Neurocomputing 500 (2022) 231–240

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Towards effective detection of elderly falls with CNN-LSTM neural networks

Enol García [a], Mario Villar [a], Mirko Fáñez [a], José R. Villar [a,*], Enrique de la Cal [a], Sung-Bae Cho [b]

[a] Computer Science Department, University of Oviedo, Oviedo, Spain
[b] Computer Science Department, Yonsei University, Republic of Korea

ARTICLE INFO

Article history:
Received 19 February 2021
Revised 11 May 2021
Accepted 13 June 2021
Available online 25 May 2022

Keywords:
Fall detection
Recurrent neural networks
Data augmentation

ABSTRACT

Fall detection is a very challenging task that has a clear impact in the autonomous living of the elderly individuals: suffering a fall with no support increases the fears of the elderly population to continue living by themselves. This study proposes the use of a non-invasive tri-axial accelerometer device placed on a wrist to measure the movements of the participant. The novelty of this study is two fold: on the one hand, the use of a Long-Short Term Memory Neural Network (LSTM) for classification of the Time Series and, on the other hand, the proposal of a novel data augmentation stage that introduces variability in the training by merging the Time Series gathered from both human activities of daily living. The experimentation shows that the combination of a LSTM model together with the data augmentation produces more robust and accurate models that perfectly cope with the validation stage; the high impact fall event detection can be considered solved.

1. Introduction

Falls are accepted as an illness for the elderly population coded as E880-E888 in International Classification of Disease-9 (ICD-9), and as W00-W19 in ICD-10 [1]. The number of falls increments with the age and frailty and people living in nursing homes fall more than people living by their own. The percentage of elderly living in long-term care institutions is approximately the 30%-50% and about the 40% experience recurrent falls [1]. Whenever a senior suffers a fall, the faster the assistance the better [2]. Therefore, developing devices and models able to detect the falls may help in i) extending the autonomous living of the elderly and ii) notifying the caring staff about a fall of an inhabitant of the nursing home or caring institution. Nevertheless, a solution with the suitable accuracy is still a focus of research [3,4].

In spite of the effort made in research, the performance of the Fall Detection (FD) methods is still far from being acceptable [5]: the data sets used in learning the models do not represent the high variability of possible scenarios, decreasing the obtained generalization. In addition, the capability of the models must match the mentioned variability in the fall scenarios and cases. Finally, ubiquity is one of the requirements that are requested to FD, so people can be monitored everywhere without constraints and restrictions.

This study focuses on wearable FD using a tri-axial accelerometer (3DACC) placed on a wrist together with Deep Learning (DL) neural network (NN) models. On the one hand, a 3DACC placed on a wrist allows the user to perform his/her Activities of Daily Life (ADL) without extra requirements apart from using a smartband or a smartwatch. On the other hand, DL has become one of the leading paradigms for Machine Learning that has provided improvements in different domains of application from computer vision [6], to disease recognition [7], including body activity monitoring [8]. More interestingly, the capacity and the generalization capabilities of DL NN are well known in the literature [9]. In this study we focus on Long-Short Term Memory (LSTM) models, which have been found suitable for modelling Time Series (TS) [10]. To tackle the problem of generalization, this research proposes a Data Augmentation (DA) technique that merges TS from ADL and Fall, with the aim of providing a wider scenario of fall events in the training of the models.

The main contributions of this study are i) a novel DA method that will allow to enrich and obtain a more realistic and balanced training data set by merging TS from an ADL into a FALL TS, ii) the use of a LSTM to classify the TS and iii) a complete experimentation is designed, providing an in-depth glance of the state-of-the-art solutions in FD. The structure of the paper is as follows. The related work follows next, while Section 3 includes a detailed

* Corresponding author.
E-mail addresses: mirko@mirkoo.es (M. Fáñez), villarjose@uniovi.es (J.R. Villar), delacal@uniovi.es (E. de la Cal), sbcho@yonsei.ac.kr (S.-B. Cho).

https://doi.org/10.1016/j.neucom.2021.06.102
0925-2312/

description of the novel DA proposal and the different DL models developed for this research. In Section 4 the data sets and the experimentation set up are completely detailed. The results and the discussion on them are depicted in Section 5. Finally, the paper ends with the conclusions drawn from the study.

2. Related work

FD has been studied using many different technologies: video monitoring [11], motion sensors [12] or floor and sound sensory systems [13]. Wearable devices have been widely used due to the unobtrusiveness and ubiquitous character: 3DACC [14], barometer and gyroscope [15], electromyography [16] or angle [17]. Concerning with the FD intelligent decision making, we can mention the use of predefined thresholds [18], Support Vector Machines (SVM)[19,20], dictionaries and TS representation [21], clustering [22], NN [23], Random Forests (RF) [24], Hidden Markov Models [25–27], ensemble of classifiers [24], among others.

Nonetheless, the generalization capabilities of the different solutions are far to cope with the real task of FD [5]. Besides, DL has attracted the attention of the research community due to its capacity and generalization capabilities and has been already applied to FD and Human Activity Recognition (HAR) as well [3]. Basically, there are three different approaches: i) Convolutional Neural Networks (CNN), ii) Temporal and Recurrent Neural Networks (RNN) and iii) Auto-Encoders (AE) and Generational Adversarial Neural Networks (GANN); as mentioned before, this study focuses on supervised solutions and this latter case is not analyzed. Concerning with CNN applied to FD, the studies presented in [28–32] showed different architectures in their studies. A sequence of 2 stages followed by a final dense layer and an output neuron was proposed in [28], each stage including a CNN layer plus a max-pooling for subsampling. In [32], 4 blocks were sequenced, each block containing several layers: a CNN followed by a batch normalization, a ReLU and a max-pooling layers. The CNN layers decreases in dimension (128, 64, 32 and 16 nodes). A batch size of 64 TS, L2 regularization, dropout and a Data Augmentation (DA) consisting in a random rotation of the TS were also included in the training of the model.

Furthermore, an interesting comparison between CNN, K-Nearest Neighbours, SVM and threshold-based solutions for fall detection was published in [29]. In this case, the authors proposed two convolutional layers of 30 and 15 nodes with a filter size of 4 for both and with pool size of 2. Finally, a dense layer of 6 nodes and a soft-max classification layer completed the CNN model. Windows of 1 s long of the acceleration magnitude were feed to the network. Finally, the studies in [30,31] compared CNN and LSTM models finding better performance on the latter. CNN has also been employed in HAR [33–38]. Each of these studies prepared their own model description, including 1D-CNN [37], input dimension in terms of a different sliding window configuration and parameter set. In the comparison found in [36], different models were compared: CNN, LSTM, CNN + LSTM, XGBoost, AE + RF and Multilayer Perceptron (MLP); surprisingly, MLP beating all the models in this comparison.

Long-Short Term Memory Neural Networks (LSTM) is the most widely used RNN in FD [39–43,30,31]. One of the simplest architectures is shown in [39], including a 3-neuron input layer followed by two Gate Recurrent Unit (GRU) layers of 20 neurons each and a final stage of 2 neurons Soft-Max layer. Similarly but using LSTM, the study in [41] included an input window of 30 samples corresponding to 1 s, two LSTM layers followed by a feed-forward NN of 200 neurons each and a feed-forward NN of 2 neurons were used to classify as Fall or Not Fall inputting 1-s windows of the acceleration's magnitude. A sliding window of 1.28 s (at 32 Hz of sampling rate) is normalized and one or two LSTM (or GRU) layers (3 neurons each) followed by a dropout, a dense layer and a Soft-Max output neuron. Moreover, [40] proposed a window size set to $w = 256$ samples corresponding to 1.28 s and 50% of overlapping. The model's architecture included an input layer ($w \times 3$) followed by a Dense layer ($w \times 3$ to $w \times 32$) plus a batch normalization and dropout, two LSTM layers ($w \times 32$ nodes each) implementing dropout and finishing with a Dense layer ($w \times 32$ to 1×32) and a Soft-Max final neuron. Actually, the proposal in [41] seems similar to the approach in [40] but using the magnitude of the acceleration instead of the three components.

In addition, 3DACC has been used together with gyroscope in [43], including a windowing of 51 values from the 6 channels. Then, an LSTM with 100 nodes followed by a Dense layer and a Soft-Max unit is proposed to classify the TS sequences. Finally, [30] proposed a combination of CNN and LSTM as follows: 3 CNN layers of 128 nodes each, followed by 2 CNN layers of 96 nodes each, a LSTM of 128 nodes and a Dense layer of 2 nodes. This proposal was also analysed in [31], including a rotation-based DA in the training of the models. In addition, there are studies of RNN used in HAR [44,36,45], which perform similarly with their own network architecture.

Besides, all the DL studies are based on staged fall data sets. These data sets include a set of participants performing ADL and mimicking fall events. These data sets are important due to the difficulty in gathering data from real cases; however, the number of fall events and how they are mimicked play an important role in the generalization capabilities of the models. On the one hand, the balance between Fall and Non Fall TS is usually in compromise; on the second hand, the fall events start with the subject still; that is, there is no ADL just before the fall event. To our knowledge, only a few DA solutions have been proposed in the literature to tackle this problem using either the rotation of the acceleration axis [30,32,31,41] or by shifting and scaling the TS [46]. The former is a 3D linear angular transformation while the latter includes a random shifting of the TS in one direction and an independent random scale of the differences among consecutive values. Clearly, there is still room for improvement in trying to obtain more balanced data sets that cover more real fall events.

3. Enriching FD with DL and data augmentation

This study focuses on supervised approaches to classify a TS window as Fall or Not Fall using DL models and how to enhance FD to obtain generalized models. To do so, we first propose a novel DA technique that merges two TS, one from a Fall and one from a compatible ADL, to produce a new TS that resembles performing an ADL just before a fall event. Furthermore, we propose the use of LSTM layers after a convolutional stage followed by a Dense classifier to classify the TS. The following subsections describe these contributions in the mentioned order.

3.1. A novel TS data augmentation

The proposed DA technique addresses the problem of providing with enough valid data in order to train DL RNN models, solving the difficulties detected in [47]. To do so, the DA mixes two TS: one from a fall event and one from a randomly chosen compatible TS. The benefits of this solution is twofold: On the one hand, mixing two TS enriches the data set with different scenarios where a fall might occur. On the second hand, the increased number of windows due to the DA helps in balancing the training data set and, thus, obtaining more robust and generalized models. So, basically, the DA technique transforms the training data set from dealing with the staged falls only (which are a sequence of stand still, fall onto a mattress and remain still, see the upper part in Fig. 1) to also include more realistic staged falls (such as the presented in the bottom part in the same figure, where an activity was being performed at the moment a fall occurs). In this Figure, the top row shows the original fall TS, while the second row depicts the chosen compatible TS; in this case, the square marks the most similar window to merge with the fall TS. The third row in Fig. 1 shows the fusion of the two former TS. Finally, the bottom row depicts the outcome of the process after scaling and shifting. In all of them, the calculated acceleration magnitude is shown. The different relevant positions in Algorithm 1 are marked in the TS.

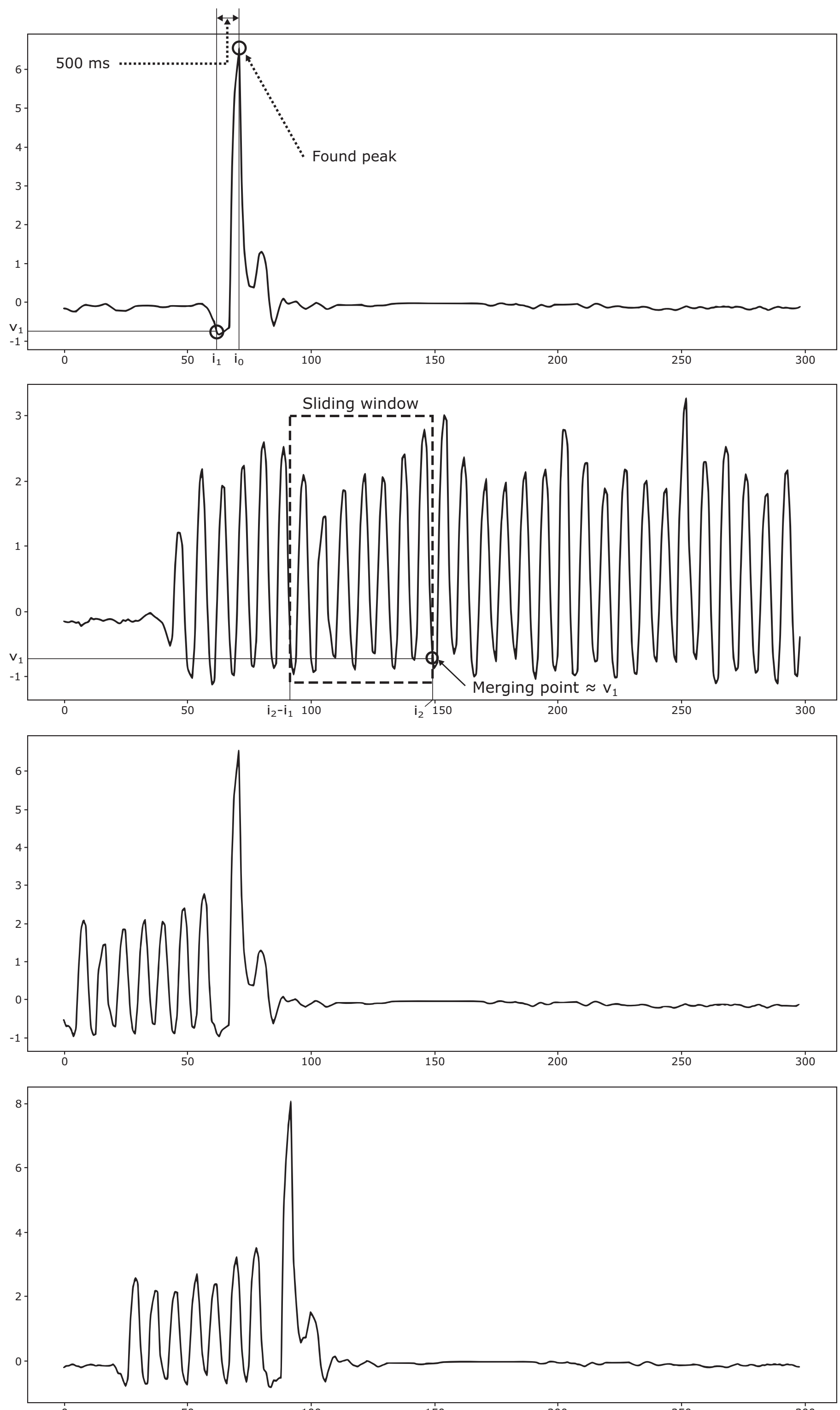


Fig. 1. Example of the performance of the Data Augmentation procedure for a fall TS: From the the original fall TS and a compatible TS on the two upper rows to the merged TS and the scaled and shifted final outcome of the DA in the two latter rows. For details, see the text.

In addition to the compatible ADL TS for each fall type, the number of augmentations (nDA_x, where x is the type of fall) for each fall type should be given. Interestingly, the DA will produce $nDA_x + 1$ TS per fall TS: nDA_x coming from the DA and the proper fall TS: this fact makes easier to consider all the fall scenarios in the training.

The design and implementation of the DA considers the magnitude of the acceleration (MA) of the TS to merge. Firstly, let us define what we refer as *compatible TS*. Given a Fall TS $\overrightarrow{ts_i}$ and its label l_i, we define a set of labels being compatible with l_i, denoted as C_i, as those labels representing a certain ADL that can be performed just before the type of fall event l_i. As an example, if we consider the type of fall *lateral falling from a bed*, it is not possible to be walking just before the fall: the subject should be lying on a surface. In the same way, you can be walking, running or standing still when you fall forward because you trip; however, you can not be sitting. Therefore, it is very important to set the compatible ADL set for each type of fall before performing the DA.

Algorithm 1 describes the steps followed in the DA task. Any TS labeled as an ADL is augmented using a random scaling factor and shift. Scaling is performed as stated in Eq. 1, with $\rho \in [-\mathrm{P}, \mathrm{P}]$ a random scalar that multiplies the differences for all the TS samples. Shifting represents a random circular shift to the right or the left; this shift involving at most *Sh* samples. Both P and *Sh* are parameters to the method.

$$a_{axis}(t) = a_{axis}(t) + \rho \times (a_{axis}(t) - a_{axis}(t-1)) \tag{1}$$

To augment a Fall TS TS_F we use a more complex procedure. Firstly, a random compatible TS TS_C is chosen from the data set. The selection of this compatible TS is chosen a first instance among those available for the same participant; if no compatible TS is found, then the selection is extended to all the compatible TS in the data set. Secondly, we compute the MA for both TS_F and TS_C, denoted as MA_F and MA_C. Thirdly, we find the position i_0 of the fall peak in MA_F using the methodology proposed in [48] and reported in [49,50]. The mean and standard deviation (std) of being standing still are used to normalize MA_F; these values are computed with the 50 first samples from all the Fall TS and they are similar to the gravity value and the deviation due to small movements, correspondingly. The threshold for peak discovering is set to $1.5 \times std$.

Next, we denote the point i_1 as the sample where the fall event started, which is 500 ms before i_0 [23]. Then we find the point i_2 in MA_C with the nearest value to $MA_F[i_1]$ such that $i_2 > i_1$. The following step is merging the two TS; to do so, the samples in TS_C from $i_2 - i_1$ to i_2 are extracted and introduced into TS_F before i_1. It is worth noticing that both TS_F and TS_C includes 3 variables, one from each axis. Finally, the obtained TS is shifted and scaled as explained before. Fig. 1 illustrates the complete procedure.

Finally, it is worth noticing that a Fall TS will produce several sliding windows; some of them will include the fall event, others will not; these sliding windows need to be properly labelled. In this study, every window containing the peak event at point i_0 will be labelled as Fall; otherwise, they are considered Non_Fall.

Algorithm 1 The Data Augmentation algorithm.

```
Algorithm 1: The Data Augmentation algorithm.
Input: A TS labelled data set
Input: The map containing all the compatible TS C
Input: The current TS t_c and participant p
Input: scaling factors and shift limits
Result: outcome, the augmented TS
if t_c is an ADL then
    outcome ← randomly scale and shift the current TS;
    Return outcome
else
    if C(t_c, p) not null then
        t_n ← select a random TS from C(t_c, p);
    else
        t_n ← select a random TS among those compatible with t_c;
    end
    T_c ← is the acceleration magnitude for t_c;
    T_n ← is the acceleration magnitude for t_n;
    i_0 ← find the position of the falling peak in T_c;
    i_1 ← shift 500 ms to the left of i_0;
    v_1 ← obtain T_c[i_1], the value in T_c at [i_1];
    i_2 ← obtain from T_n[i_1 : ] the position of the nearest value to v_1;
    outcome ← concatenate t_n[i_2 − i_1 : i_2] and t_c[i_1 : ];
    Return outcome
end
```

3.2. LSTM models applied to FD

In this study we have chosen a LSTM approach for FD, which is based on the proposal of [10]. In this study, a combination of a CNN together with a LSTM is used to detect traffic anomalies: the CNN part is in charge of the feature transformation, while the LSTM part is responsible of extracting the temporal patterns and relationships. For the sake of simplicity we do not include

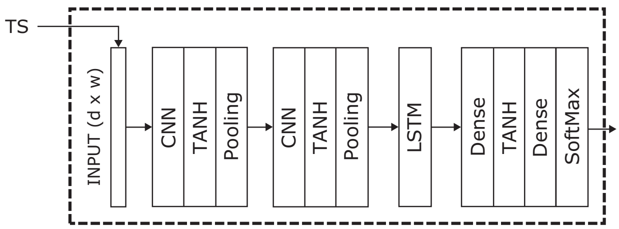


Fig. 2. The architecture of the $LSTM_{FD}$ model [10]. The constants w and d refer to the size of the multivariate TS window and the number of features, correspondingly.

the equations of the model, which are completely describe in the cited study.

Basically, the model includes two CNN stages followed by an LSTM and a final Dense layer to detect anomalies; this model is referred from now on as $LSTM_{FD}$. The architecture of the network is shown in Fig. 2, where w is the window size and d is the number of features (3 when the acceleration components are used, 4 when the magnitude of the acceleration is included or 1 when just this latter is used).

Each CNN stage includes a convolutional, an activation layer and a pooling layers. A convolutional layer includes of 64 nodes with kernel of size 5 and stride 1, an activation layer includes hyperbolic tangent activation functions; finally, a pooling layer includes max kernels of size 2 and stride 2. The aim of these pooling layers is to avoid overfitting as they decrease the spatial size and, consequently, the number of parameters in the network. As explained in [10], these layers effectively reduce spatial size by applying a max operation independently for each depth slice.

The LSTM layer includes 64 nodes and the Dense layers include 32 nodes each. In this study, three different versions of this model are used. Whenever an univariate TS is used, the CNN becomes 1D-CNN of 64 filters, just exactly the same approach as in [10].

4. Materials and methods

4.1. Staged falls data sets

Several staged fall data sets have been published in the literature [51], each data set include a set of sensors located in one or more places of a body. From all the data sets in the literature ([52,39] among others), this study choses the UCI-FALL [53], which gathered data with a 3DACC (sampling frequency of 25 Hz, with a 12 g sensors) placed on a wrist with a sufficient number of participants (17, all of them performing the same number of ADL and staged falls) and TS (1843 staged falls and 3326 ADL recordings). Up to 20 different staged fall types are considered (Forward, Lateral and Backwards falls among them), and 16 ADL such as running, walking, squatting, bending, sitting down, stumbling, lying on bed, etc.

The configuration of compatible ADL for each fall type was designed analysing each label and the magnitude of the acceleration for the different TS, but also considering logical and physical deductions. Table 1 shows the configuration. Besides, the number of TS augmentations to be performed on each of the ADL and Fall types must be determined. To do so, the number of sliding windows were estimated for each label, then the number of TS augmentations were proposed based on i) significantly increasing the number of Fall event windows and ii) balancing the data set. Apart from that, the remaining DA parameters (scaling factor P and maximum shifting samples Sh) are set to where 0.3 and 10%, correspondingly.

In this study we use a 3-s-long sliding windows with an overlapping of 2 s (that is, a shift of 1 s) considering the fall dynamics

Table 1
Compatibility between ADL and fall types for the DA.

Fall type	Compatible TS
Front-lying	walking-fw, stumble, jogging, limp, trip-over
Front-protecting-lying	walking-fw, stumble, jogging, limp, trip-over
Front-knees	walking-fw, stumble, limp, trip-over
Front-knees-lying	walking-fw, stumble, limp, trip-over
Front-quick-recovery	walking-fw, stumble, limp, trip-over
Front-slow-recovery	walking-fw, stumble, jogging, limp, trip-over
Front-Right	walking-fw, jogging, stumble, limp, trip-over
Front-Left	walking-fw, jogging, stumble, limp, trip-over
Back-Sitting	walking-bw, squatting-down, limp, sit-chair, sit-sofa, sit-air
Back-lying	walking-bw, squatting-down, limp, sit-chair, sit-sofa, sit-air, sit-bed
Back-Right	walking-bw, squatting-down, limp, sit-chair, sit-sofa, sit-air, sit-bed
Back-Left	walking-bw, squatting-down, limp, sit-chair, sit-sofa, sit-air, sit-bed
Right-sideway	walking-fw, stumble, jogging, limp, trip-over
Right-recovery	walking-fw, stumble, jogging, limp, trip-over
Left-sideway	walking-fw, stumble, jogging, limp, trip-over
Left-recovery	walking-fw, stumble, jogging, limp, trip-over
Rolling-out-bed	lying-bed, rising-bed
Podium	walking-fw, walking-bw
Syncope	walking-fw, walking-bw
Syncope-wall	walking-fw, walking-bw

Table 2
Determining the number of TS augmentations per Fall TS. Each TS augmentation of a Fall TS introduces more Non_Fall windows. Therefore, a compromise is needed. NoW stands for the Number of sliding Windows.

	UCI-FALL
Number of Non_Fall TS	3326
NoW for each Non_Fall TS	14
Number of Fall TS	1843
NoW for each Fall TS including the fall event	4
NoW for each Fall TS not including the fall event	10
Established Number of Augmentations	10

and length proposed in [23]: a sliding window can perfectly include a fall event. With this sliding window configuration, Table 2 shows the number of sliding windows.To setup the number of Fall TS augmentations we used the ratio of the total number of windows labelled as Non_Fall versus the total number of windows labelled as Fall (see Fig. 3). In this Figure, the black line represents the ratio of the total number of windows labelled as Non_Fall versus those labelled as Fall for each number of TS augmentations, while the red line represents the variation of the ratio. We have considered that the variation ratio has stabilized for 10 or more TS augmentations.

Finally, each TS needs to be scaled to the interval $[0.0, 1.0]$. To do so, each axis component is individually scaled considering the maximum value admissible for the sensor used in the data set.

4.2. The experimental setup

The classification problem is reduced to a two class problem; therefore, all the TS are labelled either as FALL (F) or NOT_FALL (NF). However, the original label of a TS will be used in the posterior analysis with the aim of determining the weakness and strengths of each configuration.

As mentioned before, a sliding window of 3 s long and a shift of 1 s will be used, thus consecutive windows show a 2 s overlap. The number of augmented TS has been calculated as explained in the previous section to balance the number of TS sliding window from each class. The final obtained numbers for each TS label are shown in Table 3.

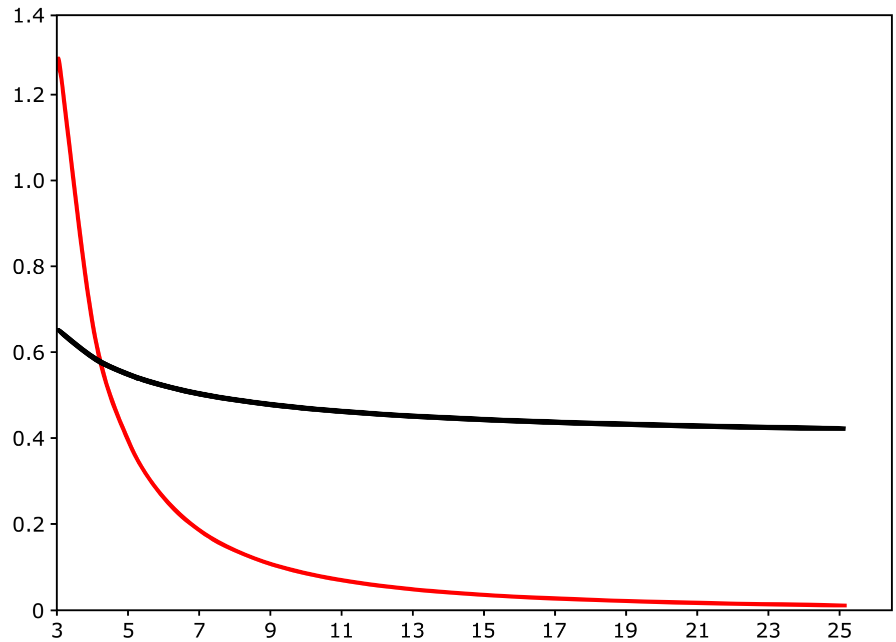


Fig. 3. Determination of the number of TS augmentations for the UCI-FALL data set. The black line represents the ratio of the total number of windows labelled as Non_Fall versus those labelled as Fall for each number of TS augmentations. The red line represents the variation of the ratio.

Table 3
Number of sliding windows for each of the two labels. NoW stands for Number of sliding Windows, while DA reps refers to the number of augmentations of each Fall TS. Indexes $_0$ and $_{DA}$ refer to before and after DA.

Data set	**FALL** NoW_0	**NOT_FALL** NoW_0	**DA reps**	**FALL** NoW_{DA}	**NOT_FALL** NoW_{DA}
UCI-FALL	239433	1342166	10	2394330	6727145

The experimentation uses a classical *Leave-One Participant-Out* cross validation, where in each fold a participant is chosen for validation (hence, all its TS are kept for validation of the model). A *10-fold cross validation* is performed on data from the remaining participants, splitting the TS into 10 different training and testing folds. Three different scenarios are evaluated: i) using the three acceleration components as a multivariate TS (denoted as XYZ), ii) using the magnitude of the acceleration instead (denoted as M-TS) and iii) using the three acceleration components plus the magnitude of the acceleration as a fourth variable TS (denoted as XYZM).

We use the Accuracy (ACC, Eq. 2), the Sensitivity (SENS, Eq. 3) and Specificity (SPEC, Eq. 4) to measure the performance of the models while training and testing. To compare the models we calculate these metrics on the validation data set; measuring their performance with the different unseen participants we can estimate the generalization capabilities of the models. The counters (TP, TN, FP and FN) reflect the number of TS from each participant that are correctly or wrongly classified; a TS is classified as a FALL if any of its sliding windows is labelled as a FALL.

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \tag{2}$$

$$SENS = \frac{TP}{TP + FN} \tag{3}$$

$$SPEC = \frac{TN}{TN + FP} \tag{4}$$

Finally, the following enhancements are used in the learning processes. Both L2 regularization and a 0.5 dropout are used. The learning rate is fixed according to the values given in the corresponding papers with the logical tuning. The batch size is set to 32, inducing batches of approximately 1121 windows; this batch size represents a compromise between having enough windows from the original data set plus extra information due to the DA. To obtain the learning rate and the number of epochs we performed the training with DA for 6 epochs for each possible value of this parameter. After these epochs, we decided to use the same learning rate for all the models (0.001) and 25 epochs.

5. Results and discussion

The obtained results are presented in Table 4 and Table 5. Table 4 shows the aggregated test results from the 10-fold cross validation in each experimental scenario. Besides, Table 5 depicts the performance of the $LSTM_{FD}$ model in the XYZM scenario when evaluated each of the leave-one-out participant. Both tables include the values of ACC, SENS and SPEC.

Some important remarks can be extracted from the training and testing results:

- When no DA is used in the training, the ACC values seems too high for the poor SENS results. However, this is due to the high number of NOT_FALL windows, which makes the ACC mask the real model performance.
- The poor SENS performance when no DA is used is due to, on the first hand, the reduced rate of fall windows used in training and, on the second hand, the lack of variability in the FALL TS.
- When using DA, the $LSTM_{FD}$ is able to learn the dynamics that are coherent with a fall event.
- The DA introduces a variability of fall TS that clearly enhances the Sensitivity of the model without penalizing the Specificity.
- Comparing the three scenarios, the single magnitude of the acceleration M-TS is clearly the worst, while the two remaining scenarios are comparable.

Table 4
Aggregation of the train and test results among all the participants, showing the different metrics for each scenario. When no DA is used the SENS values are sensibly worse and many fall windows are labelled as NOT_FALL.

	TRAIN RESULTS					
	$LSTM_{FD}$ no DA			$LSTM_{FD}$		
XYZ	**MIN**	**AVG**	**MEDIAN**	**MIN**	**AVG**	**MEDIAN**
ACC	**0.9953**	**0.9959**	**0.9958**	0.9900	0.9952	0.9954
SENS	0.6643	0.6727	0.6736	**0.9101**	**0.9331**	**0.9361**
SPEC	**0.9959**	**0.9964**	**0.9964**	0.9819	0.9945	0.9957
M-TS	**MIN**	**AVG**	**MEDIAN**	**MIN**	**AVG**	**MEDIAN**
ACC	**0.9925**	0.9928	0.9926	0.9847	**0.9933**	**0.9930**
SENS	0.6421	0.6509	0.6509	**0.8697**	**0.9192**	**0.9200**
SPEC	**0.9934**	**0.9937**	**0.9937**	0.9749	0.9943	0.9956
XYZM	**MIN**	**AVG**	**MEDIAN**	**MIN**	**AVG**	**MEDIAN**
ACC	**0.9959**	**0.9962**	**0.9962**	0.9932	0.9950	0.9950
SENS	0.6691	0.6765	0.6771	**0.9047**	**0.9334**	**0.9353**
SPEC	**0.9963**	**0.9966**	0.9965	0.9920	0.9959	**0.9968**
	TEST RESULTS					
	$LSTM_{FD}$ no DA			$LSTM_{FD}$		
XYZ	**MIN**	**AVG**	**MEDIAN**	**MIN**	**AVG**	**MEDIAN**
ACC	**0.9950**	**0.9967**	**0.9967**	0.9900	0.9952	0.9954
SENS	0.7133	0.7447	0.7478	**0.9101**	**0.9331**	**0.9361**
SPEC	**0.9871**	0.9930	0.9956	0.9819	**0.9945**	**0.9957**
M-TS	**MIN**	**AVG**	**MEDIAN**	**MIN**	**AVG**	**MEDIAN**
ACC	**0.9919**	**0.9940**	**0.9942**	0.9847	0.9933	0.9930
SENS	0.6851	0.7213	0.7230	**0.8697**	**0.9192**	**0.9200**
SPEC	**0.9849**	0.9930	0.9933	0.9748	**0.9943**	**0.9956**
XYZM	**MIN**	**AVG**	**MEDIAN**	**MIN**	**AVG**	**MEDIAN**
ACC	**0.9952**	**0.9968**	**0.9969**	0.9932	0.9950	0.9950
SENS	0.7192	0.7448	0.7508	**0.9047**	**0.9334**	**0.9353**
SPEC	**0.9864**	**0.9969**	**0.9968**	0.9920	0.9959	0.9967

Table 5
Results obtained for each leave-one-out participant for the $LSTM_{FD}$ trained without DA and with DA in the XYZM scenario. STD stands for standard deviation, while Id stands for the participant identification number.

	XYZM					
	no DA			**with DA**		
Id	**ACC**	**SENS**	**SPEC**	**ACC**	**SENS**	**SPEC**
101	0.9746	0.9500	**1.0000**	**1.0000**	**1.0000**	**1.0000**
102	0.9835	0.9901	0.9753	**1.0000**	**1.0000**	**1.0000**
103	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
104	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
105	0.9222	**1.0000**	0.8250	**1.0000**	**1.0000**	**1.0000**
106	0.9780	0.9703	0.9877	**1.0000**	**1.0000**	**1.0000**
107	0.9858	**1.0000**	0.9684	**1.0000**	**1.0000**	**1.0000**
108	0.9486	0.9160	0.9895	**1.0000**	**1.0000**	**1.0000**
109	0.9721	0.9916	0.9479	**1.0000**	**1.0000**	**1.0000**
110	0.9954	0.9917	**1.0000**	**1.0000**	**1.0000**	**1.0000**
203	**0.9945**	0.9901	**1.0000**	**0.9945**	**1.0000**	**1.0000**
204	0.9945	0.9901	**1.0000**	**1.0000**	**1.0000**	**1.0000**
205	0.9604	0.9828	0.9302	**1.0000**	**1.0000**	**1.0000**
206	0.9889	0.9900	0.9875	**1.0000**	**1.0000**	**1.0000**
207	0.9780	0.9604	**1.0000**	**1.0000**	**1.0000**	**1.0000**
208	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
209	0.9587	0.9672	0.9479	**1.0000**	**1.0000**	**0.9954**
Average	0.9785	0.9818	0.9741	**0.9997**	**1.0000**	**0.9997**
Median	0.9835	0.9901	0.9895	**1.0000**	**1.0000**	**1.0000**
STD	0.0005	0.0005	0.0020	**0.0000**	**0.0000**	**0.0000**

Table 5 compares the performance of $LSTM_{FD}$ trained with and without DA in the leave-one participant-out cross validation test results (with the TS from the participant kept for testing) and for the XYZM scenario. It can be seen how an almost ideal model performance is obtained when trained with DA as long as the model perfectly label the TS for the majority of the participants. In contrast, the model trained without DA exhibits a variability in the ACC and SENS; however, its performance is also good enough for the main part of the cases.

It is worth mentioning that the obtained results are coherent with what has been already published in the literature [54] although not comparable; this is so for several reasons.

Firstly, the cited study compare performance of CNN using data coming from sensors placed on the waist. Furthermore, the majority of studies are focused on using waist sensors, which is not the main point of the present research. However, this study makes use of data gathered from a on-wrist 3DACC, which makes the problem harder. Thus comparisons can not be presented among studies with sensors located on different body parts.

Secondly, CNN are reported with 95% of ACC in the studies when using the 3DACC placed on a wrist. Considering that, as stated in [54,55], the waist is the preferred location where to place a sensor for FD due to the possibility of fixing one of the acceleration components to the main component of the gravity for the majority of the cases. This is correct if the focus is to monitor fall detection on patients but it is not suitable when promoting the autonomous living of the elder. In this later case, solutions that do not force the subjects to use specific complements are better, e.g., a woman with a dress might be forced to use a belt, which might not make sense.

Finally, the cross-validation method employed for the majority of the studies makes use of 10-fold cross validation or similar schemes, where the TS are split into different groups, independently of the participant from whom the TS was gathered. From our point of view, this is not correct because the training and testing data set becomes dependent. That is, if you introduce TS from one participant in the training, the testing and the validation, then it is impossible to evaluate the generalization of the models as long as the training process has enough information from all the participants that are going to be used in the validation.

This is why in this study we have focused on leave-one participant-out: in this way, the validation is conformed with all the TS from a participant and neither the training nor the testing has any information regarding this participant. As a consequence, the validation in this study might have become worst than that of a method that shuffles all the TS. An in this latter case, the generalization capabilities are biased.

Independently of all these facts, the expected results when the sensor is located on the waist should be much better than those obtained when the user is located on-wrist. Besides, LSTM are known to extract and learn TS patterns more effectively than CNN and other DL networks.

Checking the results shown in [54], the results from this research become more impressive. On the one hand, the testing results are better for the CNN-LSTM combination than for the DL models used in [54]. On the second hand, the validation results are almost ideal for the $LSTM_{FD}$, outperforming the models used in the cited research.

6. Conclusions

This study focuses on fall detection through the use of non intrusive devices, in this case, three axial accelerometers placed on a wrist. A Deep Learning Recurrent model -more specifically, a Long-Short Term Memory Neural Network- have been analyzed for this purpose and referred as $LSTM_{FD}$ (see Section 3. In order to increase the variability of the patterns shown to the network training and also to mimic fall events while performing any other activities of daily living, a data augmentation stage is proposed. This data augmentation merges the Time Series from the considered activities and Time Series from a Fall to generate realistic combinations of an activity followed by a posterior fall. Three different scenarios are analyzed: using the three components of the acceleration, using the acceleration magnitude only or using the three components of the acceleration plus the magnitude.

The $LSTM_{FD}$ shows interesting and promising results for the multivariate Time Series cases and not so good performance for the case of using the magnitude of the acceleration only. The use of the data augmentation stage in the network training increased the performance measurements in all the cases, but still the magnitude of the acceleration case results are worse than the other cases. It is worth noticing that the data augmentation stage can still be enhanced with i) rotation of the axis and ii) finding the Time Series gathered from an activity of daily living that is the closest point to the fall starting event acceleration values for each component, reducing the gap that might be produced in the reported method. In any case, although there is still room for improvement, the combination of the Long-Short Term Memory network together with the data augmentation shows an impressive performance and the high impact fall events can be considered satisfactorily solved.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research has been funded by the Spanish Ministry of Science and Innovation under project MINECO-TIN2017-84804-R, PID2020-112726RB-I00 and the State Research Agency (AEI, Spain) under grant agreement No RED2018-102312-T (IA-Biomed). Moreover, this work was also partly supported by IITP grant funded by the Korean government (MSIT) (No. 2020-0-01361, AI Graduate School Program (Yonsei University)).

References

[1] A. Kalache, D. Fu, S. Yoshid, WHo Global report on falls Prevention in older Age, Tech. Rep., Wold Health Organization, URL: https://www.who.int/ageing/publications/Falls_prevention7March.pdf, 2020.
[2] HelpLine, Why is a quick response so important when the elderly fall, https://www.helpline.co.uk/blog/why-is-a-quick-response-so-important-when-the-elderly-fall/, [Online; accessed 28-June-2019], 2019.
[3] E. Casilari-pérez, F. García-lagos, A comprehensive study on the use of artificial neural networks in wearable fall detection systems, Expert Syst. Appl. 138 (2019) 112811.
[4] J.R. Villar, C. Chira, E. de la Cal, V.M. González, J. Sedano, S.B. Khojasteh, Autonomous on-wrist acceleration-based fall detection systems: unsolved challenges, accepted for publication in Neurocomputing.
[5] R.W. Broadley, J. Klenk, S.B. Thies, L.P. Kenney, M.H. Granat, Methods for the Real-World Evaluation of Fall Detection Technology: A Scoping Review, Sensors (Basel, Switzerland) 18 (7) (2018) 1–28, ISSN 14248220.
[6] J. Debayle, N. Hatami, Y. Gavet, Classification of time-series images using deep convolutional neural networks, in: J. Zhou, P. Radeva, D. Nikolaev, A. Verikas (Eds.), Tenth International Conference on Machine Vision (ICMV 2017), vol. 10696, SPIE, ISBN 9781510619418, 23, 2018.
[7] A.S. Becker, M. Marcon, S. Ghafoor, M.C. Wurnig, T. Frauenfelder, A. Boss, Deep Learning in Mammography, Invest. Radiol. 52 (7) (2017) 434–440, ISSN 0020–9996.
[8] C.A. Ronao, S.B. Cho, Human activity recognition with smartphone sensors using deep learning neural networks, Expert Syst. Appl. 59 (2016) 235–244, ISSN 09574174.
[9] C. Zhang, B. Recht, S. Bengio, M. Hardt, O. Vinyals, Understanding deep learning requires rethinking generalization, in: 5th International Conference on Learning Representations, ICLR 2017 – Conference Track Proceedings, 2019.
[10] T.-Y. Kim, S.-B. Cho, Web traffic anomaly detection using C-LSTM neural networks, Expert Syst. Appl. 106 (2018) 66 – 76, ISSN 0957-4174.
[11] E.E. Geertsema, G.H. Visser, M.A. Viergever, S.N. Kalitzin, Automated remote fall detection using impact features from video and audio, J. Biomech. 88 (2019) 25–32.
[12] Y. Peng, J. Peng, J. Li, P. Yan, B. Hu, Design and Development of the Fall Detection System based on Point Cloud, Proc. Comput. Sci. 147 (2019) 271–275.
[13] E. Principi, S. DiegoDroghini, P. Squartinia, F. Olivetti, Piazza, Acoustic cues from the floor: A new approach for fall classification, Expert Syst. Appl. 60 (2016) 51–61.
[14] A. Ngu, Y. Wu, H. Zare, A. Polican, B. Yarbrough, L. Yao, Fall Detection Using Smartwatch Sensor Data with Accessor Architecture, in: H. Chen, D. Zeng, E. Karahanna, B.I. (Eds.), Proceedings of the International Conference on Smart Health ICSH 2017, Lecture Notes in Computer Science, vol. 10347, Springer, 81–93, 2017.
[15] A.M. Sabatini, G. Ligorio, A. Mannini, V. Genovese, L. Pinna, Prior-to- and Post-Impact Fall Detection Using Inertial and Barometric Altimeter Measurements, IEEE Trans. Neural Syst. Rehabil. Eng. 24 (7) (2016) 774–783.
[16] G. Rescio, A. Leone, P. Siciliano, Supervised machine learning scheme for electromyography-based pre-fall detection system, Expert Syst. Appl. 100 (2018) 95–105.
[17] Y. Wu, Y. Su, R. Feng, N. Yu, X. Zang, Wearable-sensor-based pre-impact fall detection system with a hierarchical classifier, Measurement 140 (2019) 283–292.
[18] A. Bourke, P. van de Ven, M. Gamble, R. O'Connor, K. Murphy, E. Bogan, E. McQuade, P. Finucane, G. Olaighin, J. Nelson, Evaluation of waist-mounted tri-axial accelerometer based fall-detection algorithms during scripted and continuous unscripted activities, J. Biomech. 43 (2010) 3051–3057.
[19] R. Igual, C. Medrano, I. Plaza, A comparison of public datasets for acceleration-based fall detection, Med. Eng. Phys. 37 (9) (2015) 870–878.
[20] S.B. Khojasteh, J.R. Villar, C. Chira, V.M.G. Suárez, E.A. de la Cal, Improving Fall Detection Using an On-Wrist Wearable Accelerometer, Sensors 18 (5) (2018) 1350.
[21] M. Fáñez, J.R. Villar, E. de la Cal, J. Sedano, V.M. González, Transfer learning and information retrieval applied to fall detection, Expert Syst. (2020) e12522.
[22] M. Fáñez, J.R. Villar, E. de la Cal, J. Sedano, V.M. González, Improving Wearable-based Fall Detection with unsupervised learning, accepted for publication in International Logic Journal of the IGPL.
[23] S. Abbate, M. Avvenuti, F. Bonatesta, G. Cola, P. Corsini, AlessioVecchio, A smartphone-based fall detection system, Pervasive Mobile Comput. 8 (6) (2012) 883–899.

[24] J.R. Villar, E.A. de la Cal, V.M.G. Suárez, J. Sedano, Using Ensembles for Improving Fall Detection, in: Proceedings of the IV Jornadas de Fusión de la Información y Ensemble Learning, XVIII Conferencia de la Asociación Española para la Inteligencia Artificial, URL: https://pdfs.semanticscholar.org/1ea7/3dd7a497e03f6cf21eaf4774dd923a5830ce.pdf, 2019.

[25] H. Cao, S. Wu, Z. Zhou, C.-C. Lin, C.-Y. Yang, S.-T. Lee, C.-T. Wu, A fall detection method based on acceleration data and hidden Markov model, in: 2016 IEEE International Conference on Signal and Image Processing (ICSIP), IEEE, 684–689, ISBN 978-1-5090-2377-6, 2016.

[26] S. Yu, H. Chen, R.A. Brown, Hidden Markov Model-Based Fall Detection With Motion Sensor Orientation Calibration: A Case for Real-Life Home Monitoring, IEEE J. Biomed. Health Informatics 22 (6) (2018) 1847–1853, ISSN 2168–2194.

[27] C. Ronao, S.-B. Cho, Recognizing human activities from smartphone sensors using hierarchical continuous hidden Markov models, Int. J. Distrib. Sens. Networks 13 (2017) 1550147716683687.

[28] A.H. Fakhrulddin, X. Fei, H. Li, Convolutional neural networks (CNN) based human fall detection on Body Sensor Networks (BSN) sensor data, in: 2017 4th International Conference on Systems and Informatics, ICSAI 2017, vol. 2018, 1461–1465, ISBN 9781538611074, 2017.

[29] A. Lisowska, A. O'Neil, I. Poole, Cross-cohort evaluation of machine learning approaches to fall detection from accelerometer data, in: HEALTHINF 2018– 11th International Conference on Health Informatics, Proceedings; Part of 11th International Joint Conference on Biomedical Engineering Systems and Technologies, BIOSTEC 2018, vol. 5, 77–82, ISBN 9789897582813, 2018, doi:10.5220/0006554400770082.

[30] A.N. Aicha, G. Englebienne, K.S. van Schooten, M. Pijnappels, B. Kröse, Deep learning to predict falls in older adults based on daily-life trunk accelerometry, Sensors (Switzerland) 18 (5) (2018) 1–14, ISSN 14248220.

[31] G.L. Santos, P.T. Endo, K.H. d. C. Monteiro, E. d. S. Rocha, I. Silva, T. Lynn, Accelerometer-based human fall detection using convolutional neural networks, Sensors (Switzerland) 19 (7) (2019) 1–12, ISSN 14248220.

[32] E. Casilari, R. Lora-Rivera, F. García-Lagos, A Wearable Fall Detection System Using Deep Learning, in: 32nd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2019, 445–456, ISBN 9783030229986, 2019.

[33] T.T. Alemayoh, J.H. Lee, S. Okamoto, Deep Learning Based Real-time Daily Human Activity Recognition and Its Implementation in a Smartphone, in: 2019 16th International Conference on Ubiquitous Robots (UR), IEEE, 179–182, ISBN 9781728132327, 2019.

[34] A. Bevilacqua, K. MacDonald, A. Rangarej, V. Widjaya, B. Caulfield, T. Kechadi, Human activity recognition with convolutional neural networks, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 11053 LNAI (2019) 541–552, ISSN 16113349.

[35] A. Jahanjoo, M. Naderan, M.J. Rashti, Detection and multi-class classification of falling in elderly people by deep belief network algorithms, J. Ambient Intell. Humanized Comput. 2020 (0123456789), ISSN 18685145.

[36] J. O'Halloran, E. Curry, A comparison of deep learning models in human activity recognition and behavioural prediction on the mHealth dataset, in: 27th AIAI Irish Conference on Artificial Intelligence and Cognitive Science, vol. 2563, ISSN 16130073, 212–223, 2019.

[37] R. Sicoli, Z. Lin, Real-Time Human Activity Recognition using Convolutional Neural Network on Time Series from Mobile Sensors The Complete Dataset Stand Down Cycle Walk Jog Up Sit, URL: https://richardsicoli.github.io/, 2020.

[38] M. Zeng, L.T. Nguyen, B. Yu, O.J. Mengshoel, J. Zhu, P. Wu, J. Zhang, Convolutional Neural Networks for Human Activity Recognition using Mobile Sensors, in: 6th International Conference on Mobile Computing, Applications and Services, 197–205, 2019.

[39] T.R. Mauldin, M.E. Canby, V. Metsis, A.H. Ngu, C.C. Rivera, Smartfall: A smartwatch-based fall detection system using deep learning, Sensors (Switzerland) 18 (10) (2018) 1–19, ISSN 14248220.

[40] M. Musci, D. De Martini, N. Blago, T. Facchinetti, M. Piastra, Online Fall Detection using Recurrent Neural Networks, Tech. Rep. March 2019, Cornell University, URL: http://arxiv.org/abs/1804.04976, 2018.

[41] T. Theodoridis, V. Solachidis, N. Vretos, P. Daras, Human fall detection from acceleration measurements using a recurrent neural network, in: International Federation for Medical and Biological Engineering IFMBE Proceedings, vol. 66, 145–149, ISBN 9789811074189, ISSN 16800737, 2018.

[42] F. Luna-perejón, M.J. Domínguez-morales, Wearable Fall Detector Using Recurrent Neural Networks, Sensors (2019).

[43] I. Wayan Wiprayoga Wisesa, G. Mahardika, Fall detection algorithm based on accelerometer and gyroscope sensor data using Recurrent Neural Networks, in: IOP Conference Series: Earth and Environmental Science, vol. 258, ISSN 17551315, 2019.

[44] N. Gurov, A. Khan, R. Hussain, A. Khattak, Human Activity Recognition Using Deep Models and Its Analysis from Domain Adaptation Perspective, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 11771 LNCS (2019) 189–202, ISSN 16113349.

[45] M. Ullah, H. Ullah, S.D. Khan, F.A. Cheikh, Stacked LSTM network for Huma Activity Recognition using Smartphone data, in: 2019 8th European Workshop on Visual Information Processing (EUVIP), IEEE, ISBN 9781728144962, 175–180, 2019.

[46] E. García, J.R. Villar, E. de la Cal, Time Series data augmentation and dropout roles in Deep Learning applied to Fall Detection, in: accepted for publication in 15th International Conference on Soft Computing Models in Industrial and Environmental Applications SOCO 2020, 2020.

[47] K.S. Sczuka, L. Schwickert, C. Becker, J. Klenk, K.S. Sczuka, Re-Enactment as a Method to Reproduce Real-World Fall Events Using Inertial Sensor Data: Development and Usability Study Corresponding Author, J. Med. Internet Res. 22 (2020) 1–11.

[48] G.K. Palshikar, Simple Algorithms for Peak Detection in Time-Series, Tech. Rep., Tata Research Development and Design Centre, 2009.

[49] M. Villar, J.R. Villar, Peak detection enhancement in autonomous wearable Fall Detection, in: Proceedings of the 19th International Conference on Intelligent Systems Design and Applications (ISDA 2019), URL: http://www.mirlabs.net/isda19/program.php, 2019.

[50] J.R. Villar, M. Villar, E. de la Cal, M. Fáñez, J. Sedano, Fall detection based on local peaks and Machine Learning, in: accepted for publication in 15th International Conference on Hybrid Artificial Intelligent Systems HAIS2020, 2020.

[51] E. Casilari, J.A. Santoyo-Ramón, J.M. Cano-García, Analysis of public datasets for wearable fall detection systems, Sensors (Switzerland) 17 (7), ISSN 14248220.

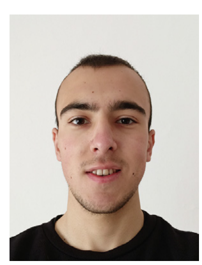
[52] S. Gasparrini, E. Cippitelli, E. Gambi, S. Spinsante, J. Wahslen, I. Orhan, T. Lindh, Proposal and Experimental Evaluation of Fall Detection Solution Based on Wearable and Depth Data Fusion, in: Proceedings of the ICT Innovations 2015, 99–108, 2015.

[53] A.T. Özdemir, B. Barshan, Detecting Falls with Wearable Sensors Using Machine Learning Techniques, Sensors 14 (2014) 10691–10708, ISSN 1424–8220.

[54] E. Casilari-pérez, R. Lora-Rivera, F. García-lagos, A Study on the Application of Convolutional Neural Networks to Fall Detection Evaluated with Multiple Public Datasets, Sensors 20 (2020) 1466.

[55] A.T. Ozdemir, An Analysis on Sensor Locations of the Human Body for Wearable Fall Detection Devices: Principles and Practice, Sensors 16 (2016) 1161.

Enol García Computer Science degree at University of Oviedo (2020), Enol is currently enrolled in the Artificial Intelligence MSc at Polytechnical University of Madrid. He has published several international conference papers, the main part of them related with fall detection. His research interest are Deep Neural Networks and their application in BioMedicine.



Mario Villar Computer Science student at University of Granada. He has published several conference papers related with fall detection. His research interests include Machine Learning and Artificial Intelligence applied to BioMedicine.

Mirko Fáñez Student of Computer Science at University of Oviedo, Spain. Mirko Fáñez is a Higher Technician in Multi-platform Applications Development (2016). Currently working on multiple Software Development projects at Instituto Tecnológico de Castilla y León (ITCL) in Burgos. He has been working on Fall Detection during the last year, developing Smartwatches software and application code. His current interest are Machine Learning and Applied Artificial Intelligence.

José R. Villar Professor of the University of Oviedo, Spain. In 1992 he received the B. Sc. Electronics and Control Engineering degree from University of Oviedo. In 2012, he received his Ph. D. in Computer Science from University of León (Spain). He has worked in several companies dealing with control and instrumentation projects. From 1998 to 2004 he was with the Electric and Electronic Department of University of León. Since then, he is a member of the Computer Science Department at University of Oviedo. His research topics includes Hybrid Algorithms applied to Real World Problems, Human Activity Recognition and Event Detection. He has published over 40 papers in JCR journals and more than 100 contributions to international conferences.

Enrique de la Cal Received the M.Sc. and Ph.D. degrees in Computer Science from the University of Oviedo, Spain, in 1995 and 2003, respectively. He has work as a pre-doctoral researcher in CSIC "Daza Valdes Research Institute". Currently, he is Professor with the Department of Computer Science of the University of Oviedo. His research interests are in the fields of Intelligent Sensors, Automatic Diseases Identification, Human Activity Recognition. He has been part of the scientific committee in several international conferences. He is co-author of more than 10 papers on impact factor journals and more than 30 contributions to international conferences.

Subg-Bae Cho He received the B.S. degree in computer science from Yonsei University, Seoul, Korea and the M. S. and Ph.D. degrees in computer science from KAIST (Korea Advanced Institute of Science and Technology), Taejeon, Korea. He was an Invited Researcher of Human Information Processing Research Laboratories at ATR (Advanced Telecommunications Research) Institute, Kyoto, Japan from 1993 to 1995, and a Visiting Scholar at University of New South Wales, Canberra, Australia in 1998. He was also a Visiting Professor at University of British Columbia, Vancouver, Canada from 2005 to 2006. Since 1995, he has been a Professor in the Department of Computer Science, Yonsei University. His research interests include neural networks, pattern recognition, intelligent man–machine interfaces, evolutionary computation, and artificial life. Dr. Cho was awarded outstanding paper prizes from the IEEE Korea Section in 1989 and 1992, and another one from the Korea Information Science Society in 1990. He was also the recipient of the Richard E. Merwin prize from the IEEE Computer Society in 1993. He was listed in Who's Who in Pattern Recognition from the International Association for Pattern Recognition in 1994, and received the best paper awards at International Conference on Soft Computing in 1996 and 1998. Also, he received the best paper award at World Automation Congress in 1998, and listed in Marquis Who's Who in Science and Engineering in 2000 and in Marquis Who's Who in the World in 2001. He is a Senior Member of IEEE and a Member of the Korea Information Science Society, INNS, the IEEE Computational Intelligence Society, and the IEEE Systems, Man, and Cybernetics Society.