

A Smartphone-based Fall Detection System for the Elderly

Panagiotis Tsinganos¹ and Athanassios Skodras²
Department of Electrical and Computer Engineering
University of Patras
Patras, Greece
¹panagiotis.tsinganos@yahoo.gr
²skodras@upatras.gr

Abstract—Falls can be severe enough to cause disabilities especially to frail populations. Thus, prompt health care provision is essential to prevent and restore any harm. The purpose of this study is to develop a smartphone-based fall detection system that can distinguish between falls and activities of daily living (ADL). The typical fall detection system consists of a sensing component and a notification module. Android devices, equipped with sensors and communication services, are the best candidates for the development of such systems. This work incorporates a threshold based algorithm, whose accuracy is enhanced by a k Nearest Neighbor (kNN) classifier. In addition, this paper proposes the implementation of a personalization and power regulation system. It achieves high fall detection accuracy, (97.53% sensitivity and 94.89% specificity), which is comparable to related works.

Keywords—falls; ADLs; fall detection; smartphone; accelerometer; machine learning

I. INTRODUCTION

According to World Health Organization (WHO) “a fall is defined as an event which results in a person coming to rest inadvertently on a lower level” [1]. The reasons why falls occur and some people are more likely to fall may be intrinsic or extrinsic. In the first case, chronic problems and diseases such as arthritis and visual impairment can increase the incidence of falls. On the other hand, environmental hazards (e.g. slippery surfaces) and dangerous activities may cause a fall. Regarding the effects, they vary from severe physical injuries to psychological issues, depression and avoidance of the activity that caused the fall.

The current fall detection system implementation [2] is based on both threshold and machine learning techniques, while it consists of four parts. The detection component collects data from the accelerometer which are preprocessed using a series of magnitude and time thresholds as implemented in [3] to identify fall-like segments. A second part extracts features from these segments that are used by a k Nearest Neighbor (kNN) classifier to distinguish between falls and activities of daily living (ADL). There are also two auxiliary components; one that monitors the activation level of the sensor and aims keeping battery consumption as low as

possible and another that adjusts the model to the user activity patterns.

An advantage of accelerometer sensors is that the generated signal in case of a fall event is highly distinguishable, as shown in Fig.1. This can be decomposed into a series of stages which can be identified during the detection process [4] using a Finite State Machine (FSM):

- The pre-fall period, characterized by conventional ADLs containing some signs of instability.
- The free-fall phase, during which the human body experiences a temporal weightless state provoked by a hasty movement toward the ground, generating values near to 0g.
- The impact phase, characterized by a vertical shock when the body hits the ground. In some cases, the initial hit can be followed by a series of minor impacts.
- A post-fall phase, in which the body lies on the ground and a remarkable change in body’s orientation is recognized.
- A recovery phase during which the patient may remain still motionless if he/she is unconscious or severely injured after the collapse.

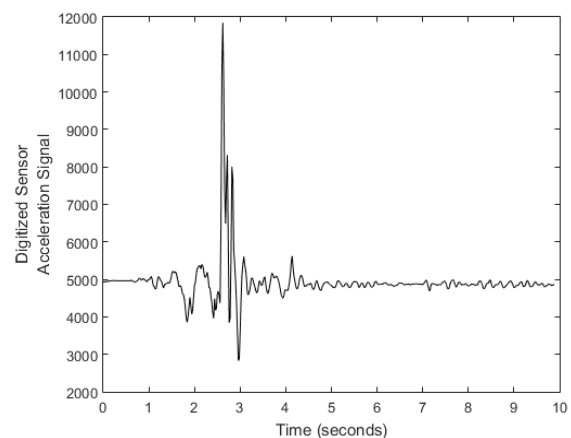


Fig.1 Wearable sensor acceleration signal of falling event

Table 1 Related fall detection works. The abbreviations for the sensors are as follows: Accelerometer (Acc.), Gyroscope (Gyr.), Orientation (Ori.) and Magnetometer (Mag.)

Ref	Year	Position	Sensors	Sampling(Hz)
[3]	2012	Belt	Acc.	51.2
[5]	2010	Shirt, Belt, Pants	Acc.	
[6]	2011	Waist	Acc.	50
[7]	2013	Chest	Acc.,Gyr.	
[8]	2011		Acc., Ori..	24.45
[9]	2016	Pants, Belt	Accel., Gyr., Ori.	50
[10]	2014	Head, Chest, Waisy, Wrist, Thigh, Ankle	Acc., Gyr., Mag.	25

These phases are not present in all types of falls or may appear more than once. For example, when falling down the stairs the recorded signal may be characterized by the presence of successive free fall and impact phases.

Detecting these phases in an acceleration signal can help distinguish fall-like events from common daily activities. However, to further improve the detection rate, machine learning techniques can be used. During this step a set of features is extracted to retain information relevant to the acquired data. Then, a classifier takes advantage of these characteristics to construct a mathematical model that distinguishes between two or more classes.

The rest of this paper is organized as follows: In Section II related work in the field of fall detection is presented. Section III gives a description of the components of the proposed fall detection system, while Sections IV and V present the results and the conclusions respectively.

II. RELATED WORK

Both academic research and industry have been interested in activity recognition over the last decades. Fall detection, a subset of the former, has been studied and an array of systems has been developed with varying degrees of accuracy. Some use only wearable inertial sensors, while others have ambient sensors be integrated in the living environment. Of particular interest are the smartphone-based implementations. Table 1 displays the most representative works in the literature.

The authors of [5] and [6] use thresholds for the magnitude of the acceleration and the acceleration at the vertical direction. When both thresholds are exceeded within a time window, a fall is detected. The system developed in [6] uses angular velocities in addition to acceleration magnitude. When both values are greater than the specified thresholds a fall is detected. Unlike the previous implementations the phone is attached to middle chest. In [8] they use both upper and lower thresholds for the acceleration magnitude. If they are both exceeded in less than 1s, the acceleration signal is compared with a fall template using wavelet transform. Then a fall is detected if the transformation yields high similarity. The authors of [9] evaluate the capacity of smartphone sensors to differentiate fall events from ADLs and they propose a low power consumption threshold-based algorithm. Although most works depend on thresholds, recent approaches [10] employ feature extraction and machine learning classifiers to improve detection accuracy.

These studies differ in many aspects of their implementation. For example, [3] and [8] employ both thresholds and machine learning techniques, whereas the rest are comprised of threshold based algorithms only. Moreover, various filtering methods are used to smooth the noisy accelerometer data, such as first and second order low-pass filters [9].

One of the problems in assessing a fall detection algorithm is the lack of an approved database of realistic falls and ADLs. Actually, every published study selects a different set of activity patterns on which they evaluate the detection algorithm. The most common activities performed are walking, running, standing up and sitting down, while forward, backward and lateral fall types are simulated. However, in most cases, these data are not made. Therefore, not only is a reasonable comparison almost impossible, but the results are not reproducible as well.

III. IMPLEMENTATION

A. Overview

The application is comprised of two major components; the detection system and the notification system. The detection component continuously collects acceleration data sampled at 50Hz, applies the fall detection algorithm, and broadcasts a fall event when a fall is identified by the classifier. Then, the notification system sends a notification (email or SMS) containing the location of the user to predefined contacts (defined in the Settings menu).

These elements were implemented using the Java programming language (Fig.2). In particular, the Android Studio IDE was used for the communication of the various

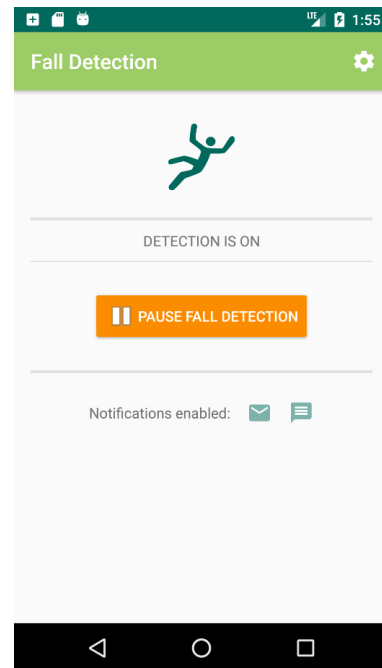


Fig.2 Home screen of Fall Detection app

objects while from the WEKA library for Java the kNN implementation was imported.

B. Data-Preprocessing

Accelerometer data are collected with a rate of 50 samples per second. This sampling frequency is regularly used in the literature and satisfies the Nyquist theorem, since body movements contain frequency components mainly in the range of 0Hz to 15Hz ([11], [12]). To account for sensor errors two filtering techniques have been applied to the triaxial data (acceleration vector). Firstly, an offset value is used to correct the collected measurements at each axis. At a next stage, a 60ms time window with $F_s=50\text{Hz}$ is used to apply a median low-pass filter. The major advantage of this filter is that narrow spikes are removed while abrupt changes caused by body motion are preserved.

C. FSM

Once a new accelerometer sample is collected, the triaxial data along with the magnitude and time are forwarded to the detection algorithm.

The first part of the detection algorithm can be modeled as a Finite State Machine (FSM) that uses a series of thresholds both on the acceleration magnitude and the time to eliminate false positives. It consists of five states as shown in Fig.3 and is described extensively in [3].

D. Features

Then, a set of 14 features is extracted from fall-like segments that are used by a kNN classifier to distinguish between a 'FALL' and an 'ADL'. They are taken from the acceleration magnitude data collected in the interval [-2500, 1000]ms, centered at the instant of maximum peak above the threshold. Of the selected features, seven are derived from the time domain (AAMV, IDI, MPI, PDI, ARI, FFI, SCI) and are fully explained in [3], five are statistical measures (SKEW, KURT, IQR, POWER_IMP, STD_IMP) and two are extracted from the wavelet transform as described in Table 2.

The time domain features capture the main characteristics of the fall acceleration pattern, such as the abrupt transition from low values to the peak, the impact and free fall durations. On the other hand, statistical features contain information about the distribution of the acceleration values of the fall pattern. A negative skewness (SKEW) means that the majority of the samples has high values, whereas the higher the kurtosis (KURT) value the more the outliers. Considering the fall pattern in Fig.1, we expect a positive skewness and a high kurtosis value. Two metrics of variability (IQR and STD_IMP) have been used since the distribution of the fall pattern is not normal in general. Finally, the power of the impact segment (POWER_IMP) indicates the magnitude of the impact and can be used to distinguish between a fall and sitting on a chair.

Two features are related to the Continuous Wavelet Transform (CWT). These were extracted from the CWT of the acceleration magnitude using a custom-built wavelet. This wavelet was generated from a 2s segment of the median signal of all the fall patterns in the dataset aligned at their peak value.

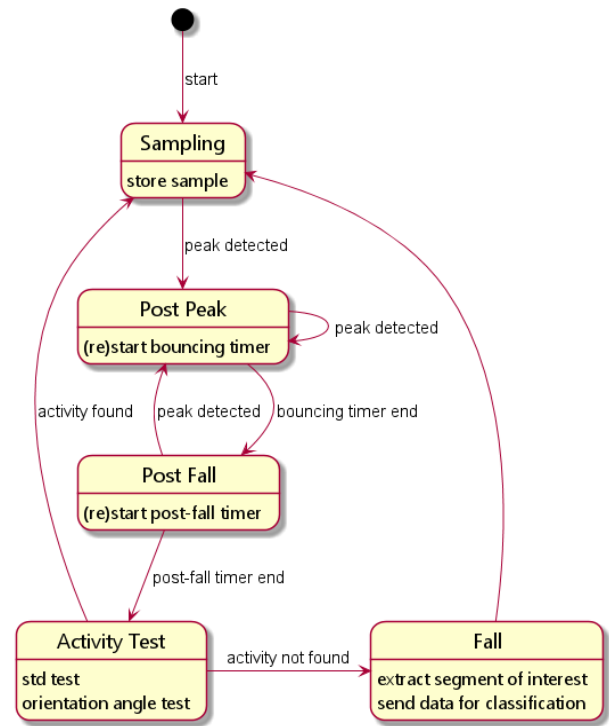


Fig.3 Fall detection FSM

The resulting waveform was then adapted to satisfy the wavelet definition using the MATLAB Wavelet Toolbox (Fig.4). This technique has been previously used in [13] and [14]. The output of the wavelet transform is a two-dimensional signal with columns that correspond to time and rows to scale. To account for border effects, i.e. high values at the borders of the CWT, 10% of the signal is removed from the left and right borders.

Using the CWT of the acceleration patterns allows the localization of the transient state of the signal during the impact. Thus, a fall incidence can be detected if a maximal surface is searched. On the contrary, an ADL will not generate a distinctive maximum, since these activities usually generate repetitive patterns (Fig.5).

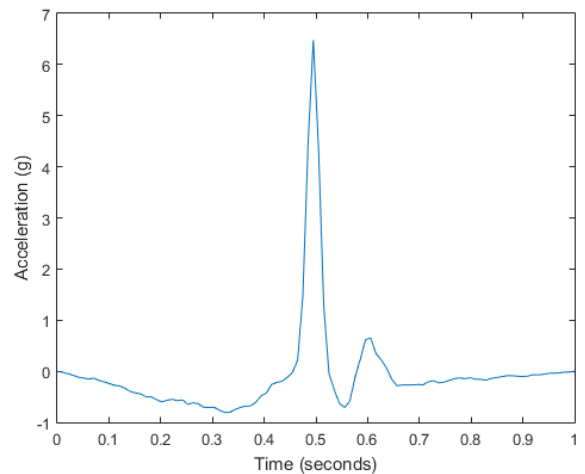


Fig.4 The adapted wavelet

Table 2 Extracted features description

Name	Domain
AAMV (average absolute acceleration variation)	time
IDI (impact duration)	time
MPI (highest value of acceleration)	time
PDI (peak duration)	time
ARI (activity level of a window that contains the impact)	time
FFI (average acceleration of free-fall stage)	time
SCI (number of steps)	time
SKEW (skewness of acceleration segment)	time (statistical)
KURT (kurtosis of acceleration segment)	time (statistical)
IQR (interquartile range of acceleration segment)	time (statistical)
POWER_IMP (power of the impact)	time (statistical)
STD_IMP (standard variation of the impact)	time (statistical)
CWENERGY (square of the highest coefficient of cwt)	time-frequency
CWPEAKS (number of peaks in the cwt)	time-frequency

E. kNN Classifier

Regarding the kNN classifier the following parameter values provided higher accuracy. The k parameter has been set to 7, while the Manhattan distance function has been used. Additionally, a distance based voting has been used instead of the simple majority voting to account for the skewed distribution of the dataset. These parameter values were selected after a cross validation procedure in order to achieve high accuracy. To account for the fact that kNN is sensitive to noise and to reduce computation time, an edition filter has been used. Before the classifier is trained we apply the Edited Nearest Neighbor (ENN) rule, as described in [15]. This algorithm starts with the whole dataset and each instance is removed if it does not agree with the majority class of its

nearest neighbors.

F. Personalization

In an attempt to reduce the false positives, i.e. the misclassification of ADLs, we propose that the features of these activities be appended to the training dataset. To achieve this when an activity is classified as fall by the classifier the user is given a configurable amount of time to signal a false positive. In that case, the features tuple of the activity is stored into the memory of the phone along with the rest dataset. As a result new information regarding the activity patterns of the user is appended to the model.

G. Power consumption

The importance of monitoring power consumption has been previously stressed ([4], [9], [16]). The power requirements for a continuously monitoring application are increased, since sensors are sampled at regular intervals when the activity is in the foreground and the screen is on or a background service keeps the CPU running.

The proposed battery-drain reduction algorithm adjusts the sampling frequency according to the level of activation of the sensor. Considering that most of the time an older person does not perform any action that significantly changes the measured values (inactivity state), any deviation from this state indicates heightened activity, thus higher fall probability. The values of the normal state are modeled using a Gaussian probability density function $f(x) \sim N(\mu, \sigma^2)$, where μ and σ^2 are the mean and the variance, respectively. When a new sensor value arrives the mean and the variance are updated using the following functions:

$$\mu' = v/(v+1) \cdot \mu + \alpha_{v+1}/(v+1) \quad (1)$$

$$\sigma'^2 = (v-1) \cdot \sigma^2/v + \mu^2 + \alpha_{v+1}^2/v - (v+1) \cdot \mu'^2/v \quad (2)$$

As the range of normal values, we have used the interval $x \in [\mu - 2\sigma, \mu + 2\sigma]$. In normal state the sampling frequency is set to 10Hz and when significant activity is detected it is increased to 50Hz. In Fig.6, it can be seen that when the user stood still the sensor values were characterized as normal and

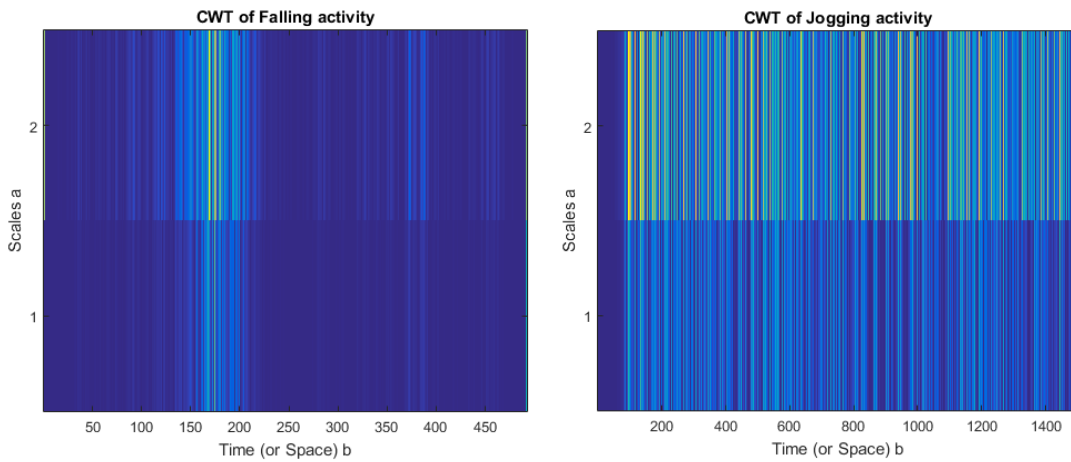


Fig.5 Continuous Wavelet Transform of a fall and an ADL

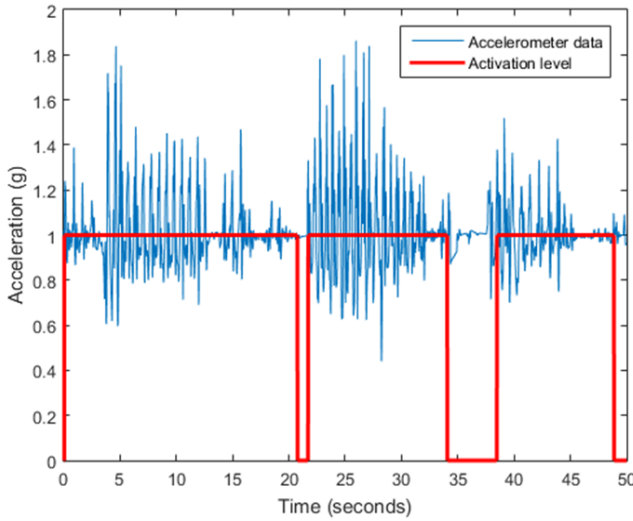


Fig.6 Accelerometer activation level during intermittent walking when continued walking the sensor was considered activated.

IV. RESULTS

In the current work, four different classifier models are being compared before the final model is implemented. Since our dataset is imbalanced (more feature tuples from ‘FALL’ class), the accuracy metric is not an appropriate performance measure [17]. Therefore, the evaluation of these classifiers is based on the area under the ROC curve (AUC) metric. Model statistics are generated from 10-fold cross-validation and a pairwise comparison is performed with a t-test. This allows examining whether the difference in the performance of the candidate models is significant, thus one classifier is preferred over the others. Once the classifier is chosen, the total performance of the system is reported in terms of accuracy, specificity (SP) and sensitivity (SE) in order to be compared with other works in the literature.

Between the most common classifiers, meaning Artificial Neural Network (ANN), Support Vector Machine (SVM), Decision trees and kNN, the latter performed better based on the MobiFall dataset [18]. Specifically, for each model a set of parameters was chosen with a grid search such that the AUC metric was maximized. As a consequence the following models were compared:

- ANN was a multilayered perceptron with one hidden layer of 7 nodes, momentum 0.7 and learning rate 0.1
- SVM with RBF kernel and gamma set to 0.15
- kNN of 7 neighbors, Manhattan distance and inverse

Table 3 Paired t-test comparisons. The first column represents the base classifier in each test. Parentheses contain the AUC value. A “v” denotes that the classifier outperforms the base model, while “*” denotes that the base model is better.

	<i>kNN</i> (1.00)	<i>ANN</i> (0.99)	<i>SVM</i> (0.95)	<i>J48</i> (0.96)
<i>kNN</i>	-	*	*	*
<i>ANN</i>	v	-	*	*
<i>SVM</i>	v	v	-	
<i>J48</i>	v	v		-

Table 4 Confusion matrices

Actual	Predicted		Actual	Predicted	
	Fall	ADL		Fall	ADL
Fall	618	29	Fall	631	16
ADL	55	1824	ADL	96	1783

a) Cross validation

b) Final model

distance weighting

- J48 decision tree with pruning set to 0.2

Performing a cross-validation we can see that kNN scores the highest performance (Table 3). A further comparison reveals that this model is better regarding not only the AUC performance, but other metrics as well. The high AUC value in the 10-fold cross validation indicates that the algorithm has good generalization properties. Consequently, the KNN algorithm was chosen based on its overall performance on the dataset.

The MobiAct [18] dataset contains 647 fall patterns and 1879 patterns of daily activities. This set of activities was separated into train and test datasets using a 10-fold cross validation in order to evaluate the entire classifier model. The results of this evaluation are shown in Table 4a. Then some common evaluation metrics can be computed as follows:

- sensitivity=recall=618/647=95.52%
- specificity=1824/1879=97.07%
- precision=618/673=91.83%

The final model is built using all the activities in the dataset. Of these, the FSM module with the chosen parameters detects 932 possible falls, 637 of which are real falls. This reduced dataset is used to build the classifier model. Using the confusion matrices that resulted from each of the two components the overall performance of the algorithm can be calculated (Table 4b). To this effect, the kNN classifier is trained from the reduced dataset after the ENN rule is applied with k=3, whereas it is evaluated using the whole dataset. Employing the above equations results to sensitivity and specificity equal to 97.53% and 94.89% respectively.

To further assess the proposed algorithm, a comparison with other similar studies of Table 1 is carried out (Table 5). Since the dataset and the implemented algorithm of each model was not available, the comparison is based on the reported values of SE and SP metrics. The recorded results from our implementation are comparable to other algorithms in the literature (Table 5). As we may see, the proposed model performs better than the threshold based algorithms implemented on smartphones. This means that a machine learning classifier definitely improves the detection performance of a threshold based algorithm (e.g. [6] and [7]). This advantage is considerable when the smartphone is fixed

Table 5 Comparison of algorithms based on SE and SP

	<i>Abbate S.</i> [3]	<i>Dai J.</i> [5]	<i>Lee R.</i> [6]	<i>Yang BS.</i> [7]	<i>Tsinganos P.</i> [2]
<i>SE</i>	100%	97.33%	77 %	95.24%	97.53%
<i>SP</i>	100%	91.30%	81%	94.25%	94.89%

Table 6 Battery consumption with the power management system enabled and disabled

	ON	OFF
ASUS Zenfone	0.01% (0.03mAh)	0.66% (16.45mAh)
LG D160	0.36% (6.19mAh)	0.92% (15.67mAh)

on the body (e.g. [2]).

Regarding the effect of the proposed personalization technique, we can estimate the performance improvement by adding the false positives to the training dataset one by one and building the classifier model. Specifically, using the data we collected with an ASUS Zenfone 2 smartphone, we classify each instance and if an ADL is classified as a fall, then this tuple is appended to the training dataset. Next, the whole dataset is used to test whether the addition of a tuple has improved the performance. An improvement in the AUC metric of the classifier was recorded (after seven false positives were appended, it was increased by 10%).

It has been stated that how active the user is at a given time can be used to change the sampling frequency and release system resources (e.g. a wake lock). Indeed, when this technique is applied battery life is preserved since the CPU usage is reduced. To quantify the reduction in the required power the Android application GSAM Battery Monitor was installed from the Google Play into two smartphones; a LG D160 and an ASUS Zenfone 2. The results are summarized in Table 6.

There is a difference between the two smartphones that affects how our application drains the battery. On the ASUS smartphone less than 0.1% of the battery was consumed with power management enabled, whereas on the LG there was a 0.364% drain. Similar values were recorded for the CPU times as well. What is clear, however, is that when the power management system is enabled the battery consumption is drastically reduced regardless of the smartphone.

V. CONCLUSIONS

This study developed a fall detection system based on an Android smartphone device. Identifying signal patterns and extracting parameters that can be used by a classifier can be applied to distinguish falls from daily activities. This solution is based on the work of Abbate S. [3] and incorporates a personalization and power management method.

In a future work we would like to collect accelerometer data from a wider range of population and test the algorithm in a real world environment. Another aspect of the fall detection system that we would like to study further is how the algorithm is affected by user context. For example, a location and time aware application would activate and deactivate the fall detection monitoring as needed. Finally, improving the personalization component is of primary importance since it is responsible for adjusting the classifier to the user activity patterns and its performance.

ACKNOWLEDGMENT

We would like to thank Professor Manolis Tsiknakis (Department of Informatics Engineering, TEI Crete) for

providing the MobiAct [18] dataset. We also thank the authors of [3] and especially Guglielmo Cola (Pervasive Computing & Networking Lab, University of Pisa) for providing clarifications on their work.

REFERENCES

- [1] "Falls | WHO," WHO, 2016. [Online]. Available: <http://www.who.int/mediacentre/factsheets/fs344/en/> [Accessed: 18-Nov-2016].
- [2] P. Tsinganos, "Smartphone-based Fall Detection System for the Elderly", MSc Thesis, BME Course, Univ of Patras, Patras, Jun. 2017
- [3] S. Abbate, M. Avvenuti, F. Bonatesta, G. Cola, P. Corsini, and A. Vecchio, "A smartphone-based fall detection system", *Pervasive and Mobile Computing*, vol. 8, no. 6, pp. 883–899, Dec. 2012.
- [4] E. Casilari, R. Luque, and M.-J. Morón, "Analysis of Android Device-Based Solutions for Fall Detection", *Sensors*, vol. 15, no. 8, pp. 17827–17894, Jul. 2015.
- [5] J. Dai, X. Bai, Z. Yang, Z. Shen, D. Xuan, "PerFallD: A pervasive fall detection system using mobile phones", *Proceedings of the 8th IEEE International Conference on Pervasive Computing and Communications (PERCOM 2010)*, Mannheim (Germany), 29 Mar.-2 Apr. 2010
- [6] R. Y. W. Lee and A. J. Carlisle, "Detection of falls using accelerometers and mobile phone technology", *Age and Ageing*, vol. 40, no. 6, pp. 690–696, 2011.
- [7] B.-S. Yang, Y.-T. Lee, and C.-W. Lin, "On Developing a Real-Time Fall Detecting and Protecting System Using Mobile Device", *Proceedings of the International Conference on Fall Prevention and Protection*, pp. 151–156, 2013.
- [8] V. Viet and D. Choi, "Fall Detection with Smart Phone Sensor", *The 3rd International Conference on Internet (ICONI) 2010*, pp. 27–31, Dec. 2011.
- [9] I. N. Figueiredo, C. Leal, L. Pinto, J. Bolito, and A. Lemos, "Exploring smartphone sensors for fall detection", *The Journal of Mobile User Experience*, vol. 5, no. 2, 2016.
- [10] A. Özdemir and B. Barshan, "Detecting Falls with Wearable Sensors Using Machine Learning Techniques," *Sensors*, vol. 14, no. 6, Jun. 2014, pp. 10691–10708.
- [11] I. Cleland et al., "Optimal placement of accelerometers for the detection of everyday activities", *Sensors (Basel, Switzerland)*, vol. 13, no. 7, pp. 9183–9200, 2013
- [12] M. Shoaib, S. Bosch, O. Durmaz Incel, H. Scholten, and P. J. M. Havinga, "A Survey of Online Activity Recognition Using Mobile Phones", *Sensors*, vol. 15, pp. 2059–2085, 2015.
- [13] A. Fleury, N. Noury, and M. Vacher, "A wavelet-based pattern recognition algorithm to classify postural transitions in humans", *European Signal Processing Conference (Eusipco)*, pp. 2047–2051, 2009.
- [14] L. Palmerini, F. Bagalà, A. Zanetti, J. Klenk, C. Becker, and A. Cappello, "A Wavelet-Based Approach to Fall Detection", *Sensors*, vol. 15, no. 5, pp. 11575–11586, 2015.
- [15] S. García, J. Derrac, J. R. Cano, and F. Herrera, "Prototype Selection for Nearest Neighbor Classification: Survey of Methods", *Sci2S.Ugr.Es*, pp. 1–28, 2010.
- [16] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "Energy Efficient Smartphone-Based Activity Recognition using Fixed-Point Arithmetic", *Journal of Universal Computer Science*, vol. 19, no. 9, pp. 1295–1314, 2013.
- [17] J. Han, M. Kamber, and J. Pei, *Data Mining Concepts and Techniques*, 3rd ed. MK, 2012.
- [18] G. Vavoulas, C. Chatzaki, T. Malliotakis, M. Padiaditis, and M. Tsiknakis, "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 (ICT4AWE)*, vol. 1, pp. 143-151, 2016.