Chapter 14
A Mobile Solution Based on Soft Computing for Fall Detection

Serkan Ballı, Ensar Arif Sağbaş, and Musa Peker

Abstract Falling is an important health risk, especially for the elderly people. This situation prevents individuals from living independently. Automatic and high-accuracy detection of the falls will contribute in preventing the negative situations that may occur. In this study, a mobile solution with a new architecture for the detection of falls is presented. For this purpose, motion sensor data have been collected simultaneously from smartwatch and smartphone with Android operating system. Data sets for both smartwatch and smartphone have been created by labeling the falls and actions which are not falling in the data. The performances of Decision Tree, Naive Bayes, and k-Nearest Neighbor (kNN) methods have been tested on these data sets, and the kNN method has given the best result on two data sets. Accordingly, the kNN method is used for classification in the developed Android-based mobile solution. In addition, it is aimed to detect and prevent actions that could lead to bad results by monitoring the heart rate of the user with the built-in heart rate monitor on the smartwatch.

14.1 Introduction

Falling is the cause of hospitalization in elders, which can result in injury [1]. A low-cost, high-efficiency mechanism for detecting falls is also important for many health and safety applications, including elderly care [2]. There are numerous deadly injuries due to falls. It is important to establish an automatic fall detection system, including the home environment, as reduction of the rescue period after the fall

S. Ballı · M. Peker
Faculty of Technology, Information Systems Engineering, Muğla Sıtkı Koçman University, Muğla, Turkey
e-mail: serkan@mu.edu.tr; musa@mu.edu.tr

E. A. Sağbaş (✉)
Faculty of Engineering, Department of Computer Engineering, Ege University,İzmir, Turkey
e-mail: ensar.arif.sagbas@ege.edu.tr

© Springer International Publishing AG, part of Springer Nature 2019

S. Paiva (ed.), *Mobile Solutions and Their Usefulness in Everyday Life*,
EAI/Springer Innovations in Communication and Computing,
https://doi.org/10.1007/978-3-319-93491-4_14

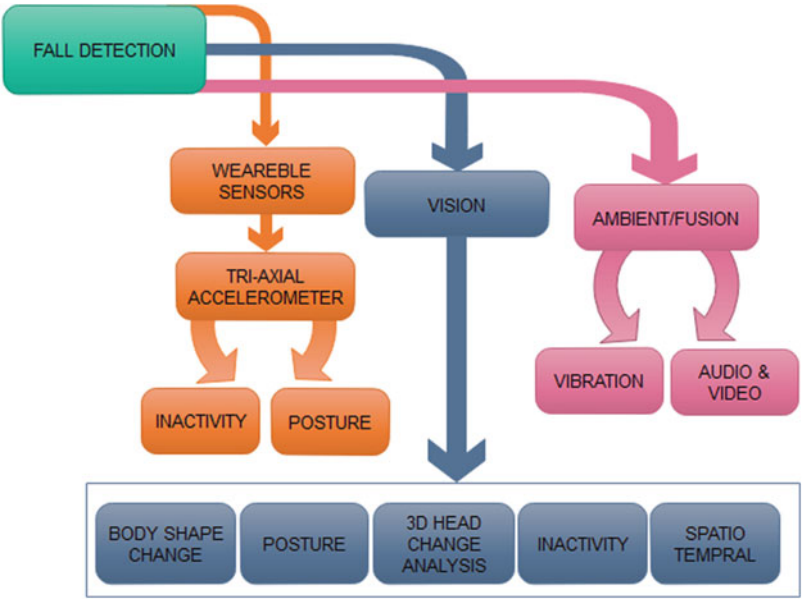

Fig. 14.1 Classification of fall detection methods [5]

detection may increase the overall survival rate of the aged person to an extreme extent [3]. Automatic fall detection helps people stay at home safely by reducing the negative consequences of falls in elders by encouraging independent living [4]. The methods used for automatic fall detection are classified in Fig. 14.1.

Thanks to wearable technology, many products such as watches, shoes, glasses, and clothes we use in everyday life have more features than normal [6]. With these devices, various information about the user can be obtained and evaluated. When falling detection is considered, most systems that have been described till today include threshold-based algorithms; machine learning-based fall detections carried out in recent years provide increased accuracy [1]. There are studies in the literature that provide mobile solutions for falling by using machine learning methods and wearable detectors.

Gibson et al. [7] have presented and evaluated an accelerometer-based multiclassifier fall detection and diagnostic system for remote health control. Kwolek and Kepski [8] have presented a new approach for reliable fall detection. In their study, the moment of the collision has been determined using the acceleration data; thus, it has been possible to calculate the time exposure transitions. Srinivasan et al. [9] have developed a wireless sensor network system for automatic fall detection. To detect falls, they used a combination of motion detectors placed in the field of view and a three-axis accelerometer attached to the body. Amin and Zhang [10] have described the signal processing algorithms and techniques involved in detecting the fall of the elders with radar signals. Radar signals have a nonstationary structure

and play a fundamental role in defining the motion, including determining and classifying falling features. Wang et al. [3] have investigated correlations between different types of radio signals and their activities by analyzing the radio propagation model. In this context, they have proposed a fall detection system that is named WiFall. Skubic et al. [11] have described two studies in which fall detection sensor technology has been tested. Bourke et al. [4] have extracted 12 different features from the data set which includes 89 fallings and 368 days of action. Machine learning method has been applied to the extracted features, and a number of algorithms based on different feature combinations have been created. Chen et al. [2] have proposed an accurate, over-sourced, adaptive fall detection approach by using smart devices with integrated wireless connectivity and sensors. Hsieh et al. [12] have proposed a machine learning-based fall detection algorithm using multiple Support Vector Machine (SVM) and k-Nearest Neighbor (kNN) classifiers with linear, quadratic, or polynomial kernel functions. Aziz et al. [1] have compared the accuracy of machine learning-based and threshold-based fall detection approaches in the fall data set obtained from ten young participants. Cola et al. [13] have investigated the use of a barometer (e.g., embedded in a pair of eyewear) placed in the wearer's head as a means to develop existing wearable sensor-based fall detection methods.

In this study, an Android-based mobile solution has been developed in order to minimize the injuries via notifying related persons or organizations by detecting falls for elderly people. For this purpose, motion sensor data have been obtained from smartwatches and smartphones with Android operating system, and these data have been evaluated by machine learning method. With this mobile solution that has a new architecture of using smartwatch and smartphone together, it is aimed to minimize the problems that may arise as a result of falls. After the detection of the fall, information about the patient or elderly (position, position changes, and actions) is transferred to the necessary units (caregiver, hospital, etc.) in mobile environment.

Smart device sensors will be explained later in Sect 14.2. The method used in experimental study and the findings will be discussed in Sect 14.3. After that, the architecture of the developed Android-based mobile solution will be elaborated in Sect 14.3.1. Finally, the study will end with the obtained results.

14.2 Sensors

In the scope of the study, gyroscope and accelerometer sensors have been used to detect the fall. When fall detection is performed, the global positioning system (GPS) sensor is used to determine the location of the fall. In addition, thanks to the heart rate monitor, it is aimed to prevent the occurrence of a negative situation for the person by detecting and monitoring the heart rate of the person during the day.

14.2.1 Accelerometer

The built-in accelerometer sensor of smartphones and smartwatches measures the acceleration affecting the smartphone and smartwatch in the direction of the axes shown in Fig. 14.2. The raw sensor information is obtained from the accelerometer in three axes in m/s^2. The content of the raw accelerometer sensor data is given in Eq. 14.1:

$$\text{Acc}_i =< x_i, y_i, z_i >,\ i = (1, 2, 3, \ldots) \tag{14.1}$$

Time information is also obtained in addition to the acceleration values. Most existing accelerometers allow you to set (on the user interface) how many sample data will be collected in seconds. Thanks to this, users can select the most appropriate sample rate for his/her study [14].

Accelerometer is often used in smart device-based action recognition applications. This sensor's popularity comes from the fact that the sensor can directly calculate the physical movement of the device or user. For example, if the user moves from walking to jumping state, accelerometer signals will change on the vertical axis [14, 15]. According to Fig. 14.2, the X-axis gives information for the side face of the device, the Y-axis for a vertical position, and the Z-axis for the flat (supine) position [16]. For example, if the Z value is 0 or very close, it means that the device is standing on one of its edges. When operating with an accelerometer, it should be kept in mind that, the accelerometer calculates the linear acceleration of the device, the numerical value obtained is the gravitational force affecting the device, and if the device is in motion, it is the acceleration of the device and gravitational force [17].

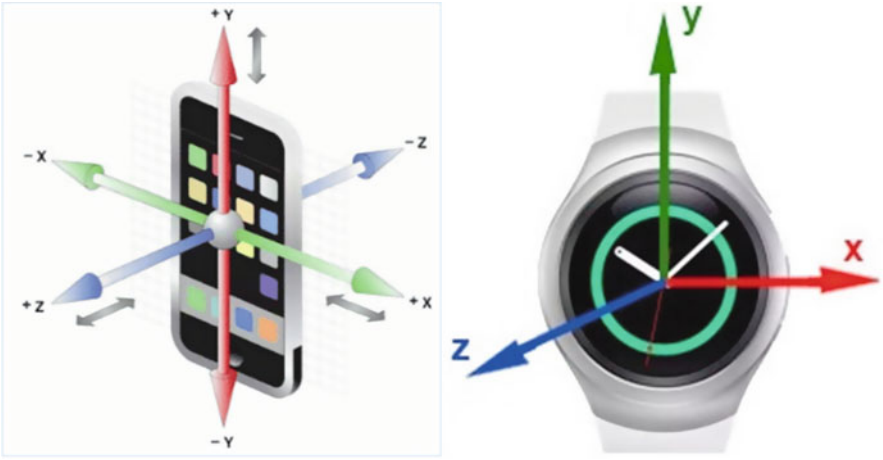

Fig. 14.2 Accelerometer sensor axes of smartphone and smartwatch

14.2.2 Gyroscope

A gyroscope is a tool which is used to detect or measure direction. It is used to find direction in aircrafts and ships in everyday life [17, 18].

The rotation of the Earth, shown in Fig. 14.3, also carries the characteristic feature of a gyroscope. The Earth's rotation on its axis creates a balancing effect and allows it to rotate while showing the pole star. Rapidly rotating propeller, sphere, ball, etc. are basically a gyroscope. As shown in Fig. 14.4, the gyroscope consists of a rotor (disk) with free rotation and interconnected joints (gimbal joints) [19].

The gyroscope sensor on the smart devices gives the angular velocity that the smart device has made on the x, y, and z axes. The gyroscope axis trajectories for smartphones are shown in Fig. 14.5. The raw data obtained from the gyroscope sensor report the rotation of the smart device around the three physical axes in rad/s. The contents of raw gyroscope sensor data are given in Eq. 14.2:

$$\text{Rotation}_i =< x_i, y_i, z_i >,\ i = (1, 2, 3, \ldots) \tag{14.2}$$

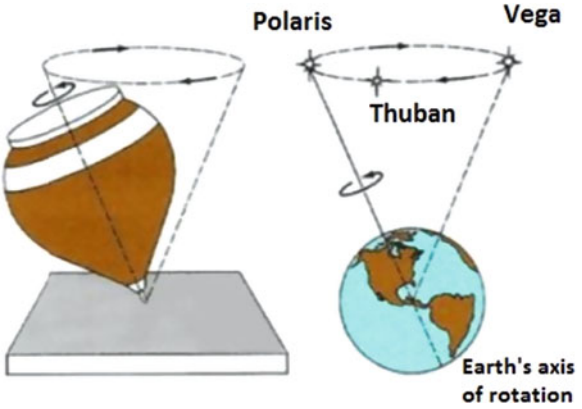

Fig. 14.3 The world's rotation movement [19]

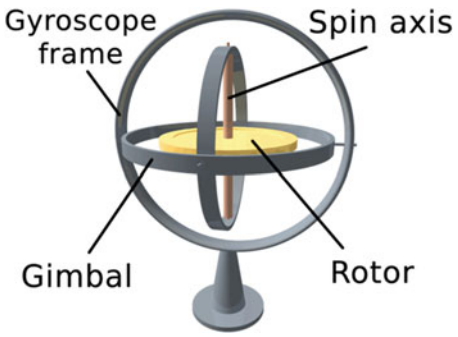

Fig. 14.4 Structure of gyroscope [20]

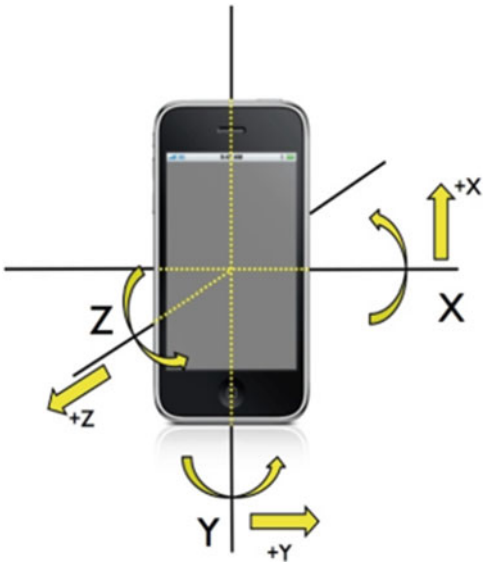

Fig. 14.5 Sensor axes of smartphone gyroscope

The gyroscope sensor is used in smartphone games with character orientation. This sensor is used to perform direction determination in action recognition studies [14, 15].

14.2.3 Global Positioning System (GPS)

GPS, which is shown in Fig. 14.6, is a satellite-based guidance system developed by the US Department of Defense in the beginning of the 1970s. This system has been developed primarily for military purposes, but personal usage has become possible with time. GPS is a passive system that can provide location and time information to unlimited persons anywhere in the world under any weather conditions. In other words, users have position information by processing the signals coming from the satellite [15, 21].

In smart devices, location can also be determined with the help of wireless networks and associated base stations. However, if the device does not have a GPS sensor, position detection can be performed with limited accuracy. The GPS signal contains the following parameters [17, 22]:

- *Latitude:* Obtained in degree unit. Latitude positive values denote the north of the equator, and negative values denote the south of the equator.
- *Longitude:* Measurements are according to zero meridian. It is obtained in degree units. Positive values indicate eastern meridians, and negative values indicate western meridians.

Fig. 14.6 GPS satellites in orbit around the world

- *Velocity:* The device's instantaneous velocity is calculated in meters/second. It cannot be calculated if it does not receive a GPS signal. The velocities with a negative value are invalid.
- *Altitude:* Altitude is calculated in meters. Positive values indicate the altitude of the device from the sea level.

In addition, other features of GPS are as follows [17, 23, 24]:

- GPS devices are passive receivers. They do not give feedback to satellites.
- GPS satellites are synchronized using atomic clocks.
- Satellites periodically transmit signals containing current location and time information. The distance between the receiver and the satellite can be calculated according to the time of arrival of these signals to the receiver. Figure 14.7 shows an example of positioning in two dimensions. If the satellite positions (S_1, S_2, S_3) and distances (ρ_1, ρ_2, ρ_3) between the user and the satellites are known, the user position indicated by U can be found. If two distance information is used, there will be two candidates for the user's position because the circles will have two intersection points. With the third distance information, the user position can be determined precisely [25].
- GPS does not work in an indoor environment.

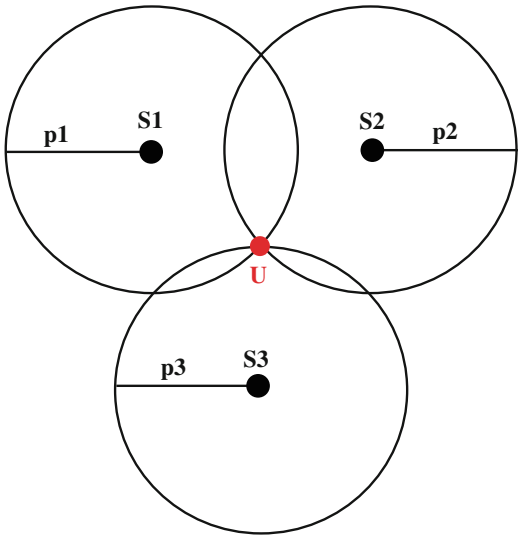

Fig. 14.7 Positioning in two dimensions [25]

- GPS quickly consumes the battery life of the device.
- A position fix takes a very long time (30 s–12 min).
- The buildings reflect or block GPS signals. Due to this reason, the accuracy rate decreases in settlements.

14.2.4 Heart Rate Monitor

The heart rate sensor transmits the information of how many times the user's heart beats in a minute. The accuracy rate reported by the sensor provides information about the situation in which the pulse has been read [26]. Many wearable devices with a heart rate sensor use a method called Photoplethysmography (PPG) to calculate heart rate. PPG is a technical term that calculates the amount of light scattered by blood flow by illuminating the skin. Thus, the change in heart rate can be calculated. The structure of the heart rate sensor is shown in Fig. 14.8 [27]. PPG uses four technical components to calculate heart rate.

- *Optical transmitter:* Generally, it consists of at least two LED light sources for transmitting light waves into the skin. This is because of differences in skin tones.
- *Digital signal processor:* The digital signal processor captures the light waves broken from the user's skin and calculates significant heart rate data between 1 and 0.
- *Accelerometer:* An accelerometer measures the motion, and it is used with digital signal processor's signals as input to motion-tolerant PPG algorithms.

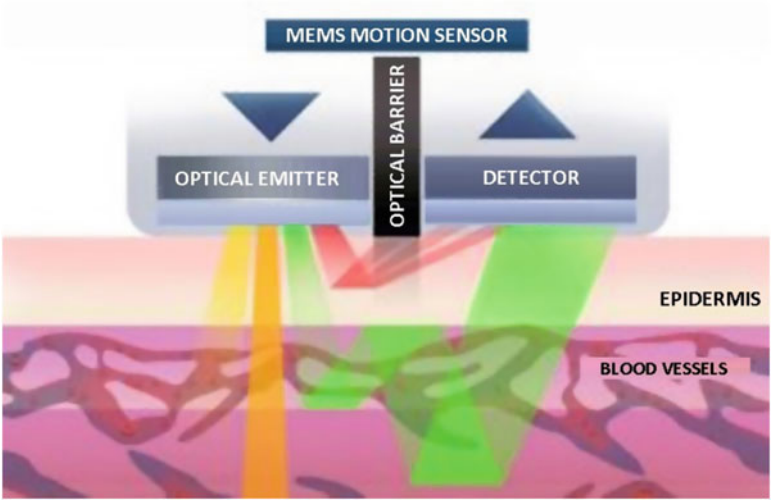

Fig. 14.8 Optical heart rate sensor structure [27]

- *Algorithms:* Information such as calories burned, heart rate variability, and heart oxygen level can be calculated by the data from the digital signal processor and accelerometer.

14.3 Methods

14.3.1 k-Nearest Neighbor (kNN)

The k-nearest neighbor method, which is a classification method, is an algorithm that classifies based on distance. This method, which does not have a connection between its features, is one of the controlled machine learning algorithms that is simple to interpret and implement and easy to have results. kNN uses the nearest neighbor samples to classify or estimate patterns in n-dimensional feature space. In order to be able to classify in the kNN algorithm, the number of nearest neighbors to be considered is expressed with a positive integer k. If k is 1, the pattern to be classified will be included in the class where the nearest neighbor is located. This method is also used for estimation.

In the determination of the nearest neighbors, the distance between the selected sample and the samples in the training set is measured. The distances between the samples are sorted from least to the maximum; this sequence also shows the order that is from the closest neighbor (from the selected sample) to the farthest neighbor [28]. For distance calculation, Manhattan, Euclid, and Minkowski distance measures

are used. The formulas for these criteria are presented in Eqs. 14.3, 14.4, and 14.5, respectively:

$$d\,(i,\,j) = \left|X_{i1} - X_{j1}\right| + \left|X_{i2} - X_{j2}\right| + \cdots + \left|X_{ip} - X_{jp}\right| \tag{14.3}$$

$$d\,(i,\,j) = \sqrt{\left|X_{i1} - X_{j1}\right|^2 + \left|X_{i2} - X_{j2}\right|^2 + \cdots + \left|X_{ip} - X_{jp}\right|^2} \tag{14.4}$$

$$d\,(i,\,j) = \sqrt[p]{\left|X_{i1} - X_{j1}\right|^p + \left|X_{i2} - X_{j2}\right|^p + \cdots + \left|X_{ip} - X_{jp}\right|^p} \tag{14.5}$$

14.3.2 Decision Tree (C4.5)

The C4.5 algorithm provides obtaining the classification trees of features with categorical and numerical values. During the creation of the classification trees, starting (from which feature) of branching is important. Uncovering all possible tree structures by taking advantage of a training data set and selecting the most suitable ones among these tree structures causes the repetition of many operations. For this reason, the classification tree algorithms calculate the various values at the beginning of the process and proceed to create the tree according to these values. Entropy can be used for this purpose. Branching of the tree will take place according to the entropy value.

Assume that the class attribute is divided to k class as $\{C_1, C_2, \ldots, C_k\}$ according to the values class attribute will take. Class attribute is the probability distribution of P_T classes for (T) and calculated as shown in Eq. 14.6:

$$P_T = \left(\frac{|C_1|}{|T|}, \frac{|C_2|}{|T|}, \ldots, \frac{|C_k|}{|T|}\right) \tag{14.6}$$

$|C_i|$ gives the number of elements in the C_i set. For example, $p_1 = |C_1|/|T|$ probability. Thus, $P_T = (p_1, p_2, \ldots, p_k)$. The average amount of information for T is expressed in entropy using Eq. 14.7 [29]:

$$H(T) = H\,(P_T) = -\sum_{i=1}^{k} p_i \log_2\,(p_i) \tag{14.7}$$

14.3.3 Naïve Bayes

Naive Bayesian Classifier is a simple algorithm based on probability, with strong attribute independence assumption. The Naive Bayesian Classifier performs learn-

ing through the test data and includes the highest sample into the class. Assume that C denotes a class. $x\langle x_1, x_2, x_3, \ldots, x_m\rangle$ values are the values of the observed features. c denotes a known class label, and $x\langle x_1, x_2, x_3, \ldots, x_m\rangle$ denotes the values of known and observed features. The Bayes Theorem calculates the probability to estimate the class according to x test data:

$$p\left(C = c_j \mid X = x\right) = \frac{p\left(C = c_j\right) p\left(X = x | C = c_j\right)}{p\left(X = x\right)} \tag{14.8}$$

After that, it estimates the class with the highest probability. In this example, $X = x\ X_1 = x_1\ \hat{}\ X_2 = x_2\ \hat{}\ X_3 = x_3\ \hat{}\ \ldots\ \hat{}\ X_m = x_m$. $p(X = x)$ is ignored in cases where it does not show any change between classes, and Eq. 14.8 is as follows:

$$p\left(C = c_j \mid X = x\right) = p\left(C = c_j\right) p\left(X = x \mid C = c_j\right) \tag{14.9}$$

where $(C = c_j)$ and $p(X = x \mid C = c_j)$ are predicted from learning data. X_1, X_2, X_3, $\ldots$, X_m features are conditionally independent from each other. In this case, Eq. 14.9 is as follows:

$$p\left(C = c_j \mid X = x\right) = p\left(C = c_j\right) \prod_{i=1}^{m} p\left(X_i = x_i \mid C = c_j\right) \tag{14.10}$$

Using the Naive Bayes equation given in Eq. 14.10, it is much easier to calculate test samples and estimate from the learning data. The Naive Bayes Classifier can handle both categorical and numeric features. For each discrete feature, $p(C = c_j \mid X = x)$, which in Eq. 14.10 is modeled with real numbers between 0 and 1. Estimation probabilities are obtained by the frequency of samples in the training data. In this approach, if x_i is not among the training data then zero will be obtained as a result of $p(X_i = x_i \mid C = c_j)$ [30–32].

14.4 Experimental Results and Applications

14.4.1 Data Set and Feature Extraction

The data set has been created with motion sensor (accelerometer and gyroscope) data simultaneously collected from smartwatch and smartphone. To this end, Android and Android Wear-based mobile applications that work in sync with each other have been developed. Among the obtained motion sensor data, the parts of the fall action have been separated manually and the patterns of the FALL class have obtained. As a result of examining the data, it has been deemed appropriate to use a window interval of 0.7 s for FALL patterns. As in the previous studies [6, 15] carried out with smartwatches and smartphones, smart devices were set up to

collect 50 sensor data in a second and patterns were generated with $0.7 \times 50 = 35$ sample data. Data samples of falls for smartwatches and smartphones are shown in Fig. 14.9 (accelerometer) and Fig. 14.10 (gyroscope).

104 FALL patterns have been obtained for smartwatch, and 115 FALL patterns have been obtained for smartphone. Classes other than the fall class are completed with the data (walking, descending a ladder, ascending a ladder, using stationary, and using an elevator) used in the study [15] for smartphone, and data (brushing teeth, writing, writing board, using keyboard, stationary, vacuuming, and walking activities) used in the study [6] for smartwatch. However, these classes have been evaluated as NOT_FALL. In other words, incorrect classifications between NOT_FALL classes have not affected FALL classification. Features of the patterns have been created by calculating the max, min, standard deviation, and mean values of the data obtained from the triaxial accelerometer and gyroscope at 0.7 s. The extracted features are presented in Table 14.1.

14.4.2 Classification

Classification of data sets created for smartwatch and smartphone has been carried out with machine learning methods. At this stage, the kNN, Naive Bayes, and C4.5 algorithms have been used and their performances have been compared. In the developed fall detection mobile application, the most successful method has been

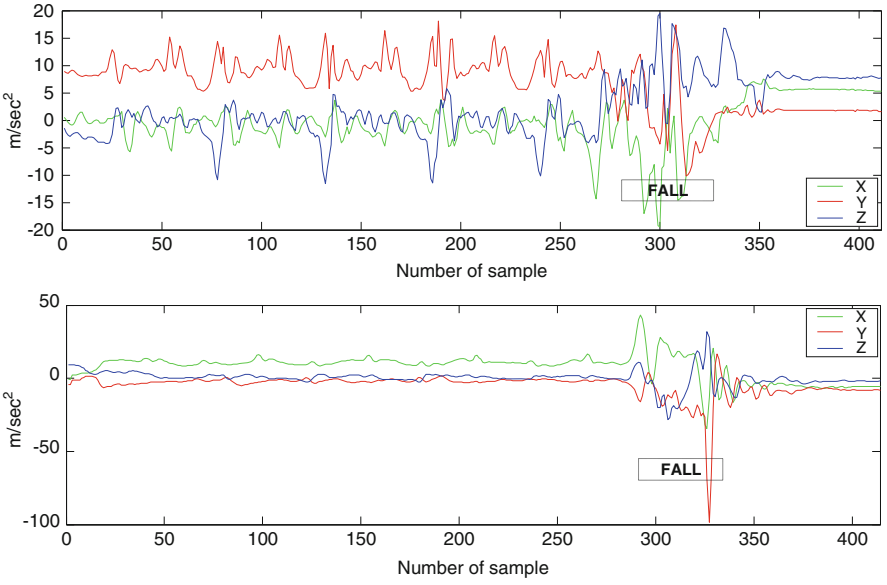

Fig. 14.9 The fall data obtained from accelerometer sensor

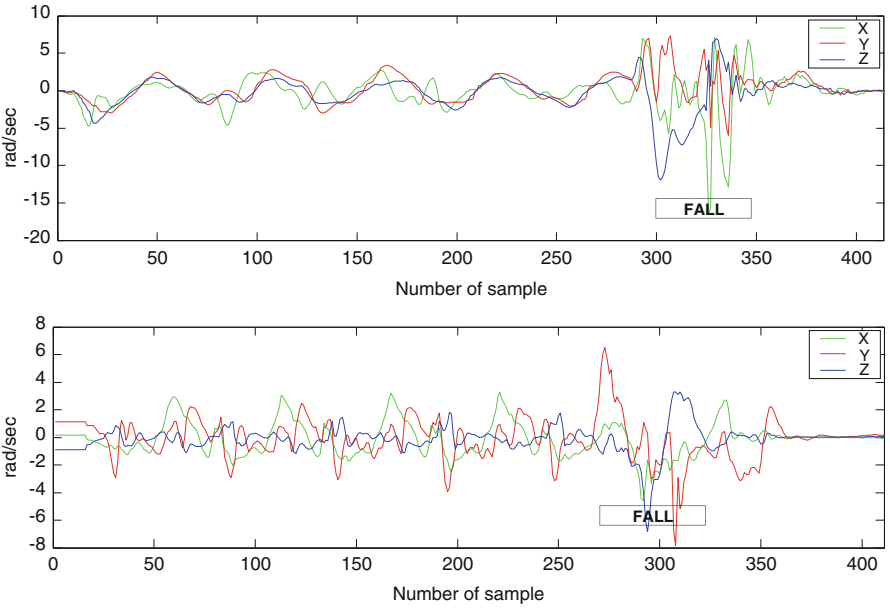

Fig. 14.10 The fall data obtained from the gyroscope sensor

Table 14.1 Feature list

Feature number	Sensor	Description of features
1, 2, 3	Accelerometer	Standard deviation of accelerometer (x, y, and z axes) sensor data
4, 5, 6	Accelerometer	Mean of accelerometer (x, y, and z axes) sensor data
7, 8, 9, 10, 11, 12	Accelerometer	Maximum and minimum values of accelerometer (x, y, and z axes) sensor data
13, 14, 15	Gyroscope	Standard deviation of gyroscope (x, y, and z axes) sensor data
16, 17, 18	Gyroscope	Mean of gyroscope (x, y, and z axes) sensor data
19, 20, 21, 22, 23, 24	Gyroscope	Maximum and minimum values of gyroscope (x, y, and z axes) sensor data

chosen. Numerical results of the experiments (classification accuracy, Root Mean Square Error (RMSE), area under Curve (AUC), and F-measure) are given in Table 14.2. Tests have been conducted using the ten-fold cross validation method which is a reliable and frequently preferred data selection method.

When Table 14.2 is examined, it is seen that classification has been carried out with an accuracy of over 97% in all of the tests performed with the smartwatch. In accordance with the accuracy rates, the lowest quadratic error is 0.0523, AUC 0.997, and 0.986 is the highest F-measure obtained with the kNN method. In the tests performed with the smartwatch, the most successful result has been obtained from the kNN method as in the smartphone. With this method, 0.1172 RMSE, 0.989

Table 14.2 The result obtained with smartwatch and smartphone

Method	CA	RMSE	AUC	*F*-Measure
Smartwatch				
Naive Bayes	97.6	0.0752	0.997	0.976
kNN ($k = 5$)	98.55	0.0523	0.997	0.986
C4.5	98.44	0.0594	0.996	0.984
Smartphone				
Naive Bayes	91.74	0.164	0.982	0.915
kNN ($k = 5$)	94.78	0.1172	0.989	0.948
C4.5	91.3	0.1628	0.951	0.913

Table 14.3 Confusion matrices obtained from the kNN method

Smartphone		
Classified as	**FALL (%)**	**NOT_FALL (%)**
FALL	93.9	6.1
NOT_FALL	0	100
Smartwatch		
Classified as	**FALL**	**NOT_FALL**
FALL	97.1	2.9
NOT_FALL	0	100

AUC, and 0.948 F-Measure values have been obtained. In the developed mobile applications, the kNN method has been used in the direction of the experiments performed. The confusion matrix obtained with the kNN method is presented in Table 14.3.

When the confusion matrices are examined, it is seen that NOT_FALL actions are classified with 100% accuracy in both devices. FALL action is classified with approximately 94% by smartphone and approximately 97% with smartwatch.

14.4.3 Mobile Solution for Fall Detection

Within the scope of the study, falls have been determined by using the data together obtained from smartwatch and smartphone. The data obtained from both devices are classified within itself, and if both devices decide FALL, then the system decides that the person falls. ListenerService, whose codes have been shared in Fig. 14.11, has been used in the communication between the smartphone and smartwatch. The fall detection algorithm is shown in Fig. 14.13.

In the developed mobile solution, the smartwatch only gathers the motion sensor data and transmits to the smartphone. In the same time frame, the smartphone also acquires sensor data. The extraction of the features from the raw data and the classification of them by the kNN method are performed by smartphone. In order for the system to operate at the same time, the user sends a "BEGIN" signal to the phone by pressing the "BEGIN" button (Fig. 14.14a) in the smartwatch interface and this button becomes "END" button (Fig. 14.14b). The smartphone receiving the

```
public class ListenerService extends WearableListenerService{
    @Override
    public void onMessageReceived(MessageEvent messageEvent) {
        if (messageEvent.getPath().equals("/message_path")) {
            String message = new String(messageEvent.getData());
            Intent messageIntent = new Intent();
            messageIntent.setAction(Intent.ACTION_SEND);
            messageIntent.putExtra("message", message);
            LocalBroadcastManager.getInstance(this).sendBroadcast(messageIntent);
            Log.v("myTag", "Message path received on watch is: " + messageEvent.getPath());
            Log.v("myTag", "Message received on watch is: " + message);
        }else {
            super.onMessageReceived(messageEvent);
        }//else
    }//onMessageReceived
}//WearableListenerService
```

Fig. 14.11 ListenerService codes

```
SensorManager sensorManager;
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensorManager.registerListener((SensorEventListener) MainActivity.this,
sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE), 20000);
sensorManager.registerListener((SensorEventListener) MainActivity.this,
sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER), 20000);
```

Fig. 14.12 Codes for activating sensors

"BEGIN" signal changes its state from "WAITING" (Fig. 14.20a) to "ACTIVE" (Fig. 14.20b) and starts collecting the sensor data. The sample code block needed to collect the sensor data is as shown in Fig. 14.12.

With this process, the watch and the phone start collecting sensor data at the same time, and patterns are generated every 0.7 sec. These patterns are classified by the kNN method and are added to circular queue structures separately for the smartphone and smartwatch (Fig. 14.15). These queue structures are set to consist of three elements. This method has been applied with the purpose of preventing the possible error caused by the time elapsed between the communication of smartwatch and smartphone.

If FALL action is found in both of the queues when both of the circular queues reach at three elements, the system decides that the FALL action is performed and the smartphone sends a "FALL" signal to the smartwatch. The system that decides the FALL action gives the user 3 s to cancel this decision. The "END" button on the smartwatch turns into the "CANCEL" button (Fig. 14.14c), and the screen background turns into red as an alert and the smartwatch vibrates. After 3 s, if the user does not cancel this decision, the final fall decision is made and the "CANCEL" button turns into the "FALL" button (Fig. 14.14d). Then, the "FALL" signal is sent to the smartphone. A 3-sec waiting period for canceling the detected fall is provided by the codes presented in Fig. 14.16.

After the smartphone receives the "FALL" signal, the system stops (Fig. 14.20c) and the location of the fall is sent to the predefined phone number via SMS (Fig. 14.20d). The use of the SMS library is as shown in Fig. 14.17:

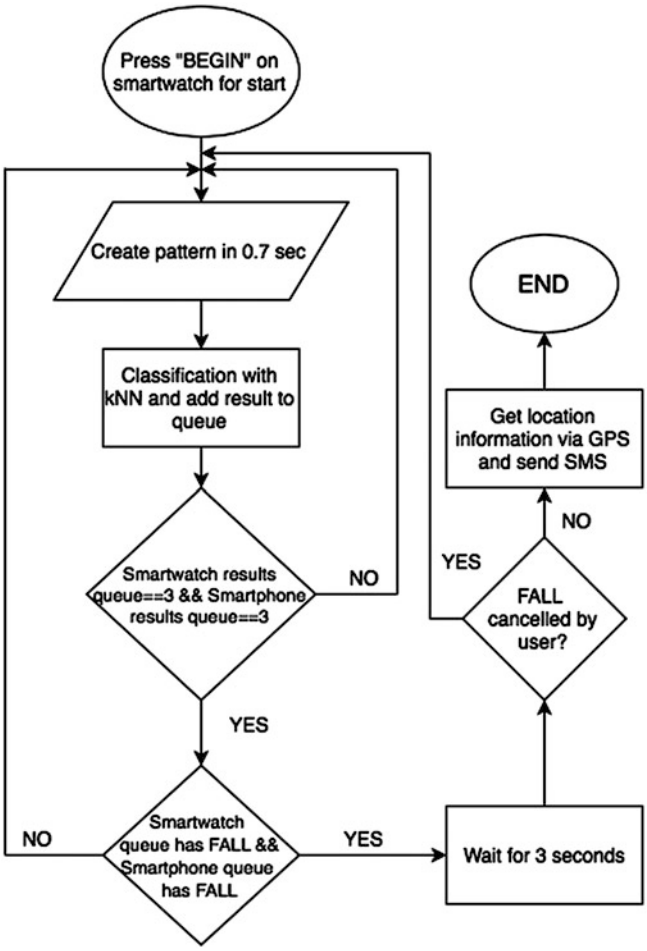

Fig. 14.13 Fall detection algorithm

Fig. 14.14 Smartwatch mobile application screenshots

For location detection, sending SMS, and using the heart rate sensor on devices with Android operating system, the lines in Fig. 14.18 should be added to the AndroidManifest.xml file.

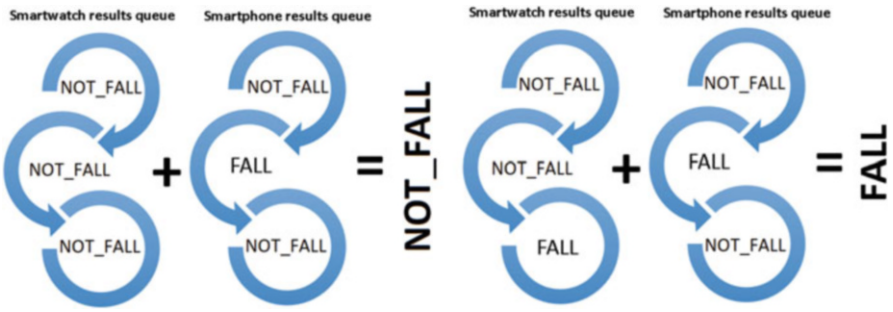

Fig. 14.15 Circular queue structure

```
handler.postDelayed(new Runnable() {
    @Override
    public void run() {
        if (status == 2) { //status=2 FALL
            new SendToDataLayerThread("/message_path", "FALL").start();
            button.setText("FALL");
        }
    }
},3000);
```

Fig. 14.16 Codes for cancellable waiting process

```
String phoneNo = "123456789";
SmsManager sms = SmsManager.getDefault();
String messageText = "Message text here!";
sms.sendTextMessage(phoneNo, null, messageText, null, null);
```

Fig. 14.17 Use of SMS library

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.SEND_SMS"/>
<uses-permission android:name="android.permission.BODY_SENSORS"/>
```

Fig. 14.18 Permission list for using sending SMS, accessing GPS, and heart rate sensor

```
LocationListener locationListener;
LocationManager locationManager;
locationListener = new myLocationListener();
locationManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0, locationListener);
```

Fig. 14.19 Codes for location detection with GPS

The code lines required for location detection with GPS are shown in Fig. 14.19.

At the time when the falls are detected, the smartwatch can also check the user's pulse. With a specified threshold point, it is possible to detect situations that may be dangerous. Thus, if an individual is an elderly one, it is possible for the

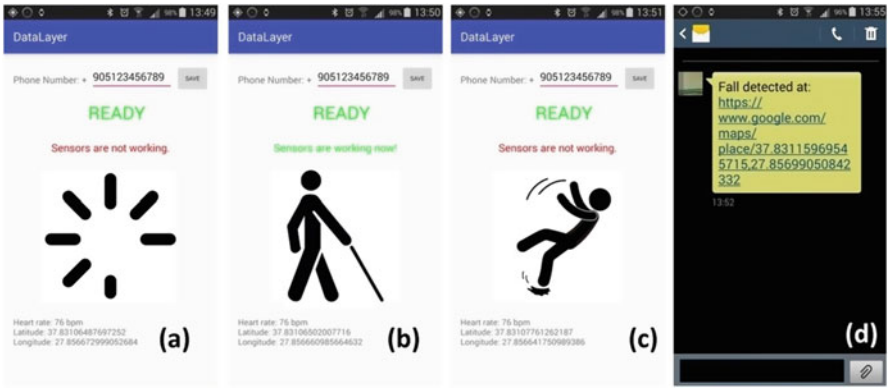

Fig. 14.20 Smartphone mobile application screenshots

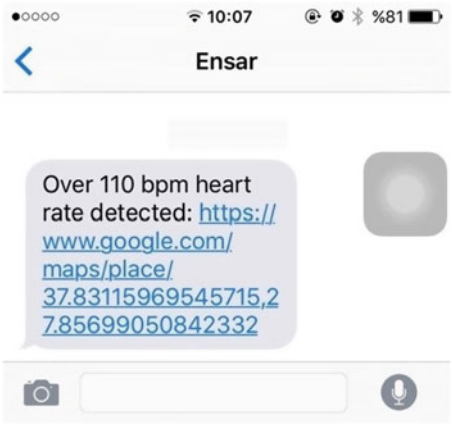

Fig. 14.21 Warning SMS at high heart rate

elderly person's relative to take precaution by being informed (Fig. 14.21) when the followed heart rate signals show an active movement in a harmful level (for instance, running).

In the study, heart rate is determined by the smartwatch, and when the upper limit is exceeded, information is sent by SMS.

14.5 Conclusion

In this study, a new architecture that uses smartwatch and smartphone together has been developed to perform the fall detection more accurately. The features have been extracted from the data obtained from both the smartwatch and smartphone motion sensors. It has been observed that very sharp transitions have occurred in the motion sensor when the falls are occurred. But these transitions can also occur when

the user shakes his/her hand hardly or when he/she hits an object on the ground with his/her foot. Thanks to the developed system, it is required to detect the fall for two devices at the same time, so that false fall detections are avoided. Classifications have been carried out by machine learning methods, and over 90% accuracy rate has been obtained. In addition to this, it is aimed to prevent the unwanted situations by following the heart rate information of the user.

Acknowledgments This study is supported by Muğla Sıtkı Koçman University Scientific Research Projects under the grant number 016-061.

References

1. Aziz O, Musngi M, Park EJ, Mori G, Robinovitch SN (2017) A comparison of accuracy of fall detection algorithms (threshold-based vs. machine learning) using waist-mounted tri-axial accelerometer signals from a comprehensive set of falls and non-fall trials. Med Biol Eng Comput 55(1):45–55
2. Chen HM, Chen CK, Wu HI, Tsay RS (2016) An accurate crowdsourcing-based adaptive fall detection approach using smart devices. In: Healthcare informatics (ICHI), 2016 IEEE international conference on, IEEE, Oct 2016, pp 456–460
3. Wang Y, Wu K, Ni LM (2017) Wifall: device-free fall detection by wireless networks. IEEE Trans Mob Comput 16(2):581–594
4. Bourke AK, Klenk J, Schwickert L, Aminian K, Ihlen EA, Mellone S, ... Becker C (2016). Fall detection algorithms for real-world falls harvested from lumbar sensors in the elderly population: a machine learning approach. In: Engineering in medicine and biology society (EMBC), 2016 IEEE 38th annual international conference of the, IEEE, Aug 2016, pp 3712–3715
5. Mubashir M, Shao L, Seed L (2013) A survey on fall detection: principles and approaches. Neurocomputing 100(144):152
6. Ballı S, Sağbaş EA (2017) The usage of statistical learning methods on wearable devices and a case study: activity recognition on smartwatches. Advances in statistical methodologies and their applications to real problems, InTech, Rijeka, Croatia, 259–277
7. Gibson RM, Amira A, Ramzan N, Casaseca-de-la-Higuera P, Pervez Z (2016) Multiple comparator classifier framework for accelerometer-based fall detection and diagnostic. Appl Soft Comput 39:94–103
8. Kwolek B, Kepski M (2016) Fuzzy inference-based fall detection using kinect and body-worn accelerometer. Appl Soft Comput 40:305–318
9. Srinivasan S, Han J, Lal D, Gacic A (2007). Towards automatic detection of falls using wireless sensors. In: Engineering in medicine and biology society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE. IEEE, Aug 2007, pp 1379–1382
10. Amin MG, Zhang YD, Ahmad F, Ho KD (2016) Radar signal processing for elderly fall detection: the future for in-home monitoring. IEEE Signal Process Mag 33(2):71–80
11. Skubic M, Harris BH, Stone E, Ho KC, Su BY, Rantz M (2016) Testing non-wearable fall detection methods in the homes of older adults. In: Engineering in medicine and biology society (EMBC), 2016 IEEE 38th annual international conference of the, IEEE, Aug 2016, pp 557–560
12. Hsieh CY, Huang CN, Liu KC, Chu WC, Chan CT (2016) A machine learning approach to fall detection algorithm using wearable sensor. In: Advanced materials for science and engineering (ICAMSE), international conference on, IEEE, Nov 2016, pp 707–710

13. Cola G, Avvenuti M, Piazza P, Vecchio A (2016) Fall detection using a head-worn barometer. In: International conference on wireless mobile communication and healthcare. Springer, Cham, pp 217–224
14. Su X, Tong H, Ji P (2014) Activity recognition with smartphone sensors. Tsinghua Sci Technol 19(3):235–249
15. Sağbaş EA, Ballı S (2016) Comparison of logistic regression and kNN methods in activity recognition with smartphone sensor data. 1st international conference on engineering technology and applied sciences, 21–22 Apr 2016, Afyonkarahisar, Turkey, pp 894–899
16. Batmaz B, Çelik Z, Bayılmış C, Kırbaş İ (2015) A person tracking system based on smartphone. Sakarya Univ J Sci 19(1):75–82
17. Sağbaş EA, Ballı S (2015) Usage of the smartphone sensors and accessing raw sensor data. Academic Computing Conference (Akademik Bilişim Konferansı). 4–6 Feb 2015, Eskişehir, Turkey, pp 158–164
18. Gyroscope https://tr.wikipedia.org/wiki/Jiroskop. Accessed 28 Oct 2016
19. Ayabakan T (2014) *Gyroscopes and gyrostabilizers*. Master's thesis, Yıldız Technical University, İstanbul, 98 pages
20. 3D Jiroskop: https://en.wikipedia.org/index.php?q=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnL3dpa2kvRmlsZTozRF9HeXJvc2NvcGUucG5n. Accessed 11 Oct 2017
21. El-Rabbany A (2002) Introduction to GPS: the global positioning system, 2nd edn. Artech House, Norwood
22. Sensor Data, http://wavefrontlabs.com/Wavefront_Labs/Sensor_Data.html. Accessed 28 Oct 2016
23. Sensors and Cellphones: http://web.stanford.edu/class/cs75n/Sensors.pdf. Accessed 28 Oct 2016
24. Smartphone sensors: https://www.uni-weimar.de/kunst-und-gestaltung/wiki/images/Zeitmaschinen-smartphonesensors.pdf. Accessed 28 Oct 2016
25. Çınar S (2005) *Vehicle tracking and guidance system using GPS,* Master's thesis, Hacettepe University, Ankara, 96 pages
26. Sensor: https://developer.android.com/reference/android/hardware/Sensor.html. Accessed 7 Oct 2016
27. Optical heart rate monitoring: What you need you know: http://valencell.com/blog/2015/10/optical-heart-rate-monitoring-what-you-need-to-know/. Accessed 7 Oct 2016
28. Köktürk F (2012) Comparing classification success of k-nearest neighbor, artificial neural network and decision trees, Doctoral thesis, Bülent Ecevit University, Zonguldak, 77 pages
29. Özkan Y, Erol Ç (2015) Biyoenformatik DNA Mikrodizi Veri Madenciliği. Papatya Bilim, İstanbul, 432
30. Chandra B, Gupta M, Gupt MP (2007) Robust approach for estimating probabilities in Naive-Bayes classifier. In: Pattern recognition and machine intelligence, 18–22 Dec 2007, Kolkata, India, pp 11–16
31. Sağbaş EA, Ballı S (2016) Transportation mode detection by using smartphone sensors and machine learning. Pamukkale Univ J Eng Sci 22(5):376–383
32. Ballı S, Sağbaş EA (2017) Diagnosis of transportation modes on mobile phone using logistic regression classification. IET Softw 12(2):142–151