access*, vol. 9, pp. 146810–146821, 2021.
[30] S.-H. Wang, T.-I. Chou, S.-W. Chiu, and K.-T. Tang, "Using a hybrid deep neural network for gas classification," *IEEE Sensors J.*, vol. 21, no. 5, pp. 6401–6407, Mar. 2021.
[31] E. Sangenis, C.-S. Jao, and A. M. Shkel, "SVM-based motion classification using foot-mounted IMU for ZUPT-aided INS," in *Proc. IEEE Sensors*, Oct. 2022, pp. 1–4.
[32] G. L. Goh, G. D. Goh, J. W. Pan, P. S. P. Teng, and P. W. Kong, "Automated service height fault detection using computer vision and machine learning for badminton matches," *Sensors*, vol. 23, no. 24, p. 9759, Dec. 2023.
[33] M. You, S. Park, S.-H. Lee, and T. Yang, "Proxy individual positioning via IEEE 802.11 monitor mode and fine-tuned analytics," in *Proc. IEEE 90th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2019, pp. 1–5.
[34] D. Liciotti, M. Bernardini, L. Romeo, and E. Frontoni, "A sequential deep learning application for recognising human activities in smart homes," *Neurocomputing*, vol. 396, pp. 501–513, Jul. 2020.
[35] C. A. Ronao and S.-B. Cho, "Human activity recognition with smartphone sensors using deep learning neural networks," *Expert Syst. Appl.*, vol. 59, pp. 235–244, Oct. 2016.
[36] D. N. Tran and D. D. Phan, "Human activities recognition in Android smartphone using support vector machine," in *Proc. 7th Int. Conf. Intell. Syst., Modelling Simulation (ISMS)*, Bangkok, Thailand, Jan. 2016, pp. 64–68.

WOOJIN PARK received the B.S. degree from the School of Computer Engineering, Chungbuk National University, South Korea, in 2023, where he is currently pursuing the master's degree with the Department of Computer Engineering.

HYEYOUNG AN received the B.S. degree from the College of Computer Science, Chungnam National University, South Korea, in 2001, and the M.S. degree from the Department of Electrical and Computer Engineering, Chungbuk National University, in 2023, where she is currently pursuing the Ph.D. degree with the Department of Computer Engineering. Her research interests include networking technologies, sensing and data analytics, optimization, AI, and the IoT.

SOOCHANG PARK (Member, IEEE) received the Ph.D. degree from Chungnam National University, South Korea, in 2011. He has been an Associate Professor with the Department of Computer Engineering, Chungbuk National University (CBNU), South Korea, since 2021, where he joined CBNU, in 2017. He was with The Hong Kong University of Science and Technology (HKUST), as a Research Associate, in 2016. He was with the Institut Mines-Telecom, Telcom SudParis, France, as a Research Engineer, from 2013 to 2015; and with Rutgers University, USA, as a Postdoctoral Researcher, in 2012. His research interests include networking technologies, sensing and data analytics, mobile and Internet of Things based smart applications.