Fall detection based on local peaks and Machine Learning

José R. Villar[1], Mario Villar[2], Mirko Fañez[3], Enrique de la Cal[1], and Javier Sedano[3]

[1] University of Oviedo, Spain
{villarjose,delacal}@uniovi.es
[2] University of Granada, Spain
mario.villarsanz@gmail.com
[3] Instituto Tecnológico de Castilla y León, Spain
{mirko.fanez,javier.sedano}@itcl.es

Abstract. This research focuses on Fall Detection (FD) using on-wrist wearable devices including tri-axial accelerometers performing FD autonomously. This type of approaches makes use of an event detection stage followed by some pre-processing and a final classification stage. The event detection stage is basically performed using thresholds or a combination of thresholds and finite state machines. In this research, we extend our previous work and propose an event detection method free of thresholds to tune or adapt to the user that reduces the number of false alarms; we also consider a mixture between the two approaches. Additionally, a set of features is proposed as an alternative to those used in previous research. The classification of the samples is performed using a Deep Learning Neural Network and the experimentation performs a comparison of this research to a published and well-known technique using the UMA Fall, one of the publicly available simulated fall detection data sets. Results show the improvements in the event detection using the new proposals.

1 Introduction

The study of Fall Detection (FD) represents a challenge in many different domains, from the monitoring of patients to the improvement of the autonomous living of elderly people. There are many different solutions, including video systems [1], intelligent tiles [2], sound detection [3], etc. Wearables play an important role in FD as they can be easily deployed, allowing care institutions to optimize their services with a relatively low cost [4,5]. More specifically, autonomous on-wrist wearable devices may be crucial in helping the elder population to continue living by their own. With autonomous on-wrist wearable devices we are referring to smart devices, such as smart-watches, that can be extended to detect any possible fall event using their own computational capabilities, without the assessment of any external service. In this research, we focus on smart-watches with built-in tri-axial acceleromenters (3DACC), which is by far the most chosen option [6,7,8,9,10]. Reviews on FD can be found in [11,12].

The main part of the solutions make use of Machine Learning (ML) to classify the current instance as a fall. Some examples of these methods are those presented in [8,6], where a feature extraction stage and Support Vector Machines classifies the Time Series (TS) windows. However, thresholds have been also used in FD [9,13,14], labelling the instance according to whether or not the magnitude of the acceleration surpass the pre-defined values. Thresholds have also been used in FD to define simple rules that drive the final decision [9,10,15]. Refer to [7] for a comparison of these type of methods.

Currently, it could be said that there are two main research approaches: using deep learning solutions and using classical ML solutions. Concerning the former, there are several published studies [16,17], but the capacity of current wearable devices is still far from that desired to include this type of models [18]. This study focuses on the second type of solutions; more specifically, in those studies concerned with the dynamics in a fall event [19,20]. These studies includes a FD method, a pre-processing stage and a classification stage using an ML method. For instance, Abbate et al. proposed the use of these dynamics as the basis of the FD algorithm [19], with moderate computational constraints but a high number of thresholds to tune. The proposal of Abbate et al has been modified in a series of papers [21,22,23] to adapt the sensor placement on a wrist. We refer to this event detection as *on-wrist Abbate*. Recently, local peak detection was proposed to identify the fall events together with a different set of transformations of the acceleration magnitude [24], which represents the starting point of this research.

The main contribution in this study consist of introducing a Finite State Machine (FSM) to the event detection mechanism proposed in [24]. Interestingly, this new event detection makes use of no user predefined threshold, which represents a step ahead in the event detection mechanisms in the literature. We refer to this event detection mechanism as *MAX-PEAK-FSM*. Alternatively, we evaluate a mixture of both approaches, denoted as *ABBATE-MAX-PEAK*. Furthermore, several new features are computed for the acceleration data window surrounding each detected peak; a study on this topic is performed. Finally, a comparison with different solutions in the literature are presented.

The structure of the paper is as follows. The next section deals with the description of both the MAX-PEAK and the new proposal MAX-PEAK-FSM together with the ABBATE-MAX-PEAK. The feature description and how these features are processed is included in this section as well. Section 3 details the experimentation set up and includes the results and discussion. Finally, the conclusions are drawn.

2 Fall detection using local maximum peaks

Fig. 1 shows the block diagram proposed to detect the fall events. Firstly, the magnitude of the acceleration is calculated. The first stage is the event detection method, which signals whether a TS sample includes a candidate of a fall event. The second stage is the feature extraction from a peak window surrounding the found peak. Finally, the set of features is classified either as a Fall or as a

Not_Fall peak. This classification stage is performed with a feed-forward Neural Network (NN) following the studies in [19,22,24]. This section deals with the event detection which is described in the next subsection, while the FSM is described in subsection 2.2.

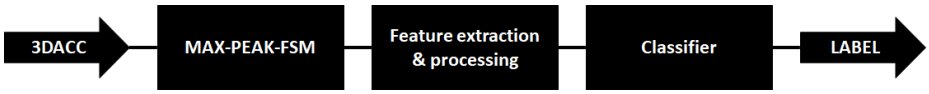

Fig. 1: Block diagram of the proposal. The local maximum peaks are filtered and only those found relevant are anylized. The feature extraction aims to represent the most interesting characteristics of the TS surrounding the peak. Finally, a feed-forward NN classifies the instance.

2.1 Event detection using local maximum peaks

The block diagram of the event detection stage is presented in Fig. 2. The values within the Time Series (TS) of the magnitude of the acceleration are smoothed using the mean value within sliding window of size $\frac{1}{4}$ of second and shift 1 sample. From now on, the TS contains the mean values computed from the smooth step.

Afterwards, we apply the S_1 transformation proposed in [25]. S_1 is calculated using Eq. 1, where k is the predefined number of samples and t is the current sample time-stamp. It is worth noticing that, although we analyze the window $[a_{t-2k-1}, a_t]$ at time t, the peak candidate is a_{t-k}, the center of the interval. The S_1 transformation represents a scaling of the TS, which makes the peak detection easier using a predefined threshold α. The algorithm for detecting peaks is straightforward: a peak occurs in time t if the value S_t is higher than α and is the highest in its $2k$ neighborhood. The value of k is determined as the inverse of the sampling frequency.

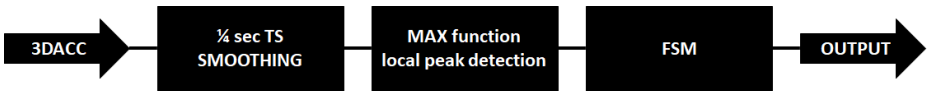

Fig. 2: Event detection mechanism. The 3DACC signal is smoothed using a $\frac{1}{4}$ second window. The filtered signal is analyzed using the local maximum peak proposed in [25]. When a local maximum peak is detected the FSM filters those repetitive peaks that appears in certain activities, such as walking or running. The output is whether a peak might be a fall event candidate (thus, needing further processing) or not.

$$S_1(t) = \frac{1}{2} \times \{ \max_{i=t-2k}^{t-k-1} (a_{i+1} - a_i) + \max_{i=t-k+1}^{t} (a_i - a_{i-1}) \} \tag{1}$$

As stated in [24], using statistics theory avoids the problem of the defining α. To do so, we defined walking as the reference activity (henceforth subindex w). Each user u needs to perform this activity during a short period of time, where the mean μ_w^u and the standard deviation σ_w^u are computed; the TS is then normalized using $< \mu_w^u, \sigma_w^u >$. S_1 is calculated for the normalized TS during this walking period, calculating its mean ($\mu_{wS_1}^u$) and standard deviation ($\sigma_{wS_1}^u$). Then we set $\alpha = 3 \times \sigma_{wS_1}^u$, which means (for a normal distribution) that *a high value that is statistically the upper limit for S_1 when walking is a peak candidate.*

2.2 A Finite State Machine labelling the relevant peaks

The FSM proposed in [19] was designed considering the dynamics of a fall event; it has been found that in good percentage of the cases the FSM certainly detects the fall dynamics. However,this proposal makes use of thresholds to determine the magnitude of the acceleration peak and the time window around the peak value. Thus, we simplify this FSM (see left part of Fig. 3. Each detected local maximum peak changes the state to *Timing*, starting a timer of 2.5 seconds as was suggested in [19]. If the timer fires, the state moves into *Is a Peak*, where the feature extraction and further processing together with the classification takes place. The state changes to *No Peak* once the peak is labelled.

Moreover, we merged the two FSMs, the one proposed in [19] and the one proposed before. This solution, called ABBATE-MAX-PEAK, uses the S1 value and the concept of statistically out of range wrt the acceleration value when walking instead of the acceleration and a predefined threshold. However, the timers and the calculation of the peak window is performed as stated in the work of Abbate et al.

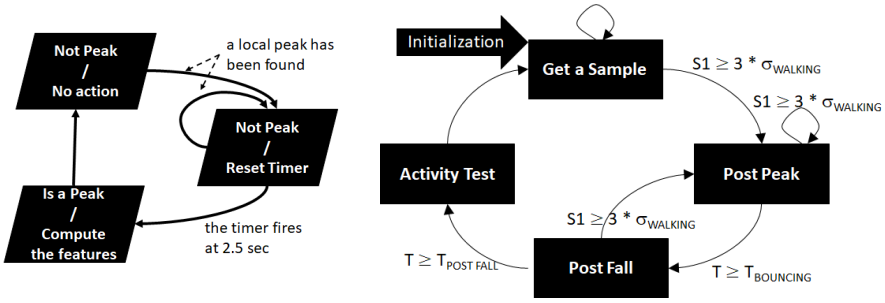

Fig. 3: The proposed FSMs. To the left, the FSM used in the MAX-PEAK solution. To the right, the FSM used in the ABBATE-MAX-PEAK approach. This latter makes use of the same FSM as proposed in Abbate et al [19], but using the S1 and the concept of statistically out of range proposed in the previous section.

2.3 Extended feature subset

The research in [19] introduced up to 8 features that were computed in the context of the dynamic window around each detected peak; these features includes too many parameters and predefined thresholds and tuning them becomes a hard task. In [24] 4 different well-known transformations were proposed; the main novelty were that these 4 features were computed for both the pre-peak and post-peak parts of the peak window. The rationale was that, provided there are differences in the human behavior before and after a fall, the differences must be reflected on these features as well. Therefore, the $[a_{t-2k-1}, a_t]$ window is split in two: before ($I_B = [a_{t-2k-1}, a_{t-k-1}]$) and after ($I_A = [a_{t-k+1}, a_t]$) the peak, respectively. For each of these sub-intervals the following transformations were calculated, with s and e meaning the bounds of the corresponding intervals:

- Average Absolute Acceleration Magnitude Variation, **AAMV**, computed as $AAMV = \sum_{t=s}^{e-1} |a_{t+1} - a_t|/N$, with N the number of samples in the interval $[s, e]$.
- Energy of the Acceleration Magnitude, **E**, calculated as $E = \sum_{t=s}^{e} a_t^2/N$
- Mean of the acceleration magnitude, **MN**, in the interval $[s, e]$.
- Standard Deviation of the acceleration magnitude, **SD**, in the interval $[s, e]$.

In this research we propose 3 more features from [26] together with the S_1 function proposed in [25] and adapted to this specific problem. This is the listing of these features.

- Amount of Movement, **AoM**, computed as $AoM = |max_{t=s}^{e}(a_t) - min_{t=s}^{e}(a_t)|$.
- Mean Absolute Deviation, **MAD**, computed as $MAD = \frac{1}{N} \sum_{t=s}^{e} |a_t - MN_{t=s}^{e}|$.
- Maximum of the differences, $\mathbf{S}_1$, proposed in [25] and computed using Eq. 1.

When using the MAX-PEAK, the 4 features from [24] plus the 2 first features from [26] are computed for i) the subinterval before the peak, ii) the subinterval after the peak and iii) for the whole peak window. The S_1 feature will be computed for the current peak and used as an input feature as well. For the MAX-PEAK we have a total of 19 features. In the case of ABBATE-MAX-PEAK, the same features as proposed in [19,22] are used.

3 Experiments and results

3.1 Data sets and experimentation set up

This research makes use of UMA Fall [27], a public staged falls data set. A total of 17 participants contributed to this data set, with up to 208 TS of simulated falls and a total of 531 TS. The TS were sampled at 20 Hz; in this research, we focus on the TS from the 3DACC sensor placed on a wrist. Each participant performed a non-fixed number of Activities of Daily Living (ADL) and falls.

The experimentation is divided in two parts. The first one is devoted to compare fall event detection, while the second one compares the performance of this study. In both of them, this study is compare with the proposals in [19,24].

The first part compares the methods using the well-known counters True Positive -TP-, True Negative -TN-, False Positive -FP- and False Negative -FN- in order to evaluate the performance of the event detection methods; we count a positive as a fall while a negative as non-fall. These counters are updated according to whether there are or not peaks detected in each of the participant's TS and the label this TS has. For each participant, the walking activity TS are used to set up its α value.

The second part makes use of densely connected neural network using Keras. The model includes, in all the cases, 3 layers (with 150 neurons each and a ReLU activation) plus one final layer of 1 neuron using a sigmoid activation and L2 regularizer with 0.001 as updating coefficient. A 0.4 percentage dropout layer is included between each two layers to avoid the overfitting. In all the cases, the models were allowed 30 epochs with a batch size computed as the number of instances divided by the number of epochs. The features extracted from the detected peaks are scaled to the interval [0.0, 1.0]. All the instances are labelled as FALL or NOT_FALL.

In this second part, we perform Leave-one-PARTICIPANT-out cross validation: we keep the data from the current participant for validation; the remaining instances are used to train and test the Deep Learning NN. A 10-fold cross validation is performed then using the train and test part. The mean Sensitivity and mean Specificity obtained for the validation data set are the metrics used in the comparison of the methods.

3.2 Comparison of event detection methods

Table 1 shows the counters for each of the event detection methods. The performance of the on-wrist Abbate is much worse than that of MAX-PEAK and MAX-PEAK-FSM if we consider the undetected alarms: 56 undetected falls for the on-wrist Abbate, 2 for the MAX-PEAK and MAX-PEAK-FSM and 3 for the ABBATE-MAX-PEAK. Besides, the on-wrist Abbate performs really well with the TN, perhaps due to the relatively high threshold used in detecting the fall events. The MAX-PEAK-FSM and the ABBATE-MAX-PEAK clearly outperform the MAX-PEAK in terms of reducing the false alarms.

On the other hand, when analyzing the UMA Fall data set, the number of peaks detected by each method were 201 for the on-wrist Abbate, 3073 for the MAX-PEAK, 449 for the MAX-PEAK-FSM and 531 for the ABBATE-MAX-PEAK. This means that for several TS the event detection considered more than one peak. Obviously, the sensitiveness of MAX-PEAK is much higher than those of on-wrist Abbate and MAX-PEAK-FSM, producing by far more false alarms than the other methods. Hence, the MAX-PEAK produces highly imbalanced data sets and we are not going to use it the next experimentation concerning with instances classification. A priory, the MAX-PEAK-FSM and the ABBATE-MAX-PEAK seem the best event detection methods.

Table 1: Event detection results for each participant.

	on-wrist Abbate				MAX-PEAK				MAX-PEAK-FSM				ABB-MAX-PEAK			
Pid	**TN**	**FP**	**FN**	**TP**	**TN**	**FP**	**FN**	**TP**	**TN**	**FP**	**FN**	**TP**	**TN**	**FP**	**FN**	**TP**
1	16	2	5	15	7	11	0	20	9	9	0	20	10	8	0	20
2	15	3	2	10	7	11	0	12	13	5	0	12	13	5	0	12
3	17	2	2	16	5	14	0	18	8	11	0	18	9	10	0	18
4	18	3	7	10	4	17	1	16	10	11	1	16	11	10	1	16
5	15	0	0	6	5	10	0	6	8	7	0	6	9	6	0	6
6	4	0	2	4	0	4	0	6	0	4	0	6	0	4	0	6
7	20	2	0	0	2	20	0	0	3	19	0	0	3	19	0	0
8	16	3	0	0	3	16	0	0	6	13	0	0	7	12	0	0
9	16	2	2	16	5	13	0	18	12	6	0	18	12	6	0	18
10	19	2	0	0	7	14	0	0	9	12	0	0	10	11	0	0
11	19	0	1	0	4	15	0	1	8	11	0	1	8	11	0	1
12	22	1	9	0	0	23	0	9	5	18	0	9	7	16	0	9
13	7	0	5	7	4	3	0	12	6	1	0	12	6	1	0	12
14	5	0	0	6	1	4	0	6	2	3	0	6	2	3	0	6
15	9	1	8	3	3	7	0	11	6	4	0	11	7	3	0	11
16	56	8	5	51	8	56	0	56	27	37	0	56	27	37	1	55
17	12	6	8	10	3	15	1	17	7	11	1	17	7	11	1	17
Total	**286**	**35**	**56**	**154**	**68**	**253**	**2**	**208**	**139**	**182**	**2**	**208**	**148**	**173**	**3**	**207**

3.3 Classification of Time Series

Results from the different configurations are shown in Table 2 and in the box plots in Fig. 4 and Fig. 5. Although paying attention to the figures in the Table the two approaches MAX-PEAK-FSM and ABBATE-MAX-PEAK dominates the on-wrist Abbate, if we pay attention to the box plots there is no clear winner.

On the one hand, the ABBATE-MAX-PEAK shows a surprising no variation performance, either for good (with the specificity) or for bad (the sensitivity). Nevertheless, we can not say the sensitivity of this configuration is worse than for the other methods: the results vary differently for each participant.

With all these figures and graphs we can state that perhaps more features need to be extracted and a better feature set must be needed in order to obtain a good robust performance for all the participants.

4 Conclusions

This research is focused on FD using wearable devices using a 3DACC sensor located on a wrist. The study has analyzed different event detection methods together with different transformations of the acceleration windows when an event is detected. The comparison has been performed using densely connected layers in a Deep Learning configuration using the Keras framework.

Results show that the two proposed event detection methods were much better than the previously used method because i) there is no need of a predefined

	on-wrist Abbate		MAX-PEAK -FSM		ABBATE-MAX-PEAK	
Pid	SEN	SPE	SEN	SPE	SEN	SPE
1	0.6667	0.9888	0.7633	0.9813	0.6850	0.8889
2	0.8391	1.0000	0.7696	1.0000	0.9231	1.0000
3	0.9027	0.9893	0.8676	0.9946	1.0000	1.0000
4	0.7730	1.0000	0.7676	1.0000	0.8882	1.0000
5	0.6000	1.0000	0.5923	1.0000	0.8571	1.0000
6	0.6167	0.9462	0.5958	0.9769	0.3750	0.9200
7	-	0.9551	-	0.9561	-	0.8842
8	-	0.9720	-	0.9764	-	0.9467
9	0.7085	0.9972	0.6872	0.9962	0.8429	0.7750
10	-	0.9608	-	0.9598	-	0.9250
11	0.5667	0.9799	0.4667	0.9871	0.5000	1.0000
12	0.6111	0.9825	0.6667	0.9796	0.5444	0.9167
13	0.7688	0.9985	0.7937	0.9970	1.0000	1.0000
14	0.8182	1.0000	0.8182	1.0000	0.6000	0.9000
15	0.7368	0.9946	0.7263	0.9957	0.8000	1.0000
16	0.8250	0.9900	0.7984	0.9909	0.9436	0.9442
17	0.6560	0.8949	0.6600	0.8782	0.7824	0.8923
AVRG	0.7207	0.9794	0.7124	0.9806	0.7673	0.9408
AVRG	**0.7207**	**0.9794**	**0.7124**	**0.9806**	**0.7673**	**0.9408**

Table 2: Classification results in this leave one participant out cross validation. SEN and SPE stand for Sensitivity and Specificity, while AVRG refers to the average of the metrics over all the participants. An hyphen (-) stands when the participant did not perform any staged fall, thus the value of the sensitivity becomes not calculable.

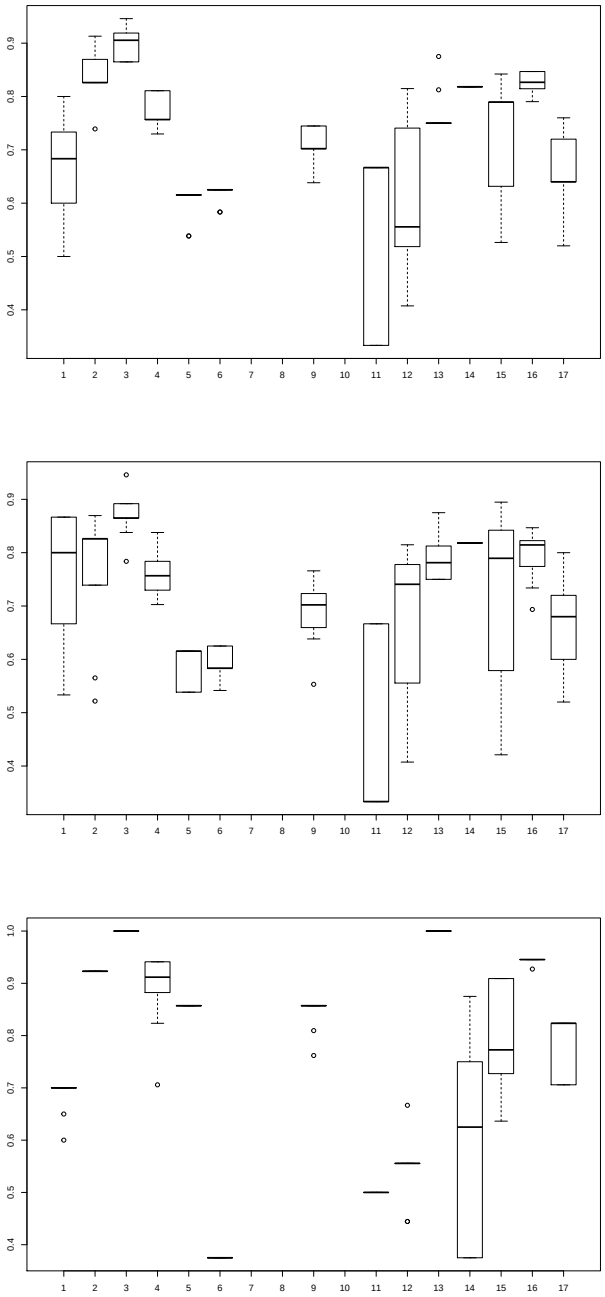

Fig. 4: Box plot of the sensitivity for the validation data set per participant. Upper, central and lower parts are for the on-wrist Abbate, the MAX-PEAK-FSM and the ABBATE-MAX-PEAK, correspondingly.

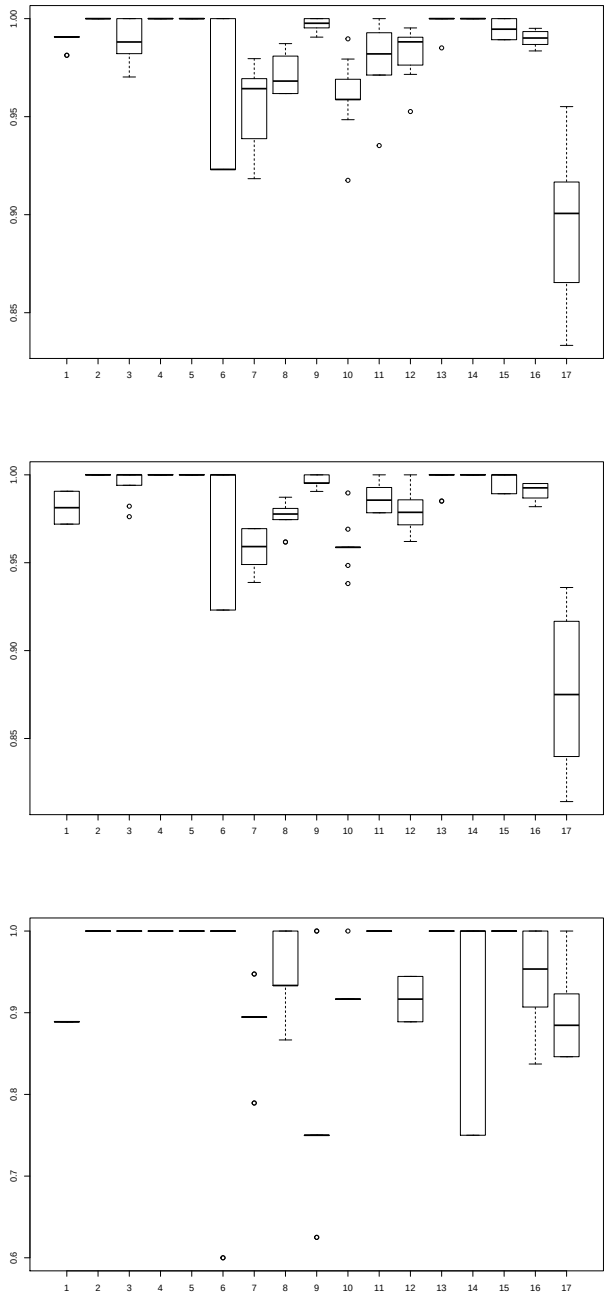

Fig. 5: Box plot of the specificity for the validation data set per participant. Upper, central and lower parts are for the on-wrist Abbate, the MAX-PEAK-FSM and the ABBATE-MAX-PEAK, correspondingly.

threshold and ii) because the number of undetected falls is almost negligible. While the baseline method fails detecting 56 out of 210 falls, the proposals detect 207 or 208 out of 210 falls. However, the transformations that have been studied did not improve the results obtained from the on-wrist Abbate solution. This might be due to an inefficient set of features that becomes redundant. Extra transformations (in the domain of the frequency and others) might be needed in order to enhance the results. Additionally, the use of other Deep Learning models, such as Long Short Term Memory or GA networks, can lead to better results at the cost of draining battery in case of being run in the wearable device or in a Smartphone.

ACKNOWLEDGMENT

This research has been funded by the Spanish Ministry of Science and Innovation, under project MINECO-TIN2017-84804-R, and by the Grant FC-GRUPIN-IDI/2018/000226 project from the Asturias Regional Government.

References

1. Zhang, S., Wei, Z., Nie, J., Huang, L., Wang, S., Li, Z.: A review on human activity recognition using vision-based method. Journal of Healthcare Engineering **2017** (2017)
2. Rimminen, H., Lindström, J., Linnavuo, M., Sepponen, R.: Detection of falls among the elderly by a floor sensor using the electric near field. IEEE Transactions on Information Technologies and Biomedicine **14** (2010) 1475–1476
3. Principi, E., DiegoDroghini, Squartinia, S., Olivetti, P., Piazza, F.: Acoustic cues from the floor: A new approach for fall classification. Expert Systems with Applications **60** (2016) 51–61
4. Igual, R., Medrano, C., Plaza, I.: Challenges, issues and trends in fall detection systems. BioMedical Engineering OnLine **12**(66) (2013)
5. Godfrey, A.: Wearables for independent living in older adults: Gait and falls. Maturitas **100** (jun 2017) 16–26
6. Zhang, T., Wang, J., Xu, L., Liu, P.: Fall detection by wearable sensor and one-class svm algorithm. In Huang DS., Li K., I.G., ed.: Intelligent Computing in Signal Processing and Pattern Recognition. Volume 345 of Lecture Notes in Control and Information Systems. Springer Berlin Heidelberg (2006) 858–863
7. Hakim, A., Huq, M.S., Shanta, S., Ibrahim, B.: Smartphone based data mining for fall detection: Analysis and design. Procedia Computer Science **105** (2017) 46–51
8. Wu, F., Zhao, H., Zhao, Y., Zhong, H.: Development of a wearable-sensor-based fall detection system. International Journal of Telemedicine and Applications **2015** (2015) 11
9. Bourke, A., O'Brien, J., Lyons, G.: Evaluation of a threshold-based triaxial accelerometer fall detection algorithm. Gait and Posture **26** (2007) 194–199
10. Huynh, Q.T., Nguyen, U.D., Irazabal, L.B., Ghassemian, N., Tran, B.Q.: Optimization of an acc. and gyro.-based fall det. algorithm. Journal of Sensors (2015)
11. Chaudhuri, S., Thompson, H., Demiris, G.: Fall detection devices and their use with older adults. Journal of Geriatric Physical Therapy **37** (2014) 178–196

12. Casilari-Pérez, E., García-Lagos, F.: A comprehensive study on the use of artificial neural networks in wearable fall detection systems. Expert Systems with Applications **138** (2019)
13. Fang, Y.C., Dzeng, R.J.: A smartphone-based detection of fall portents for construction workers. Procedia Eng. **85** (2014) 147–156
14. Fang, Y.C., Dzeng, R.J.: Accelerometer-based fall-portent detection algorithm for construction tiling operation. Autom. Constr. **84** (2017) 214–230
15. Kangas, M., Konttila, A., Lindgren, P., Winblad, I., Jämsaä, T.: Comparison of low-complexity fall detection algorithms for body attached accelerometers. Gait and Posture **28** (2008) 285–291
16. Casilari, E., Lora-Rivera, R., García-Lagos, F.: A wearable fall detection system using deep learning. In: Advances and Trends in Artificial Intelligence. From Theory to Practice. IEA/AIE 2019. Lecture Notes in Computer Science, vol 11606. (2019) 445–456
17. Xiaodan Wu, Cheng, L., Chu, C.H., Kim, J.: Using Deep Learning and Smartphone for Automatic Detection of Fall and Daily Activities. In: Smart Health. ICSH 2019. Lecture Notes in Computer Science , vol 11924. (2019) 61–74
18. Santos, G.L., Endo, P.T., de Carvalho Monteiro, K.H., da Silva Rocha, E., Silva, I., Lynn, T.: Accelerometer-based human fall detection using convolutional neural networks. Sensors (2019) 1–12
19. Abbate, S., Avvenuti, M., Bonatesta, F., Cola, G., Corsini, P., AlessioVecchio: A smartphone-based fall detection system. Pervasive and Mobile Computing **8**(6) (December 2012) 883–899
20. Delahoz, Y.S., Labrador, M.A.: Survey on fall detection and fall prevention using wearable and external sensors. Sensors **14**(10) (2014) 19806–19842
21. Khojasteh, S.B., Villar, J.R., de la Cal, E., González, V.M., Sedano, J., YAZG̈AN, H.R.: Evaluation of a wrist-based wearable fall detection method. In: 13th International Conference on Soft Computing Models in Industrial and Environmental Applications. (2018) 377–386
22. Khojasteh, S.B., Villar, J.R., Chira, C., González, V.M., de la Cal, E.: Improving fall detection using an on-wrist wearable accelerometer. Sensors **18** (2018) 1350
23. Villar, J.R., de la Cal, E., Fañez, M., González, V.M., Sedano, J.: User-centered fall detection using supervised, on-line learning and transfer learning. Progress in Artificial Intelligence **2019** (2019) 1–22
24. Villar, M., Villar, J.R.: Peak detection enhancement in autonomous wearable fall detection. In: 19th International Conference on Intelligent Systems Design and Applications. (2019)
25. Palshikar, G.K.: Simple algorithms for peak detection in time-series. Technical report, Tata Research Development and Design Centre (2009)
26. Villar, J.R., González, S., Sedano, J., Chira, C., Trejo-Gabriel-Galán, J.M.: Improving human activity recognition and its application in early stroke diagnosis. International Journal of Neural Systems **25**(4) (2015) 1450036–1450055
27. Casilari, E., Santoyo-Ramón, J.A., Cano-García, J.M.: Umafall: A multisensor dataset for the research on automatic fall detection. Procedia Computer Science **110**(Supplement C) (2017) 32 – 39