An Efficient Machine Learning-Based Fall Detection Algorithm using Local Binary Features

Majd SALEH
INSERM, U1099, Rennes, 35000, France
Université de Rennes 1, LTSI, Rennes, 35000, France
majd.saleh@univ-rennes1.fr

Régine LE BOUQUIN JEANNÈS
INSERM, U1099, Rennes, 35000, France
Université de Rennes 1, LTSI, Rennes, 35000, France
regine.le-bouquin-jeannes@univ-rennes1.fr

***Abstract*—According to the world health organization, millions of elderly suffer from falls every year. These falls are one of the major causes of death worldwide. As a rapid medical intervention would considerably decrease the serious consequences of such falls, automatic fall detection systems for elderly has become a necessity. In this paper, an efficient machine learning-based fall detection algorithm is proposed. Thanks to the proposed local binary features, this algorithm shows a high accuracy exceeding 99% when tested on a large dataset. In addition, it enjoys an attractive property that the computational cost of decision-making is independent from the complexity of the trained machine. Thus, the proposed algorithm overcomes a critical challenge of designing accurate yet low-cost solutions for wearable fall detectors. The aforementioned property enables implementing autonomous, low-power consumption wearable fall detectors.**

***Index Terms*—fall detection, binary features, local features, machine learning, elderly.**

I. Introduction

According to the world health organization, millions of elderly suffer from falls every year. Falls are one of the major causes of death worldwide especially for adults older than 65 years [1]. As a rapid medical intervention would considerably decrease the serious consequences of such falls, automatic fall detection systems for elderly has become a necessity. Fall detectors could be divided into ambient and wearable devices [2], [3]. Ambient fall detectors employ computer-vision, vibration, acoustic and pressure-based techniques, while wearable ones are generally based on Inertial Measurement Units (IMUs) including accelerometers, gyroscopes, and magnetometers.

Wearing a fall detector is convenient and advantageous since this type of device could accompany the elderly in both indoor and outdoor environments, supplying continuous fall detection, contrary to the ambient solutions. Note that only wearable fall detectors are considered in this paper.

Algorithms used in wearable fall detectors could be divided into two groups, threshold-based and machine learning-based algorithms. The threshold-based algorithms simply compare the quantified human activity captured using IMUs with predefined threshold(s) to decide the presence of a fall event, while machine learning-based ones extract some features from a sliding time window applied to the human activity signal and make a decision by passing these features to an already trained machine. When designing a wearable fall detector, the algorithm to be used should be not only embeddable but also of low computational cost for the sake of feasible power consumption. Therefore, low-cost threshold-based algorithms have been widely used in recent years, [4]–[7]. However, researchers have shown that the low computational cost of threshold-based algorithms is at the expense of accuracy [8]–[10]. To cope with this issue, machine learning-based algorithms have been proposed to achieve highly accurate fall detection. However, these algorithms are computationally demanding and embedding them in wearable devices could be impractical. The high computational cost of such algorithms comes from both features extraction and decision-making system. Regarding features extraction, a good summery could be found in [11] where a list of fifty features has been drawn up. Most of these features are much more computationally demanding than thresholding. In addition, making a decision by a trained machine is computationally demanding in general. For instance, when using artificial neural networks (ANN), the transfer function of each neuron, which is generally a nonlinear function in the hidden layers, is called whenever a decision is made by the ANN. In the same way, when using support vector machine (SVM), the minimal requirement (assuming a simple linear kernel) is to perform a number of multiplication and additions equal to the dimensionality of the features vector.

To overcome the aforementioned drawbacks, a novel fall detection algorithm is proposed in this paper. Whereas it employs the powerful machine learning techniques, its computational cost is comparable with the threshold-based techniques. As a matter of fact, the proposed algorithm enjoys an attractive property that the computational cost of decision-making is independent from the complexity of the trained machine as will be shown later in this paper.

This paper is organized as follows: Section II is dedicated to investigate the feasibility of using acceleration thresholds in human fall detection. Section III presents the proposed fall detection algorithm. The performance of this algorithm is evaluated in Section IV. Finally, Section V concludes the paper and discusses some future work.

II. Feasibility of using acceleration thresholds in human fall detection

In order to analyze the feasibility of using thresholds in fall detection, a large open dataset [12], which consists of about 4000 activity files simulating 34 different types of falls and activities of daily living (ADLs), is used. Each file consists in recordings of human triaxial acceleration, $\boldsymbol{a} = [a_x\ a_y\ a_z]$, for one particular type of ADL or fall. For each file, the maximum sum vector magnitude of acceleration, $\max ||\boldsymbol{a}||$, is calculated, where $||\boldsymbol{a}|| = \sqrt{a_x^2 + a_y^2 + a_z^2}$. Figure 1 shows the histogram of these maximal accelerations where the width of each bin is 0.1 g. As the numbers of files of ADLs and falls are different, the histograms are normalized is such a way that the sum of bins equals one for both ADLs and falls. For the sake of clarity, the scale of y-axis is logarithmic. This figure shows that there is an overlap between falls and ADLs' histograms in a wide range of accelerations ($[1.7, 7.5]$ g) which means that they could not be classified by means of their maximal accelerations. For instance, if a threshold of $T = 1.7$ g is used, all falls will be detected. However, 48% of ADLs satisfy this condition. On the other hand, if a threshold of $T = 4$ g is used, only 8% of ADLs will be misclassified. However, 29% of falls will not be detected. Similar results had been achieved for individual acceleration axis (x, y or z) and proved that using thresholds for fall detection is not reliable since they lead to poor sensitivity/specificity trade-off which explains the weak performance of threshold-based fall detection algorithms. Figure 2 shows the similarity in acceleration levels for particular types of ADL and fall. Particularly, the following sequence of ADLs is considered in Figure 2.a: sitting for a moment, trying to get up and collapse into a chair, while a backward fall caused by fainting or falling asleep, is considered in Figure 2.b. A number of 60 random samples of both the aforementioned events are illustrated.
However, thresholding could be quite useful when the time structure of falls is taken into account. This motivated us to describe human activities using local binary features based on acceleration thresholding within sliding time windows. These features are expected to discriminate between falls and ADLs which show similar acceleration levels. A detailed description of the aforementioned features as well as the proposed fall detection algorithm is explored in the next section.

III. The proposed fall detection algorithm

The activity of the elderly is captured using a triaxial accelerometer mounted to his/her waist as illustrated in Figure 3.a. Fall activities consist in three major phases, namely prefall, impact and postfall phases [13]. The triaxial acceleration signals during these phases are illustrated in Figure 3.b for a typical forward fall. The impact phase occurs when the body loses its balance and moves in a free fall manner for a short time until it hits the ground causing an impact shock in acceleration. After the impact, the body stays lying on the floor if the person is not able to recover.
The proposed method depends on the following observations,

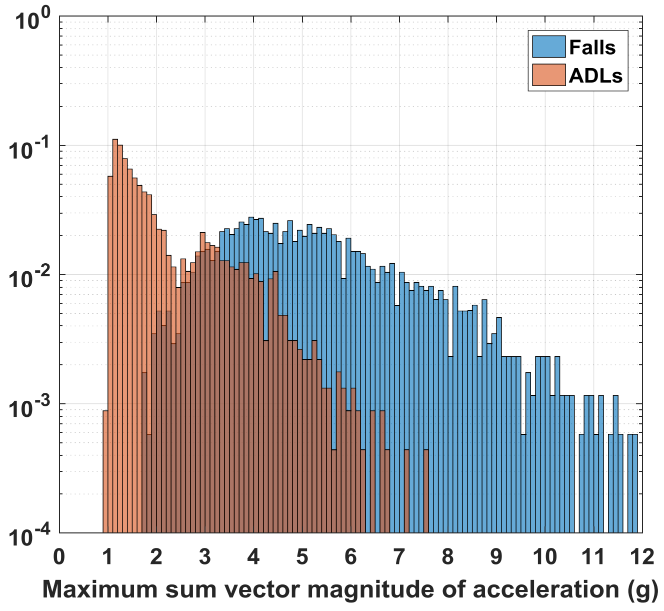

Fig. 1. Normalized histograms of maximum sum vector magnitude of accelerations calculated for 1723 activity files representing simulated falls (blue) and 2269 activity files representing ADLs (brown) from an open large dataset [12]

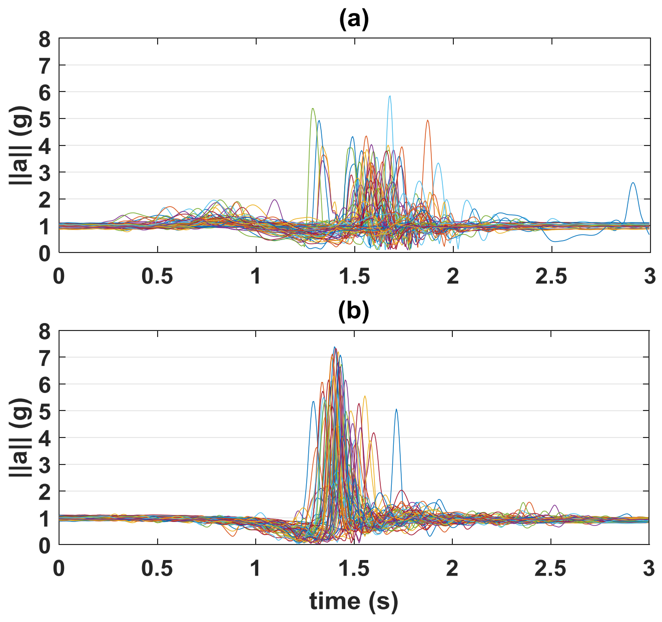

Fig. 2. Sum vector magnitude of acceleration $||\boldsymbol{a}||$ vs. time for: a) 60 ADL signals representing the following activity: sitting for a moment, trying to get up, and collapse into a chair, b) 60 fall signals representing the following fall type: fall backward while sitting, caused by fainting or falling asleep.

1) describing falls using local features is necessary to capture their time structure which is critical to discriminate between falls and ADLs that show similar acceleration levels 2) the postfall phase could be efficiently characterized using a_y acceleration as this acceleration component oscillates around -1 g in normal ADLs and it becomes very close to 0 g

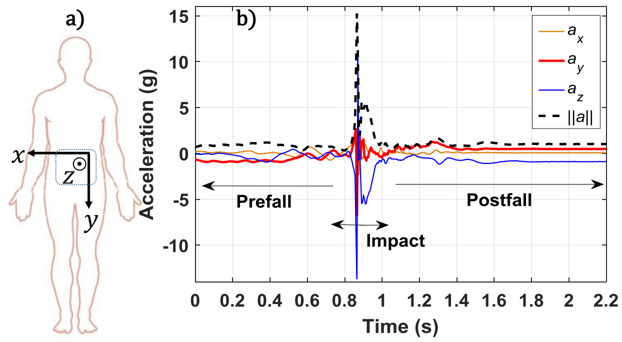

Fig. 3. a) Location of the fall detector and directions of acceleration components, b) The major phases of fall illustrated for a typical forward fall.

TABLE I
CONFIGURATIONS OF THE PROPOSED FEATURE EXTRACTION METHOD

Variable	w	f_s	f_e	T_1	T_2	$\boldsymbol{M}$
value	2.2	20	5	-0.5	1.7	**00111100000**
unit	s	Hz	Hz	g	g	-

when the body is lying and 3) the sum vector magnitude of acceleration, $||\boldsymbol{a}||$, is the best quantity to characterize the impact as it is invariant by changing the way of falling (forward, backward or lateral).

Features are extracted from a sliding window of width w, with feature extraction frequency $f_e = f_s/\kappa$ where f_s is the sampling frequency and $\kappa \in \boldsymbol{Z}^+$. Two binary features vectors, $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$, are built by thresholding a_y and $||\boldsymbol{a}||$, respectively. The dimensionality of $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$ is $N = w \times f_e$. One component of $\boldsymbol{B}_1$ is set to one if and only if **all** the corresponding κ successive samples of a_y exceed a threshold T_1, i.e. this feature reflects the continuity of the lying position, while one component of $\boldsymbol{B}_2$ is set to one if and only if **any** of the corresponding κ samples of $||\boldsymbol{a}||$ exceeds a threshold T_2, i.e. this feature reflects the presence of a potential impact. The final features vector $\boldsymbol{B}$ is built from $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$ as follows:

$$\boldsymbol{B} = (\boldsymbol{B}_1 \wedge \neg \boldsymbol{M}) \vee (\boldsymbol{B}_2 \wedge \boldsymbol{M}) \tag{1}$$

where $\wedge$, $\vee$ and $\neg$ denote the logical AND, OR and NOT operators, respectively and $\boldsymbol{M}$ represents a binary mask that localizes the impact phase. Figure 4 shows an example of building the features vector $\boldsymbol{B}$ where the practical values shown in Table I are used, $\kappa = f_s/f_e = 4$ and $N = w \times f_e = 11$. Table I will be discussed in Section IV.

In order to classify falls and ADLs, three machine learning-based methods were investigated that are 1) K-Nearest Neighbors (KNN), 2) Artificial Neural Networks (ANN) and 3) Support Vector Machine (SVM). Configurations of these methods will be stated in section IV. The strategy of the proposed classification method summarizes as follows:

1) Train the machine using a training set of falls and ADLs.
2) Generate all the 2^N potential features vectors.
3) Calculate the responses of the trained machine for all the potential features vectors and store them in a decision vector $\boldsymbol{D} \in \{0,1\}^{2^N}$.
4) Embed the decision vector $\boldsymbol{D}$ in the wearable fall detector (whatever the machine learning method is) and use the binary features vector $\boldsymbol{B}$ as an address to access the response (fall/ADL).

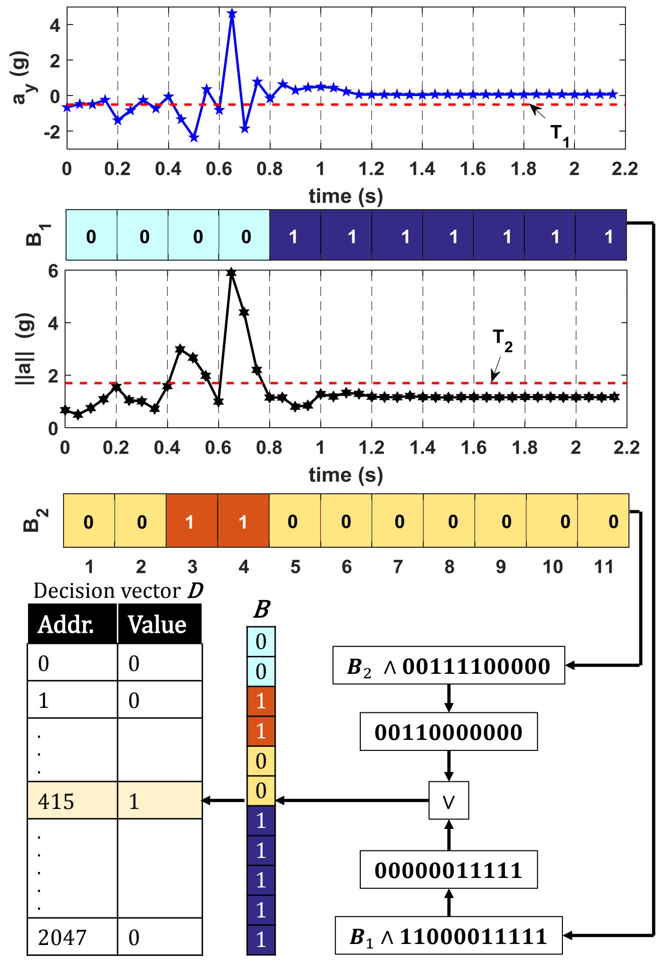

Fig. 4. An example of extracting binary features from an acceleration signal of a human activity (fall forward while jogging caused by a trip)

To show the feasibility of this strategy, it is well known that the flash memory of some micro-controllers can exceed 1 M Bytes today. This enables to use binary feature vectors with dimensionality $N = 20$. Now, the example shown in Figure 4 as well as the experimental analysis made in this study use only $11-$dimensional features vectors, which correspond to a decision vector of $2^{11} = 2048$ components.

A flowchart of the proposed fall detection algorithm is shown in Figure 5, where f_1 and f_2 are the new features that represent the last components in vectors $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$, respectively. From this flowchart, it is clear that the proposed binary features are extracted on-line and the machine learning-based response is nothing more than an address operation. Therefore, the computational cost of the proposed machine learning-based fall detection algorithm is low and practically comparable with the cost of threshold-based fall detection ones. More

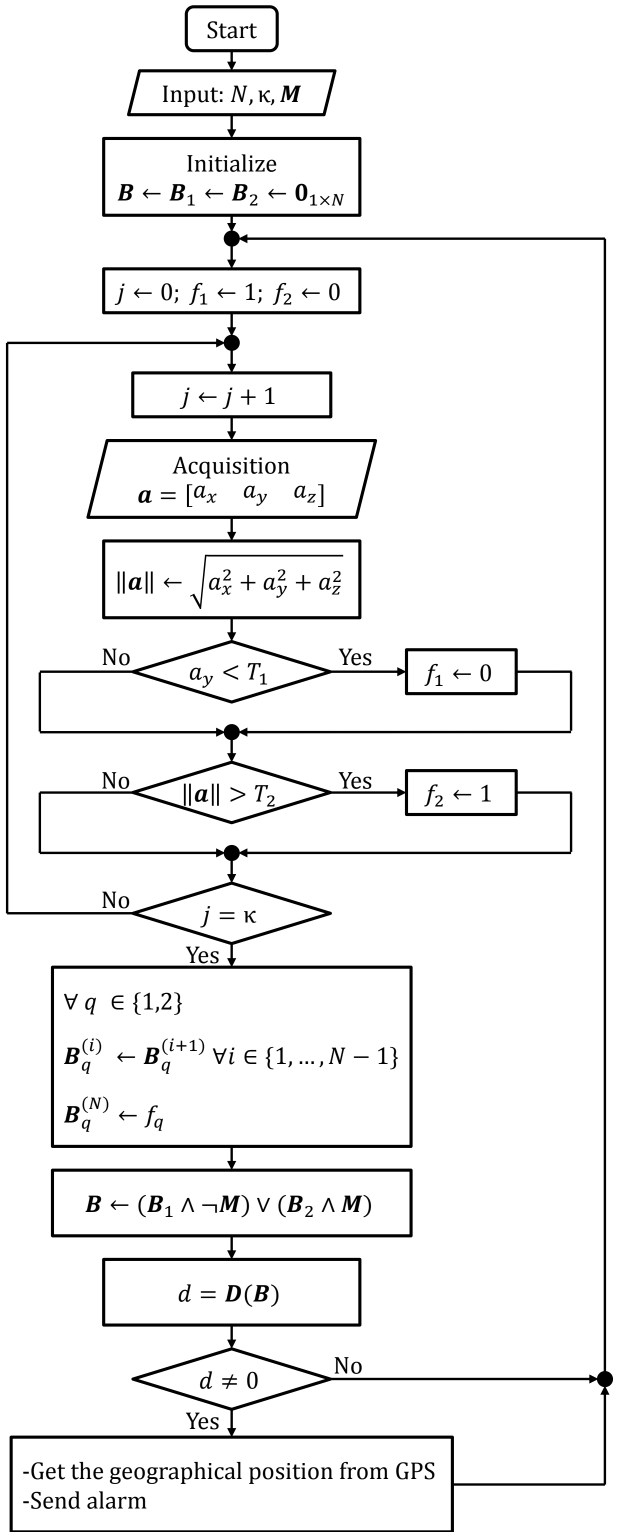
Start
Input: $N, \kappa, \boldsymbol{M}$
Initialize
$\boldsymbol{B} \leftarrow \boldsymbol{B}_1 \leftarrow \boldsymbol{B}_2 \leftarrow \boldsymbol{0}_{1\times N}$
$j \leftarrow 0;\ f_1 \leftarrow 1;\ f_2 \leftarrow 0$
$j \leftarrow j+1$
Acquisition
$\boldsymbol{a} = [a_x \quad a_y \quad a_z]$
$\|\boldsymbol{a}\| \leftarrow \sqrt{a_x^2 + a_y^2 + a_z^2}$
No
$a_y < T_1$
Yes
$f_1 \leftarrow 0$
No
$\|\boldsymbol{a}\| > T_2$
Yes
$f_2 \leftarrow 1$
No
$j = \kappa$
Yes
$\forall\, q \in \{1,2\}$
$\boldsymbol{B}_q^{(i)} \leftarrow \boldsymbol{B}_q^{(i+1)}\ \forall i \in \{1, \ldots, N-1\}$
$\boldsymbol{B}_q^{(N)} \leftarrow f_q$
$\boldsymbol{B} \leftarrow (\boldsymbol{B}_1 \wedge \neg\boldsymbol{M}) \vee (\boldsymbol{B}_2 \wedge \boldsymbol{M})$
$d = \boldsymbol{D}(\boldsymbol{B})$
$d \neq 0$
No
Yes
-Get the geographical position from GPS
-Send alarm

Fig. 5. Flowchart of the proposed fall detection algorithm

precisely, in terms of the required number of floating point operations per second (flops), the complexity of the proposed algorithm is given by $\boldsymbol{C}_1 = (7\kappa+3) \times f_e$, while the complexity of the threshold-based algorithms that threshold the sum vector magnitude of acceleration is given by $\boldsymbol{C}_2 = 6 \times f_s$. Numerically, for the configurations listed in table I, $\boldsymbol{C}_1 = 155$ flops and $\boldsymbol{C}_2 = 120$ flops.

In spite of its quite low computational cost, the proposed algorithm can even benefit from quite complicated machine learning-based methods as generating the responses is done off-line.

IV. Experimental results

In this section, the performance of the proposed fall detection algorithm is evaluated. Performance criteria are introduced first. Then, the dataset used in this evaluation is briefly described. Experimental configurations and results are then detailed and discussed.

Performance criteria: Three performance criteria, namely accuracy, sensitivity and specificity are used. They are given by:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (2)$$

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (3)$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \quad (4)$$

where $\text{TP}, \text{TN}, \text{FP}$ and FN denote true positives, true negatives, false positives and false negatives, respectively.

Dataset: The open Sisfall dataset [12] is used in this experimental analysis. It consists of 15 fall and 19 ADL types. Most ADL types were performed by 14 elderly participants. In addition, 23 young adults performed all ADL types and simulated all fall types. Participants cover both genders, a wide range of weights ($42 - 102$ kg) and a wide range of heights ($1.49 - 1.83$ m). Among different recorded activity signals, acceleration signals acquired using the ADXL345 accelerometer are used in this experimental analysis. The original data sampling frequency is 200 Hz. However, acceleration data are down-sampled at 20 Hz as the new frequency is more practical for two reasons: 1) it is sufficient to capture the activity of elderly; note that, even in the worst case, the duration of the impact phase ($300 - 500$ ms [13]) is at least 6 times greater than the new sampling period, i.e. 50 ms, and 2) it is more convenient from power consumption point of view as the latter is proportional to sampling frequency.

Experimental configurations: The configurations of the proposed fall detection algorithm are shown in Table I. The width of the sliding time window is set to $w = 2.2$ s as this duration is sufficient to capture the three main phases of a fall. Threshold T_1 is set to -0.5 g since the acceleration a_y oscillates around -1 g during normal ADLs and becomes very close to 0 g when the body is lying. In Section II, it has been experimentally shown that $||\boldsymbol{a}||$ always exceeds 1.7 g for falls. Therefore the threshold T_2 is set to this value. The binary mask $\boldsymbol{M} = 00111100000$ is set in such a way that the sequence "1111" corresponds with the impact phase of fall (with some margin) and $\neg\boldsymbol{M} = 11000011111$ localizes the prefall (sequence "11") and postfall (sequence "11111")

TABLE II
PERFORMANCE OF SEVERAL MACHINE LEARNING-BASED METHODS APPLIED TO THE PROPOSED BINARY FEATURES

Criteria $\rightarrow$ Method $\downarrow$	Accuracy (%)	Sensitivity (%)	Specificity (%)
KNN	**99.65**	97.66	**99.67**
ANN	99.30	97.72	99.32
SVM	99.40	**98.61**	99.41

phases respectively.
The tested KNN classifier is configured to do an exhaustive search using euclidean distance function with $k = 5$. The tested ANN is a feed-forward artificial neural network with one hidden layer consisting in 10 neurons, while the SVM is configured to use a Gaussian radial basis function (RBF) kernel with a standard deviation $\sigma = 2$.
Experimental results: In order to evaluate the performance of the proposed algorithm, two experiments were performed where the activity dataset was divided equally into two parts, one for training and one for testing, taking into account that activities performed by one person were considered either for training or for testing. For the sake of fairness, the set used for training in the first experiment is used for testing in the second one and vice-versa. The experimental results are averaged using both experiments. Table II shows the experimental results of the considered KNN, ANN, and SVM applied to the proposed local binary features. The aforementioned methods show similar performance. The high accuracy achieved by all the considered machine learning-based methods proves that the proposed features have an excellent discrimination power between falls and ADLs. KNN shows the best accuracy and specificity, while SVM shows the best sensitivity with a competitive specificity i.e. using SVM could be safer even if it is expected to generate slightly higher number of false alarms. Thanks for using binary features, the computational complexity of the proposed algorithm is independent from the complexity of the classifier as all the possible responses are calculated off-line and embedded in the fall detector. This property overcomes one of the major drawbacks of machine learning-based fall detection algorithms that is the high computational cost. For instance, in order to make a decision by a KNN classifier, the distance between the features vector and all training vectors should be measured, which demands high computational cost and makes embedding this classifier impractical. Interestingly, the proposed strategy simplifies decision-making to a simple address operation where the binary features vector is used as an address.

V. CONCLUSION AND FUTURE WORK

In this paper, an efficient machine learning-based algorithm was proposed for wearable fall detectors. It is based on describing the physical activity of the elderly using local binary features which are built using acceleration thresholding. A high accuracy fall detection exceeding 99% was achieved thanks to the high discrimination power of the proposed features. Moreover, the computational complexity of the proposed algorithm was shown to be quite low and comparable to the complexity of threshold-based algorithms. It has been shown that binary features vectors of higher dimensionality could be used in practice. Therefore, the performance could be further improved by investigating additional discriminative binary features in future work. Such features could be extracted form accelerations a_x, a_z or even from different sensors like gyroscopes where the angular velocity is to be thresholded.

ACKNOWLEDGMENT

This publication is supported by the European Union through the European Regional Development Fund (ERDF), the Ministry of Higher Education and Research, the French region of Brittany and Rennes Métropole.

REFERENCES

[1] WHO Media centre. Falls, fact sheet (reviewed january 2018) http://www.who.int/mediacentre/factsheets/fs344/en/, 2018.

[2] X. Yu. Approaches and principles of fall detection for elderly and patient. In *HealthCom*, Singapore, Singapore, Jul. 7-9 2008.

[3] K. Chaccour, R. Darazi, A. Hajjam El Hassani, and E. Andrès. From fall detection to fall prevention: A generic classification of fall-related systems. *IEEE Sensors Journal*, 17(3):812–822, 2017.

[4] S. Abdelhedi, R. Bourguiba, J. Mouine, and M. Baklouti. Development of a two-threshold-based fall detection algorithm for elderly health monitoring. In *IEEE, Research Challenges in Information Science (RCIS)*, Grenoble, France, Jun. 1 - 3 2016.

[5] J. Chen, K. Kwong, D. Chang, J. Luk, and R. Bajcsy. Wearable sensors for reliable fall detection. In *IEEE, Engineering in Medicine and Biology*, Shanghai, China, Sep. 1 - 4 2005.

[6] A. Kurniawan, A.R. Hermawan, and I.K.E. Purnama. A wearable device for fall detection elderly people using tri dimensional accelerometer. In *IEEE, International Seminar on Intelligent Technology and Its Applications (ISITIA)*, Lombok, Indonesia, Jul. 28 - 30 2016.

[7] F. Wu, H. Zhao, Y. Zhao, and H. Zhong. Development of a wearable-sensor-based fall detection system. *International Journal of Telemedicine and Applications*, vol. 2015, Article ID 576364, 2015.

[8] O. Aziz, M. Musngi, E.J. Park, G. Mori, and S.N. Robinovitch. 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 and Biological Engineering and Computing*, 55(1):45–55, 2017.

[9] L.P. Nguyen, M. Saleh, and R. Le Bouquin Jeannès. An efficient design of a machine learning-based elderly fall detector. In *HealthyIoT, the 4th EAI International Conference on IoT Technologies for HealthCare*, Angers, France, Oct. 24-25 2017.

[10] N. Noury, A. Fleury, P. Rumeau, A.K. Bourke, G.Ó. Laighin, V. Rialle, and J.E. Lundy. Fall detection - principles and methods. In *IEEE, Engineering in Medicine and Biology Society (EMBS)*, Lyon, France, Aug. 22 - 26 2007.

[11] N. Pannurat, S. Thiemjarus, and E. Nantajeewarawat. Automatic fall monitoring: A review. *Sensors (Basel)*, 14(7):12900–12936, 2014.

[12] A. Sucerquia, J.D. Lòpez, and J.F. Vargas-Bonilla. Sisfall: A fall and movement dataset. *Sensors Journal*, 17(1), 2017.

[13] N. Noury, P. Rumeau, A.K. Bourcke, G. Olaighin, and J.E. Lundy. A proposal for the classification and evaluation of fall detectors. *IRBM*, 29(6):340–349, 2008.