Metrol. Meas. Syst., Vol. 28 (2021) No. 1, pp. 23–40
DOI: 10.24425/mms.2021.135999

PAN POLSKA AKADEMIA NAUK

METROLOGY AND MEASUREMENT SYSTEMS
Index 330930, ISSN 0860-8229
www.metrology.wat.edu.pl

M&MS

A TWO-STEP FALL DETECTION ALGORITHM COMBINING THRESHOLD-BASED METHOD AND CONVOLUTIONAL NEURAL NETWORK

Tao Xu, Haifeng Se, Jiahui Liu

Shenyang Aerospace University, School of Automation, Shenbei New District, Shenyang, China
(✉ *xutao@sau.edu.cn*, +86 155 0982 9348, *2474014137@qq.com*, *3060212475@qq.com*)

Abstract

Falls are one of the leading causes of disability and premature death among the elderly. Technical solutions designed to automatically detect a fall event may mitigate fall-related health consequences by immediate medical assistance. This paper presents a wearable device called TTXFD based on MPU6050 which can collect triaxial acceleration signals. We have also designed a two-step fall detection algorithm that fuses threshold-based method (TBM) and machine learning (ML). The TTXFD exploits the TBM stage with low computational complexity to pick out and transmit suspected fall data (triaxial acceleration data). The ML stage of the two-step algorithm is implemented on a server which encodes the data into an image and exploits a fall detection algorithm based on convolutional neural network to identify a fall on the basis of the image. The experimental results show that the proposed algorithm achieves high sensitivity (97.83%), specificity (96.64%) and accuracy (97.02%) on the open dataset. In conclusion, this paper proposes a reliable solution for fall detection, which combines the advantages of threshold-based method and machine learning technology to reduce power consumption and improve classification ability.

Keywords: wearable, fall detection, MPU6050, threshold-based method, convolutional neural network.

© 2021 Polish Academy of Sciences. All rights reserved

1. Introduction

A fall is defined as "unintentionally coming to ground, or some lower level not as a consequence of sustaining a violent blow, loss of consciousness, sudden onset of paralysis as in stroke or an epileptic seizure" [1]. The frequency of falls increases with age and the elderly have the highest risk of severe injury due to a fall. It is estimated that approximately 28%–35% of people aged 65 over fall each year increasing to nearly 32%–42% for those over 70 years of age [2]. These falls may result in hip fractures, traumatic brain injuries and upper limb injuries. As a result, the elderly require to live in long-term care institutions. Moreover, falls also lead to a post-fall syndrome, such as loneliness, fear of social withdrawal, and depression [3]. In general, the quality of their lives is reduced.

Copyright © 2021. The Author(s). This is an open-access article distributed under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives License (CC BY-NC-ND 4.0 https://creativecommons.org/licenses/by-nc-nd/4.0/), which permits use, distribution, and reproduction in any medium, provided that the article is properly cited, the use is non-commercial, and no modifications or adaptations are made.

Article history: received June 13, 2020; revised November 6, 2020; accepted November 29, 2020; available online December 14, 2020.

Nearly half of the elderly are unable to get up after a non-injured fall and remain on the floor for a long time [4]. This may lead to a series of serious consequences, including pressure sores, dehydration, hypothermia, pneumonia and even death. Therefore, solutions with automatic fall detection and alarm notification functions are useful [5], because they ensure that the elderly get immediate assistance and reduce the risk of fall-related complications.

The solutions of fall detection are broadly classified into two categories: wearable solutions and non-wearable solutions. The latter concern the deployment of sensors in the surrounding environment where the elderly live, such as pressure sensors [6, 7], infrared sensors [8, 9] and cameras [10, 11]. The solutions based on these sensors distinguish between falls and *activities of daily living* (ADLs) by analyzing environmental information and/or the movement of the body within the detection range. But the main limitation of these solutions is that they can only be used in a specific room. In addition, the cost of deploying sensors, complex computing resources, and privacy of the elderly (especially camera-based methods) are also challenges they have to overcome [12].

Recently, the wearable solutions are research hotspots in fall detection studies. Current wearable devices with inertial sensors (such as accelerometer and/or gyroscope) are worn on the body and capture body posture changes to detect a fall. Compared with non-wearable solutions, they have the advantages of being available everywhere, lower cost and few privacy issues [13].

Methods for the wearable solutions can be divided into *threshold-based methods* (TBM) and *machine learning* (ML) [14]. The former have the advantages of low power consumption, low computational complexity and easy implementation. Wang *et al*. [15] proposed a wearable device (LPFD) based on accelerometer and air pressure sensing technology. The LPFD exploited a threshold-based fall detection algorithm to analyze kinematic information during a fall, including weightless falling, impact, and resting phases of the fall. If all actions matched the characteristics of the fall, the LPFD detected a fall event. However, the classification ability of the algorithm is poor, because it is difficult to tune an optimal classification threshold to trade off between sensitivity and specificity. A high threshold value brings out a large number of missing alarms, while a low threshold brings out a large number of false alarms. Furthermore, Aziz *et al*. [16] compared TBM with ML (including logistic regression, decision tree, naïve bayes, k-nearest neighbor, support vector machines), and claimed that the ML has better performance than the TBM.

Due to the limitations of the microcontroller in storage and computing resources, the sophisticated ML algorithm is implemented in devices with stronger processing capacity, however, wearable devices based on the microcontroller only collect and transmit body posture data. For example, Ruben *et al*. [17] proposed a fall detection algorithm based on a *convolutional neural network* (CNN). Although the algorithm achieved high accuracy (98%), the wearable device needed to transmit acceleration data in real time to a host for running the algorithm. The peak current is very high when the wearable device is performing wireless communication. Therefore, the weakness of this solution is high power consumption.

Ahsan *et al*. [18] proposed a fall detection system based on a smart phone: FallDroid. Similarly to our work, the FallDroid exploited a two-step algorithm to distinguish between falls and ADLs using a triaxial accelerometer. The first step of the algorithm relied on the TBM algorithm to effectively discard most of the ADLs. In the second step, the sophisticated *multiple kernels learning support vector machine* (MKL-SVM) was used to classify the ADLs that were not discarded by the TBM and the actual fall event. Sensitivity, specificity and accuracy of the algorithm at the thigh position were 95.8%, 88% and 91.7%, respectively. The algorithm runs most of the time at the TBM stage which reduces the average computational cost. However, some power is still wasted when the smart phone is used for other purposes in real life, such as active

Metrol. Meas. Syst., Vol. 28 (2021), No. 1, pp. 23–40
DOI: 10.24425/mms.2021.135999

wireless services and running other applications. In addition, the smart phone cannot be attached to the body in some situations, as a result, its fall-detection function is disabled.

In summary, limiting our solutions to the wearable solutions based on the TBM and/or ML, we found that the TBM with low classification ability has low power consumption and can be fully implemented in a wearable device. On the other hand, the ML can achieve satisfactory classification accuracy, but the battery life is shorter due to the extra power consumption caused by real-time data transmission. Besides, some solutions that apply the TBM and ML to the smart phone also ignore the fact that the specific function may be disabled. Therefore, it is essential to design an efficient and reliable wearable fall detection solution.

To overcome these limitations, in this we paper present a wearable device (*i.e.* TTXFD) based on MPU6050, which can collect and store acceleration samples when the microcontroller is in the sleep state (saving energy). At the same time, in this paper we also propose a two-step fall detection algorithm including the TBM stage and the ML stage. The TTXFD exploits the TBM stage of the two-step algorithm to reject as many ADLs as possible while transmitting all real fall events to a server. After receiving the acceleration samples transmitted by the TTXFD, the server executes the ML stage of the two-step algorithm. The acceleration samples are encoded as an image through data normalization, polar coordinate transformation, and mapping to image steps, successively. And then the fall detection algorithm based on CNN is used to identify a fall according to the image. Finally, the classification results of the two-step algorithm are reported and analyzed.

The rest of the paper is structured as follows: In Section 2, the hardware implementation of the TTXFD is introduced. The two-step fall detection algorithm is described in Section 3, including the TBM stage and the ML stage. Section 4 contains the experimental results. Section 5 discusses and analyzes the obtained results. Finally, Section 6 concludes our work.

2. Hardware implemented

The hardware of the TTXFD includes a 3.7 V, 500 mAh lithium battery, a low dropout voltage regulator (TPS78001), a microcontroller (MSP430F149), a motion processing unit (MPU6050), and a transmitting module (WH-NB73). Figure 1 shows the hardware structure of the TTXFD.

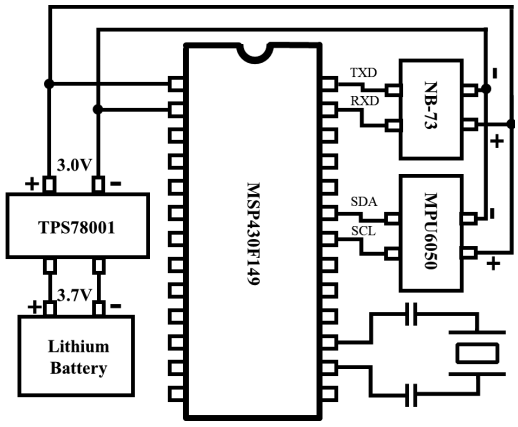


Fig. 1. The hardware structure of the TTXFD.

The TPS78001 is a low dropout voltage regulator with ultra-low power consumption that is compatible with similar products such as MSP430F149. The lower the input voltage of the microcontroller is, the lower the current will be. Therefore, the use of the TPS78001 can prolong the battery life. We have also designed a regulator using the TPS78001 which converts the lithium battery voltage to 3 V to power all components on the TTXFD.

The MPU6050 contains a 3-axis accelerometer, a 3-axis gyroscope, a 1024-byte *first-in-first-out* (FIFO) buffer, and a programmable interrupt system with the capability of capturing multiple phases of a fall. If the interrupt and the FIFO of the MPU6050 are enabled, the MPU6050 can automatically collect acceleration samples and store them into the FIFO, while the microcontroller is allowed to go into the sleep mode. The microcontroller is woken up only when the acceleration sample exceeds a pre-configured programmable threshold (the interrupt of the MPU6050 is triggered). In this way, the TTXFD cuts back on the workload generated by the microcontroller collecting and processing all acceleration samples in real time, reducing power consumption. Furthermore, the gyroscope function is turned off to save energy, the accelerometer's measurement range is configured to be ±16 g, and its sampling rate is 50 Hz as a trade-off between the fall detection rate and power efficiency.

The WH-NB73 is a product that transfers data between the serial device and the server through *Narrow Band Internet of Things* (NB-IoT), and it fits the usage scenario of being perfectly battery-powered. Appropriate use of the WH-NB73's *power saving mode* (PSM) can make the WH-NB73 sleep for a long time to save energy. When the TTXFD exploits the TBM stage to detect a suspected fall, the WH-NB73 automatically activates the network connection, exits the PSM mode, and transfers data. If there is no task to transmit data for a period of time (set to 1 min), the WH-NB73 goes into the PSM mode again.

The wearable device in the fall detection study is typically worn on the neck, wrist, foot, or thigh. Even though using a neck lanyard on the torso is more comfortable [18], the device generates swinging motions due to its poor consistency with the body. As a result, its sensitivity is poor. The SmartWatch (wrist) [19] or SmartShoe (foot) [20] is more aesthetically pleasing, but they are prone to report fall-like events, leading to higher false alarm rates. In addition, the device needs to frequently transmit data to the server as fall-like events increase, reducing battery life. Compared with the above three locations, a thigh-worn wearable device is the most acceptable. The TTXFD is only the size of a coin (28 mm × 25 mm). When the TTXFD is placed in the pants pocket, it can avoid social stigma due to its invisibility. Thus, the TTXFD is worn on the thigh in this study, whose prototype and its location are shown in Fig. 2.

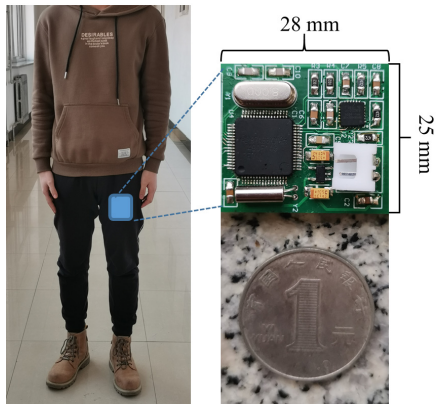


Fig. 2. The wearable device is worn on the thigh, its prototype and size.

Metrol. Meas. Syst., Vol. 28 (2021), No. 1, pp. 23–40
DOI: 10.24425/mms.2021.135999

3. Two-step fall detection algorithm

The proposed two-step fall detection algorithm includes TBM stage and ML stage. The TBM stage implemented in the TTXFD can effectively reject many ADLs, while transmitting suspected fall events (including all falls and some ADLs) to the server. The ML stage implemented in the server further detects the suspected fall events and identifies the actual fall events. The flow chart of the two-step fall detection algorithm is shown in Fig. 3.

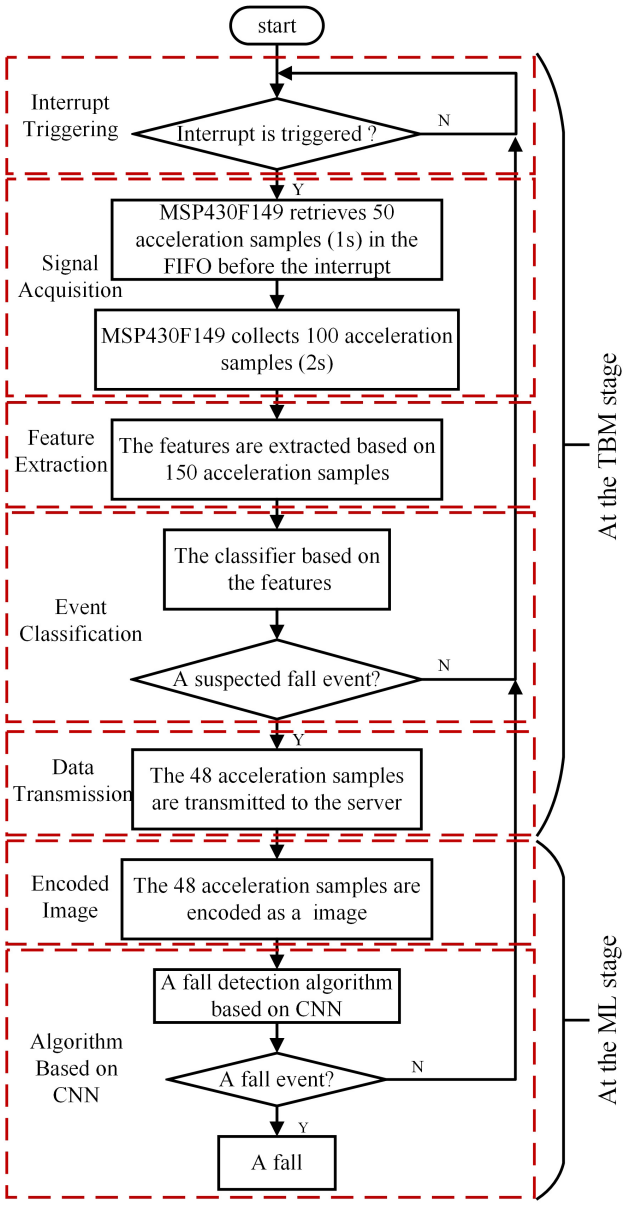


Fig. 3. Flowchart of the two-step fall detection algorithm, including the TBM stage and the ML stage.

3.1. The design of TBM stage

If the TBM stage captures the weightless falling phase and the impact phase of a fall successively, a suspected fall is detected. The TBM stage includes five procedures: interrupt triggering, signal acquisition, feature extraction, event classification and data transmission (see Fig. 3).

Interrupt triggering: the TTXFD only utilizes the MPU6050 to collect acceleration samples and store them in the FIFO and other components (*e.g.* the microcontroller and the WH-NB73) go into low power mode to prolong battery life. When the MPU6050 captures the weightless falling phase of the fall, the Free Fall Interrupt of the MPU6050 will be triggered. The interrupt triggered condition is that the *maximum absolute value* (*MAV*) of the triaxial acceleration is less than the preset threshold (*i.e.* $MAV[i] < th_0$). The *MAV* is calculated as follows:

$$MAV[i] = \max\left\{|a_x[i]|,\ |a_y[i]|,\ |a_z[i]|\right\}, \tag{1}$$

where $a_x[i]$, $a_y[i]$ and $a_z[i]$ represent the i-th acceleration sample along x-, y-, and z-axis, respectively.

Signal acquisition: after the Free Fall Interrupt is triggered, the microcontroller is activated and collects a 3 s *data window* (*DW*) for subsequent feature extraction and data transmission procedures. The *DW* consists of two parts, including 50 acceleration samples (1 s) retrieved from the FIFO before the interruption and 100 new acceleration samples (2 s) collected by the microcontroller (the span of the *DW* covers the weightless falling phase and the impact phase of the fall).

Feature extraction: the minimum ($SMV_{\min}$) and maximum ($SMV_{\max}$) of the sum magnitude vector are extracted from the *DW*. The calculation formula of the *sum magnitude vector* (*SMV*) is as follows:

$$SMV[i] = \sqrt{a_x^2[i] + a_y^2[i] + a_z^2[i]}, \tag{2}$$

where $a_x[i]$, $a_y[i]$ and $a_z[i]$ represent the i-th acceleration sample in the *DW* along the x-, y-, and z-axis, respectively. During the weightless falling phase, the body moves downward and *SMV* decreases to less than 1 g. The *SMV* has a large peak when the body hits the ground (the impact phase of the fall). Thus, the $SMV_{\min}$ and $SMV_{\max}$ in the *DW* can capture these two key phases of the fall.

Event classification: if both $SMV_{\min}$ and $SMV_{\max}$ exceed the corresponding thresholds (*i.e.* $SMV_{\min} < th_1$ and $SMV_{\max} > th_2$), the current event is classified as a suspected fall. Otherwise, it is classified as ADLs and the TBM stage returns to the interrupt trigger procedure.

Data transmission: when a suspected fall is detected at the TBM stage, the data transmission procedure is executed. The TTXFD transmits the time series of *SMV* (48 *SMV* samples) to the server through the WH-NB73, which is centered around the time of the $SMV_{\max}$. Figure 4 shows the line chart for *SMV* ($SMV = \{SMV_1, SMV_2, \ldots, SMV_{48}\}$) of 9 ADLs (getting up, lying down, sitting down, standing up, going up, going down, walking, running, and jumping) and a simulated fall. The difference between the highest and lowest of the *SMV* of d) running, f) jumping and j) a simulated fall are the biggest due to the fact that these activities are more intense. The *SMV* of e) walking is the most stable.

Metrol. Meas. Syst., Vol. 28 (2021), No. 1, pp. 23–40
DOI: 10.24425/mms.2021.135999

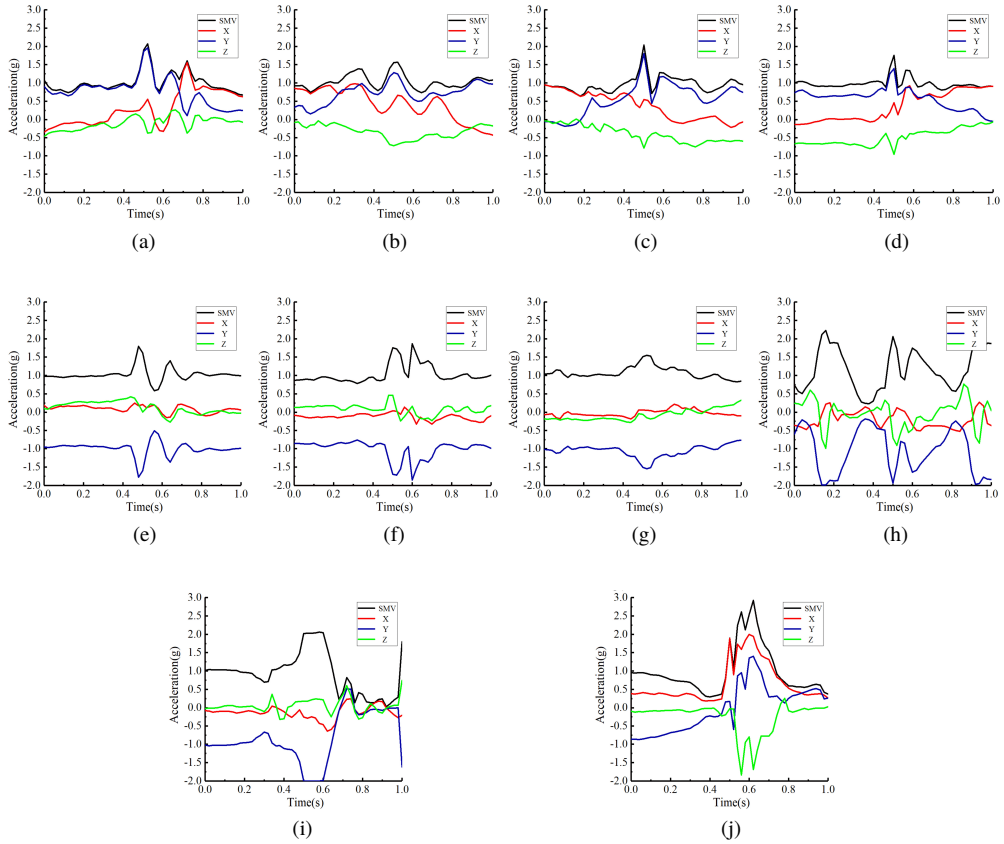


Fig. 4. The line charts for 9 ADLs and a fall of *SMV*: (a) standing up, (b) getting up, (c) going up, (d) running, (e) walking, (f) jumping, (g) going down, (h) lying down, (i) sitting down, (j) a simulated fall.

3.2. The design of ML stage

In order to improve the accuracy of fall detection at the ML stage, we apply machine learning technology to further detection of the suspected fall data (48 *SMV* samples). The CNN has become the state-of-the-art technology for image recognition tasks [21]. Therefore, we encode the time series of *SMV* as an image, and design a CNN-based fall detection algorithm to identify a fall on the basis of the image.

3.2.1. Encoded image

At the ML stage, firstly, the time series of *SMV* is encoded as an image using the method proposed by Wang *et al*. [22]. It includes three steps: data normalization, polar coordinate transformation, and mapping to image.

Data normalization: according to (3), the *SMV* is normalized to between −1 and 1 for subsequent the polar coordinate transformation step, *i.e*. $\overline{SMV} = \left\{\overline{SMV}_1, \overline{SMV}_2, \ldots, \overline{SMV}_{48}\right\}$.

$$\overline{SMV}_i = \frac{2 \times SMV_i - \max(SMV) - \min(SMV)}{\max(SMV) - \min(SMV)}. \tag{3}$$

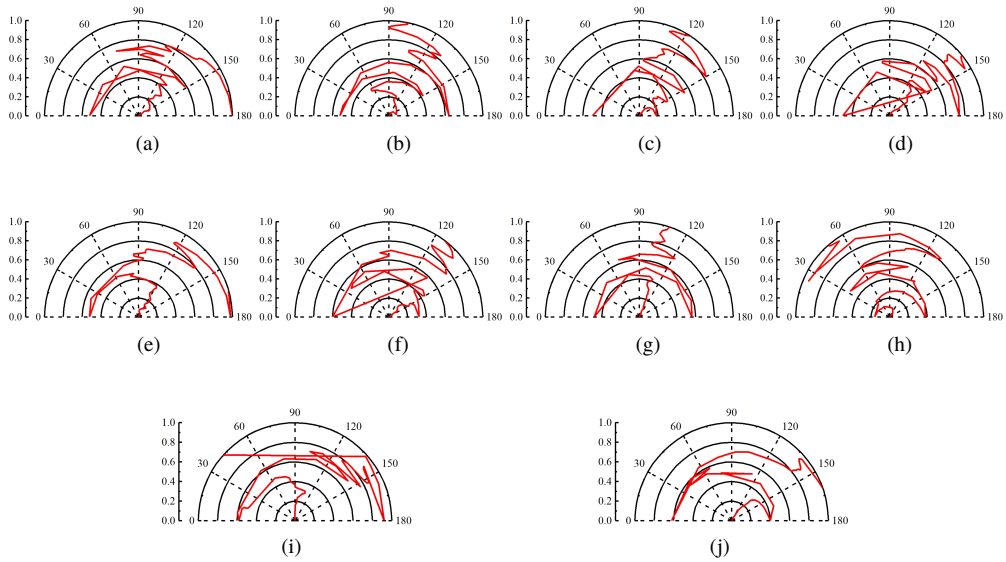


Fig. 5. 9 ADLs and a fall of the *SMV* in the polar coordinate system: (a) standing up, (b) getting up, (c) going up, (d) running, (e) walking, (f) jumping, (g) going down, (h) lying down, (i) sitting down, (j) a simulated fall.

Polar coordinate transformation: we convert the $\overline{SMV}$ to the coded value in the polar coordinate system according to (4), including angle ($\phi = \{\phi_1, \phi_2, \ldots, \phi_{48}\}$) and radius ($r = \{r_1, r_2, \ldots, r_{48}\}$) where t_i ($t_i = 0.02 \times i$) is the time stamp. Figure 5 shows that the ϕ and r in the polar coordinate system. It can be seen that all types of activities (9 ADLs and the fall) warp among different angular points in a semicircle while polar coordinates preserve absolute temporal relationship. On the other hand, due to the data being normalized, the ϕ is between 0 and π. This ensures that the data has a unique corresponding result in two different coordinate systems. It can be observed that the angle change rules of all activities are different in the polar coordinate system when the values of the radius change.

$$\begin{cases} \phi_i = \arccos\left(\overline{SMV}_i\right), & -1 \leq \overline{SMV}_i \leq 1 \\ r_i = t_i \end{cases}. \tag{4}$$

Mapping to image: the ϕ is mapped to an image according to (5), which exploits trigonometric sum to retain the relevance of ϕ at different time intervals. The trigonometric sum at each time interval is a pixel of the image. In this way, 48 triaxial acceleration samples are mapped into a 48×48 pixel image (see Fig. 6). It can be seen that the image based on the fall is different from the images based on 9 ADLs which verifies the correctness of the image-based method to detect falls.

$$Image = \begin{bmatrix} \cos(\phi_1 + \phi_1) & \cdots & \cos(\phi_1 + \phi_{48}) \\ \cos(\phi_2 + \phi_1) & \cdots & \cos(\phi_2 + \phi_{48}) \\ \vdots & \ddots & \vdots \\ \cos(\phi_{48} + \phi_1) & \cdots & \cos(\phi_{48} + \phi_{48}) \end{bmatrix}. \tag{5}$$

Metrol. Meas. Syst., Vol. 28 (2021), No. 1, pp. 23–40
DOI: 10.24425/mms.2021.135999

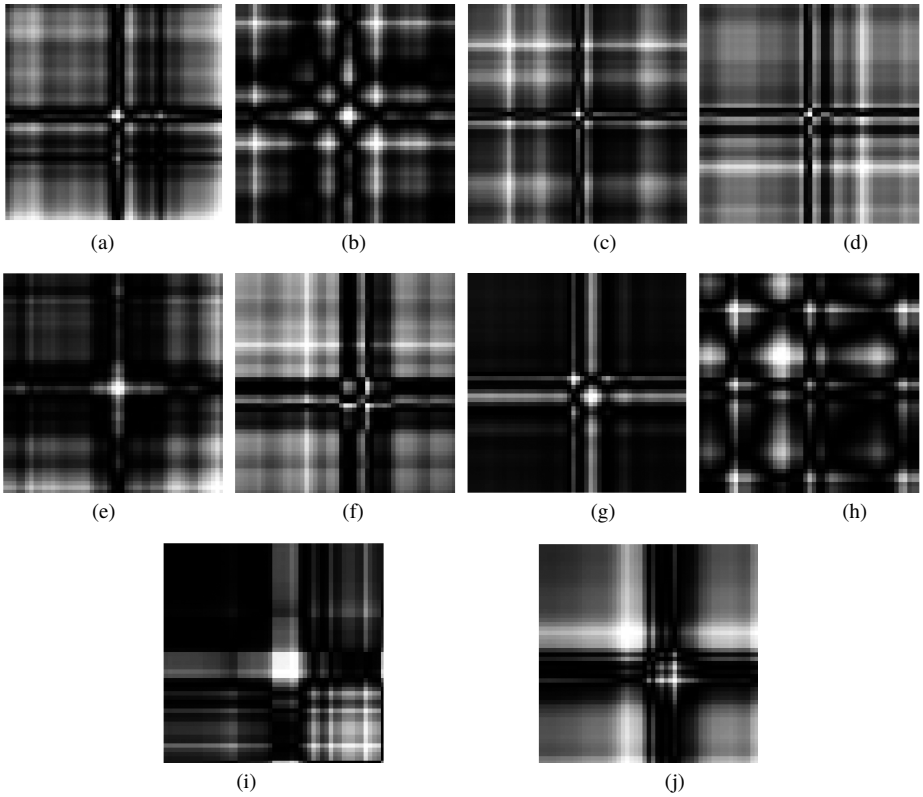


Fig. 6. Images of 9 ADLs and a fall through normalization, polar coordinate conversion and mapping to image steps: (a) standing up, (b) getting up, (c) going up, (d) running, (e) walking, (f) jumping, (g) going down, (h) lying down, (i) sitting down, (j) a simulated fall.

3.2.2. The fall detection algorithm based on CNN

LeNet-5 [23] is specifically designed to deal with variability of 2D shapes and seems to outperform other techniques. In this paper, the fall detection algorithm based on CNN is designed following the structure of the LeNet-5 which is composed of 7 layers (not counting the input layer) including two convolutional layers(C1 and C3), two max poolings (S2 and S4), two fully connected layers (F5 and F6) and an output layer. The architecture of the CNN-based fall detection algorithm is shown in Fig. 7.

C1 layer: the filters, with the size of 3×3, extract hierarchical characteristics from the input image to improve the classification performance. The C1 layer preserves the boundary information of the input image by padding the edges while making the image and feature map the same size. The size of 6 feature maps in the C1 layer is 48×48. In addition, we configure the *Rectified Linear Unit* (ReLU) as an activation function to increase the learning speed and reduce the effect of the vanishing gradient. The C1 layer has a total of 60 trainable parameters.

S2 layer: it can reduce the computational complexity of the network by reducing the data dimensionality and retain useful features. The S2 layer is a sub-sampling layer with 6 feature maps. The size of receptive fields is 2×2 and they are non-overlapping, thus the height and width of feature maps in the S2 layer are half of the C1 layer (*i.e.* the size of feature maps in the S2 layer is 24×24).

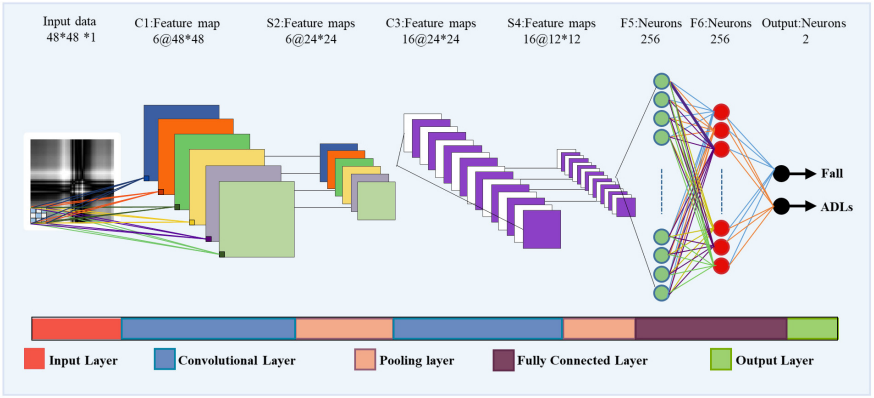


Fig. 7. Architecture of the CNN-based fall detection algorithm at the ML stage.

C3 layer: the size of filter is $3 \times 3 \times 6$. The C3 layer also preserves the boundary information by padding edges, therefore the size of 16 feature maps is 24×24. In addition, we also configure the ReLU as an activation function. The C3 layer has a total of 880 trainable parameters.

S4 layer: the S4 layer is a sub-sampling layer with 16 feature maps. The size of receptive fields is 2×2 and they are non-overlapping, thus the height and width of feature maps in the S4 layer are half of the C3 layer (*i.e.* the size of feature maps in the S4 layer is 12×12).

F5 layer: the F5 with a fully connected layer is added behind the S4 layer. It contains 256 neurons and is configured for the ReLU activation function. The F5 layer has 590080 trainable parameters. Moreover, dropout is configured when training the model for reducing overfitting.

F6 layer: similarly, the F6 layer is also a fully connected layer. It contains 256 neurons and uses the ReLU activation function. The F6 layer has 65536 trainable parameters. In addition, dropout is also configured when training the model for reducing overfitting.

Output layer: it is fully connected to the F6 layer. The number of neurons in the output layer is determined by the number of classifications required by the model and this study is a binary classification (ADLs or fall), thus the output layer has two neurons and we configure the Softmax as an activation function.

4. Experiment and results

To evaluate the two-step fall detection algorithm, two open datasets including UniMiB SHAR [24] and MobiAct [25] are selected. In our experiments, the UniMiB SHAR (the training dataset) is used to tune thresholds ($th_0 - th_2$) of the TBM stage and train the CNN-based fall detection algorithm of the ML stage in the two-step algorithm. The MobiAct (the testing dataset) is used to verify the classification performance (the sensitivity, specificity and accuracy) of the optimized two-step algorithm.

4.1. Datasets

The training dataset is obtained by extracting the UniMiB SHAR which involves 30 healthy subjects (24 males and 6 females, age: 27 ± 11 years, height: 169 ± 7 cm, weight: 64.4 ± 9.7 kg) and contains 9 types of ADLs and 8 types of falls. The dataset collected from the thigh with a sampling rate of 50 Hz. As a result, a total of 7565 ADLs and 4170 falls in the training dataset (see Table 1).

Metrol. Meas. Syst., Vol. 28 (2021), No. 1, pp. 23–40
DOI: 10.24425/mms.2021.135999

Table 1. The training dataset.

No.	Category	Instructions	Number of Samples
1	ADLs	Sitting down on a chair	150
2	ADLs	Getting up from a chair	920
3	ADLs	Lying on a bed	295
4	ADLs	Getting up from a bed	215
5	ADLs	Walking	1735
6	ADLs	Going upstairs	200
7	ADLs	Going downstairs	1320
8	ADLs	Running	1985
9	ADLs	Jumping	745
10	Fall	Forwards	525
11	Fall	Backwards	510
12	Fall	Leftwards	525
13	Fall	Rightwards,	510
14	Fall	Falling with contact with an obstacle	660
15	Fall	Falling while sitting down on a chair	430
16	Fall	Falling using compensation strategies to prevent the impact	480
17	Fall	Syncope	530
A total of 7565 ADLs and 4170 falls in the training dataset.			

The testing dataset is obtained by extracting the MobiAct, which involves 57 healthy subjects (42 males and 15 females, age: 25 ± 4 years, height: 175 ± 7 cm, weight: 76.6 ± 14.5 kg) and contains 9 types of ADLs and 4 types of falls. The dataset collected from the thigh with a sampling rate of 100 Hz. As a result, a total of 1280 ADLs and 600 falls on the testing dataset (see Table 2).

Table 2. The testing dataset.

No.	Category	Instructions	Number of Samples
1	ADLs	Standing with subtle movements	40
2	ADLs	Normal walking	40
3	ADLs	Jogging	100
4	ADLs	Continuous jumping	100
5	ADLs	Stairs up (10 stairs)	200
6	ADLs	Stairs down (10 stairs)	200
7	ADLs	Sitting on a chair	200
8	ADLs	Step in a car	200
9	ADLs	Step out of a car	200
10	Fall	Fall forward from standing, use of hands to dampen the fall	150
11	Fall	Fall forward from standing, first impact on knees	150
12	Fall	Fall sideward from standing, bending legs	150
13	Fall	Fall backward while trying to sit on a chair	150
A total of 1280 ADLs and 600 falls in the training dataset			

4.2. Performance indicators

In this study, there are four possible situations for classification results: 1) *TP*: Falls are correctly identified. 2) *FP*: ADLs are wrongly identified. 3) *TN*: ADLs are correctly identified. 4) *FN*: falls are wrongly identified. We report the performance of the algorithm in terms of multiple performance indicators, including *sensitivity* (*Sen*), *specificity* (*Spc*), and *accuracy* (*Acc*) based on *TP*, *FP*, *TN*, and *FN*. These three indicators are calculated by (6), (7) and (8), respectively.

$$Sen = \frac{TP}{TP + FN}, \tag{6}$$

$$Spc = \frac{TN}{TN + FP}, \tag{7}$$

$$Acc = \frac{TN + TP}{TP + TN + FP + FN}. \tag{8}$$

4.3. Experimental results

There are three thresholds ($th_0 - th_2$) at the TBM stage of the two-step fall detection algorithm. The th_0 is used to trigger the Free Fall Interrupt inside the MPU6050. The th_1 and th_2 are the corresponding thresholds of $SMV_{\min}$ and $SMV_{\max}$ features from the classifier, respectively. The acceleration samples on the training dataset are processed and the TBM stage of the two-step detection program is simulated in MATLAB. During the training procedure, three parameters ($th_0 - th_2$) at the TBM stage are tuned through *particle swarm optimization* (PSO) [26]. The target of this optimization is to obtain the maximum value of the fitness function, which is calculated as follows:

$$Fitness = \begin{cases} 1 + Spc, & \text{if } Sen = 100\% \\ 0, & \text{if } Sen \neq 100\% \end{cases}. \tag{9}$$

Figure 8 shows the flow of the PSO algorithm which includes seven steps: initialization, calculation of the fitness function values, calculation of the personal best (*pbest*) and global best (*gbest*), updating the velocity and position of the each particle, updating the fitness function value, updating the *pbest* and *gbest*, outputting the optimal thresholds ($th_0 - th_2$).

Each particle inside the swarm contains a d-dimensional vector position and a d-dimensional vector velocity ($d = 3$). $x_i^k = \{x_{i,1}^k, x_{i,2}^k, \ldots, x_{i,d}^k\}$ represents the position of i-th particle at k-th iteration. $v_i^k = \{v_{i,1}^k, v_{i,2}^k, \ldots, v_{i,d}^k\}$ represents the velocity of i-th particle at k-th iteration. At the initialization step of the PSO algorithm, the positions of all particles are randomly placed between $x_{\min}$ and $x_{\max}$ while the velocities of all particles are randomly placed between $v_{\min}$ and $v_{\max}$. $x_{\min}$ and $x_{\max}$ respectively represent the minimum and maximum values of position. $v_{\min}$ and $v_{\max}$ respectively represent the minimum and maximum values of velocity ($v_{\min} = \{-0.05\text{ g}, -0.05\text{ g}, -0.05\text{ g}\}$, $v_{\max} = \{0.05\text{ g}, 0.05\text{ g}, 0.05\text{ g}\}$). The upper limit of th_0 and th_1 is 1 g (gravity magnitude) because of the Free Fall Interrupt and $SMV_{\min}$ is used to capture the weightless falling phase. The upper limit of th_2 is 16 g according to the measurement range of the accelerometer. Therefore, $x_{\min} = \{0\text{ g}, 0\text{ g}, 1\text{ g}\}$, $x_{\max} = \{1\text{ g}, 1\text{ g}, 16\text{ g}\}$.

After the initialization step, the fitness function values of particles in the swarm is calculated according to (9), and then *pbest* and *gbest* are obtained based on these fitness function values. At the step of updating velocity and position, velocity v_i^{k+1} is calculated as follows:

$$v_i^{k+1} = wv_i^k + c_1 r_1 \left(p_i^k - x_i^k\right) + c_2 r_2 \left(p_g^k - x_i^k\right), \quad i = 1, 2, \ldots, N, \quad k = 1, 2, \ldots, T, \tag{10}$$

Metrol. Meas. Syst., Vol. 28 (2021), No. 1, pp. 23–40
DOI: 10.24425/mms.2021.135999

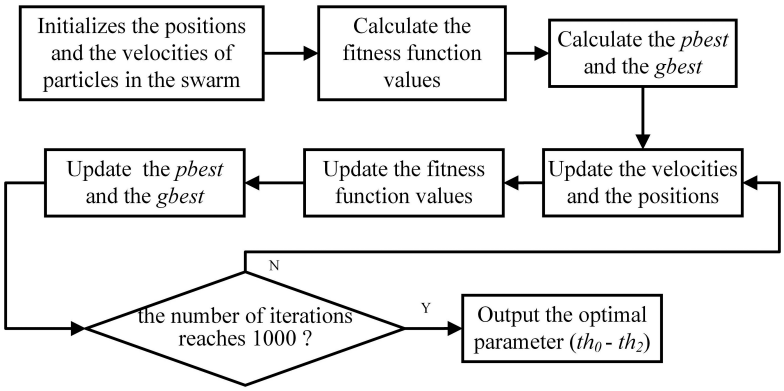


Fig. 8. Flowchart of the PSO algorithm.

where $p_i^k = \{p_{i,1}^k, p_{i,2}^k ..., p_{i,d}^k\}$ represents the *pbest* of $i-$th particle at k-th iteration, $p_g^k = \{p_{g,1}^k, p_{g,2}^k, \ldots, p_{g,d}^k\}$ represents the *gbest* of the swarm at k-th iteration, w represents inertia weight ($w = 0.9$), r_1 and r_2 are random values between [0,1], c_1 and c_2 represent cognitive coefficient and social coefficient, respectively ($r_1 = r_2 = 2$), N is the size of the swarm ($N = 30$), T is number of iterations ($T = 1000$). The updated velocity v_i^{k+1} is calculated as follows:

$$v_i^{k+1} = \begin{cases} v_i^{k+1} & \text{if } v_{\min} < v_i^{k+1} < v_{\max} \\ v_{\min} & \text{if } v_i^{k+1} \le v_{\min} \\ v_{\max} & \text{if } v_i^{k+1} \ge v_{\max} \end{cases}. \tag{11}$$

The updated position x_i^{k+1} is computed according to (12) and (13):

$$x_i^{k+1} = x_i^k + v_i^{k+1}, \tag{12}$$

$$x_i^{k+1} = \begin{cases} x_i^{k+1} & \text{if } x_{\min} < x_i^{k+1} < x_{\max} \\ x_{\min} & \text{if } x_i^{k+1} \le x_{\min} \\ x_{\max} & \text{if } x_i^{k+1} \ge x_{\max} \end{cases}. \tag{13}$$

After that, the fitness function values, the *pbest* and *gbest* are updated according to the x_i^{k+1}, successively. When the number of iterations reaches 1000, the PSO algorithm outputs the optimal thresholds, otherwise, it returns to update the particle's position and velocity.

The optimal parameters being the output of the PSO are: $th_0 = 0.65$ g, $th_1 = 0.72$ g, and $th_2 = 1.71$ g (see Table 3). In this case, the TBM stage of the two-step algorithm achieves *Sen* of 100%, *Spc* of 50.81%, and *Acc* of 68.29% on the training dataset.

Based on the optimized thresholds ($th_0 - th_2$) above, the TBM stage of the two-step algorithm detected 7891 suspected fall events (4170 falls and 3721 ADLs) on the training dataset. These suspected fall events were encoded as images in MATLAB as previously described in Section 3.2.1. The CNN-based fall detection program was simulated and the classification results of this stage were output in Python. During the training procedure, the images were fed by the simulated program. The learning rate was 0.0001, the batch size was 32, the loss function was cross entropy and the Adam algorithm is used to tune the model parameters. The target of

Table 3. Parameter settings and training results of the TBN stage from the PSO algorithm.

Parameters	Values	Optimal thresholds from PSO	Performance
$v_{\min}$	{ –0.05 g, –0.05 g, –0.05 g}		
$v_{\max}$	{ 0.05 g, 0.05 g, 0.05 g}		
$x_{\min}$	{ 0 g, 0 g, 1 g}		
$x_{\max}$	{ 1 g, 1 g, 16 g}		
c_1	2	th_0 = 0.65 g,	*Sen* = 100%,
c_2	2	th_1 = 0.72 g,	*Spc* = 50.81%,
r_1	[0–1]	th_2 = 1.71 g	*Acc* = 68.29%
r_2	[0–1]		
w	0.9		
T	30		
N	1000		

optimization was to minimize the error calculated by the loss function. When the CNN training iterated over 43 epochs, the accuracy kept stable (99.79%) based on the suspected fall data of the training dataset.

Before testing the procedure, the testing dataset was subsampled to 50 Hz. With the optimized thresholds ($th_0 - th_2$) and the trained CNN, the TBM stage of the two-step algorithm achieved *Sen* of 100%, *Spc* of 47.42%, and *Acc* of 64.20% on the testing dataset (*i.e.* the TBM stage rejecting 607 ADLs and detected 1273 suspected fall events including 600 falls and 673 ADLs, see Table 4), further the ML stage of the two-step algorithm achieved *Sen* of 97.83%, *Spc* of 93.61%, and *Acc* of 95.60% based on the suspected fall events. Finally, the two-step algorithm achieved *Sen* of 97.83%, *Spc* of 96.64%, and *Acc* of 97.02% on the testing dataset (see Table 5).

Table 4. The results for two-step algorithm on the testing dataset.

Input data: 1280 ADLs and 600 falls	
At the TBM Stage of two-step algorithm	Rejecting 607 ADLs, transmitting 673ADLs and 600 falls to the server
At the ML Stage of two-step algorithm	630 ADLs and 587 falls are correctly identified
Two-Step Algorithm	1237 ADLs and 587 falls are correctly identified

Table 5. The *Sen*, *Spc*, and *Acc* for two-step algorithm on the testing dataset.

Two-Step Algorithm	*Sen*	*Spc*	*Acc*
At the TBM stage of two-step algorithm	100%	47.42%	64.20%
At the ML stage of two-step algorithm	97.83%	93.61%	95.60%
Two-step algorithm	97.83%	96.64%	97.02%

5. Discussion

Compared with the fall detection algorithm based on either TBM or ML, we proposed a two-step fall detection algorithm using the MPU6050. Considering power consumption and classification accuracy, the method of transmitting real-time data to the server is effectively replaced by the TBM stage of the two-step algorithm and the ML stage of the two-step algorithm

Metrol. Meas. Syst., Vol. 28 (2021), No. 1, pp. 23–40
DOI: 10.24425/mms.2021.135999

makes the final decision. As a result, all actual fall events are picked up and transmitted to the server, while the two-step algorithm achieves high sensitivity (97.83%), specificity (96.64%) and accuracy (97.02%) on the open dataset.

CNN can not only automatically discover useful features without manual extraction, but also process the data in multiple arrays (images). Hence, we encode time series of triaxial accelerations as images with the relevance at different times and use the CNN-based fall detection algorithm to classify fall patterns from ADLs on the basis of the images. Compared with traditional *artificial neural networks* (ANN), the CNN-based fall detection algorithm at the ML stage outperforms the proposed algorithm based on ANN in [27] (see Table 6).

Table 6. A comparison of results presented in this article and some studies in the literature. ANN = artificial neural networks, TBM = threshold-based method, CNN = convolutional neural network, MKL-SVM = multiple kernel learning support vector machine.

Work	***Sensor***	**Position**	**Algorithm**	**Results**
Kerdegari *et al* [27]	accelerometer	waist	ANN	*Sen*: 93.03% *Spc*: 88.88% *Acc*: 91.25%
Li *et al* [28]	accelerometer and gyroscope	thigh and chest	TBM	*Sen*: 91.00% *Spc*: 92.00%
He *et al* [29]	accelerometer and gyroscope	waist	CNN	*Sen*: 97.44% *Spc*: 99.63% *Acc*: 97.47%
Ahsan *et al* [18]	accelerometer	thigh	TBM+MKL−SVM	*Sen*: 95.80% *Spc*: 88.00% *Acc*: 91.70%
Our work	accelerometer	thigh	TBM+CNN	*Sen*: 97.83% *Spc*: 96.64% *Acc*: 97.02%

We know that the test results depend on how real-world fall data can reflect the authentic performance of the algorithm. Because an accidental fall is a rare event, it is difficult to collect a sufficient amount of fall data. Therefore, most of the fall detection solutions in the literature are developed by simulated fall data which involves young subjects. However, different age groups have different body posture changes when they fall, so the performance of the two-step algorithm is likely to decrease in the real life.

Table 6 shows some solutions similar to our work and compares them to several variables, including sensor, placement locations, algorithm, and classification performance. Li *et al*. [28] proposed a wearable fall detection device based on an accelerometer and a gyroscope, which used the threshold-based method to detect fall events. However, the device was equipped with multiple sensors (in fact there were two accelerometers and two gyroscopes) with a higher sampling rate (120 Hz). Hence, more power was consumed in theory. On the other hand, it needed to be placed in multiple locations (the thigh and the chest) on the body to work which increased the intrusion and reduces the acceptability of the elderly. In addition, they did not consider exploiting machine learning technology to improve the detection accuracy, providing a low sensitivity (91.00%) and specificity (92.00%) of the device. He *et al*. [29] proposed a fall detection algorithm called an FD-CNN. Similarly, acceleration and angular velocity samples (2 s sliding window) were transmitted to the server and encoded as an RGB image which was used as the input of the FD-CNN. However, The method of encoding images was simply to arrange these samples chronologically to generate a 20×20 RGB image (each sample is a pixel). This resulted in poor relevance of the image at different pixels which made the sensitivity level of their device lower than in our

work. Ahsan *et al*. [18] proposed a fall detection algorithm based on the TBM and MKL-SVM which extracted 15 complex features from acceleration samples. In this solution a smart phone performed power-intensive signal processing tasks for a long time, reducing battery life. On the other hand, the features manually extracted from the raw data (acceleration samples) are not always the parameters which have the ability to distinguish between falls and ADLs, so the performance of the algorithm (sensitivity of 95.8%, specificity of 88.0%, accuracy of 91.7%) is also lower than in our work. What is more, the smart phone is not always placed where it matches the specific function, such as the fall-detection function is disabled when watching a movie with the smart phone.

6. Conclusions

This paper proposes a pervasive fall detection solution which consists of a wearable device and a two-step fall detection algorithm including the TBM stage and ML stage. The device utilizes a TBM with low computational complexity to filter out simple ADLs and transmit the actual fall data to the server without uploading all data in real time, resulting in reducing power consumption of the device. At the same time, considering the advantages of the CNN at the image classification task, we encode the uploaded data as an image and use the CNN-based method to further detect fall events on the server. Experimental results on the open dataset including 1280 ADLs and 600 falls demonstrate the satisfactory performance of the two-step algorithm, achieving accuracy of 97.02%, sensitivity of 97.83%, specificity of 96.64%, respectively.

In the future, we plan to prolong battery life with methods based both on hardware and firmware, such as dynamically adjusting the sampling rate of the sensor, using multiple interrupts to capture multiple key phases of a fall instead of feature extraction procedures at the TBM stage as well as choosing components with lower power consumption. The human experiments will be carried out to collect body posture data in daily life and the battery life will be estimated based on the data. Another potential future work is to refine the two-step algorithm to make it independent from the sampling rate, measurement range, placement location and dataset so that it could be adapted for different elderlies. Besides, classification performance of the two-step algorithm of the wearable device will be evaluated in in real life.

Acknowledgements

The authors would like to thank the colleagues in the project, as well as friends who have supported this work.

References

[1] Gibson, M. J. (1987). The prevention of falls in later life: a report of the Kellogg International Work Group on the prevention of falls by the elderly. *Dan Med Bull*, *34*(4), 1–24.

[2] Lord, S. R., Sherrington, C., Menz, H. B., & Close, J. C. (2007). *Falls in older people: Risk factors and strategies for prevention*, U.K. Cambridge University Press.

[3] Todd, C., & Skelton D. (2004). *What are the main risk factors for falls amongst older people and what are the most effective interventions to prevent these falls?* World Health Organization.

[4] Tinetti, M. E., Liu, W. L., & Claus, E. B. (1993). Predictors and prognosis of inability to get up after falls among elderly persons. *JAMA – Journal of the American Medical Association*, *269*(1), 65–70. https://doi.org/10.1001/jama.1993.03500010075035

Metrol. Meas. Syst., Vol. 28 (2021), No. 1, pp. 23–40
DOI: 10.24425/mms.2021.135999

[5] Wójtowicz, B., Dobrowolski, A., & Tomczykiewicz, K. (2015). Fall detector using discrete wavelet decomposition and SVM classifier. *Metrology and Measurement Systems*, *22*(2), 303–314. https://doi.org/10.1515/mms-2015-0026

[6] Muheidat, F., Tawalbeh, L., & Tyrer, H. (2018). Context-aware, accurate, and real time fall detection system for elderly people. *Proceedings of 2018 IEEE 12th International Conference on Semantic Computing (ICSC)*, Canada, 329–333. https://doi.org/10.1109/ICSC.2018.00068

[7] Chaccour, K., Darazi, R., Hassans, A. H. E., & Andres, E. (2015). Smart carpet using differential piezoresistive pressure sensors for elderly fall detection. *Proceedings of 2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications*, United Arab Emirates, 225–229. https://doi.org/10.1109/WiMOB.2015.7347965

[8] Sixsmith, A., Johnson, N., & Whatmore, R. W. (2005). Pyroelectric IR sensor arrays for fall detection in the older population. *Journal de Physique IV France*, *128*, 153–160. https://doi.org/10.1051/jp4:2005128024

[9] Nishiguchi, S., Yamada, M., Uemura, K., Matsumura, T., & Aoyama, T. (2013). A novel infrared laser device that measures multilateral parameters of stepping performance for assessment of fall risk in elderly individuals. *Aging Clinical and Experimental Research*, *25*(4), 311–316. https://doi.org/10.1007/s40520-013-0042-9

[10] Zhao, F., Cao, Z., Xiao, Y., Mao, J., & Yuan, J. (2019). Real-time detection of fall from bed using a single depth camera. *IEEE Transactions on Automation Science & Engineering*, *16*(3), 1018–1032. https://doi.org/10.1109/TASE.2018.2861382

[11] Gasparrini, S., Cippitelli, E., Spinsante, S., & Gambi, E. (2014). A depth-based fall detection system using a kinect sensor. *Sensors*, *12*(4), 2756–2775. https://doi.org/10.3390/s140202756

[12] Delahoz, Y. S., & Labrador, M. A. (2014). Survey on fall detection and fall prevention using wearable and external sensors. *Sensors*, *14*(10), 19806-19842. https://doi.org/10.3390/s141019806

[13] Ren, L., & Peng, Y. (2019). Research of fall detection and fall prevention technologies: A systematic review. *IEEE Access*, *7*, 77702–77722. https://doi.org/10.1109/ACCESS.2019.2922708

[14] Pannurat, N., Thiemjarus, S., & Nantajeewarawat, E. (2014). Automatic fall monitoring: A review. *Sensors*, *14*(7), 12900-12936. https://doi.org/10.3390/s140712900

[15] Wang, C., Lu, W., Narayanan, M. R., Chang, D. W., Lord, S. R., Redmond, S. J., & Lovell, N. H. (2016). Low-power fall detector using triaxial accelerometry and barometric pressure sensing. *IEEE Transactions on Industrial Informatics*, *12*(6), 2302–2311. https://doi.org/10.1109/TII.2016.2587761

[16] Aziz, O., Musngi, M., Park, E. J., Mori, G., & Robinovitch, S. N. (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. *Medical & Biological Engineering & Computing*, *55*(1), 45–55. https://doi.org/10.1007/s11517-016-1504-y

[17] Delgado-Escano, R., Castro, F. M., Cózar, J. R., Marín-Jiménez M. J., & Casilari, E. (2020). A cross-dataset deep learning-based classifier for people fall detection and identification. *Computer Methods and Programs in Biomedicine*, *184*, 105265. https://doi.org/10.1016/j.cmpb.2019.105265

[18] Shahzad, A., & Kim, K. (2019). Falldroid: An automated smart-phone-based fall detection system using multiple kernel learning. *IEEE Transactions on Industrial Informatics*, *15*(1), 35–44. https://doi.org/10.1109/TII.2018.2839749

[19] Yuan, J., Tan, K. K., Lee, T. H., & Koh, G. C. H. (2015). Power-efficient interrupt-driven algorithms for fall detection and classification of activities of daily living. *IEEE Sensors Journal*, *15*(3), 1377–1387. https://doi.org/10.1109/JSEN.2014.2357035

[20] Montanini, L., Campo, A. D., Perla, D., Spinsante, S., & Gambi, E. (2018). A footwear-based methodology for fall detection. *IEEE Sensors Journal*, *18*(3), 1233–1242. https://doi.org/10.1109/JSEN.2017.2778742

[21] Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

[22] Wang, Z., & Oates, T. (2015). Encoding time series as images for visual inspection and classification using tiled convolutional neural networks. *Proceedings of Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, USA, 40–46.

[23] Lecun, Y., & Bottou, L. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

[24] Micucci, D., Mobilio, M., & Napoletano, P. (2017). UNIMIB SHAR: A new dataset for human activity recognition using acceleration data from smartphones. *Applied Sciences*, *7*(10), 1101. https://doi.org/10.3390/app7101101

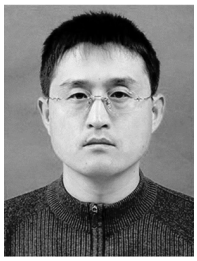
[25] Vavoulas, G., Chatzaki, C., Malliotakis, T., Pediaditis, M., & Tsiknakis, M. (2016). The MobiAct dataset: Recognition of activities of daily living using smartphones. *Proceedings of the International Conference on Information and Communication Technologies for Ageing Well and e-Health*, Italy, 143–151. https://doi.org/10.5220/0005792401430151

[26] Asrul, A., Ibrahim, S. M., Zaidi, M. T. M., Saberi, M. M., & Marizan, M. (2014). Feature selection and classifier parameters estimation for EEG signals peak detection using particle swarm optimization. *The Scientific World Journal*, *2014*, 973063. https://doi.org/10.1155/2014/973063

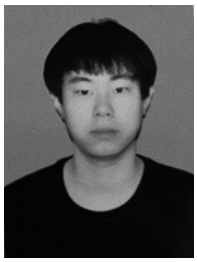
[27] Kerdegari, H., Mokaram, S., Samsudin, K., & Ramli, A. R. (2015). A pervasive neural network based fall detection system on smart phone. *Journal of Ambient Intelligence & Smart Environments*, *7*(2), 221–230. https://doi.org/10.3233/AIS-150306

[28] Li, Q., Stankovic, J. A., Hanson, M. A., Barth, A. T., Lach, J., & Zhou, G. (2009). Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information. *Proceedings of 2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*, United States, 138–143. https://doi.org/10.1109/BSN.2009.46

[29] He, J., Zhang, Z., Wang, X., & Yang, S. (2019). A low power fall sensing technology based on FD-CNN. *IEEE Sensors Journal*, *19*(13), 5110–5118. https://doi.org/10.1109/JSEN.2019.2903482



Tao Xu received the Ph.D. degree from Harbin Institute of Technology, China, in 2007. He is currently a Professor of the school of Control Engineering in Shenyang Aerospace University. He has co-authored 3 books, over 60 journal and nearly 20 conference publications. His current research interests include fall detection and fault diagnosis.



Haifeng Se received the B.Sc. degree from Shenyang Aerospace University, China in 2018. From 2018 to 2020, he pursued the M.Sc. Degree in Shenyang Aerospace University. His current research interest is fall detection.

Jiahui Liu received the B.Sc. degree from Shenyang Aerospace University, China in 2019. Now, she is pursuing the M.Sc. Degree in Shenyang Aerospace University. Her current research interest is fall detection.