An Automated Fall Detection System Using Recurrent Neural Networks

Francisco Luna-Perejon, Javier Civit-Masot, Isabel Amaya-Rodriguez, Lourdes Duran-Lopez, Juan Pedro Dominguez-Morales, Anton Civit-Balcells, and Alejandro Linares-Barranco

Robotics and Computer Technology Laboratory, University of Seville, 41012 Seville, Spain
fralunper@atc.us.es

Abstract. Falls are the most common cause of fatal injuries in elderly people, causing even death if there is no immediate assistance. Fall detection systems can be used to alert and request help when this type of accident happens. Certain types of these systems include wearable devices that analyze bio-medical signals from the person carrying it in real time. In this way, Deep Learning algorithms could automate and improve the detection of unintentional falls by analyzing these signals. These algorithms have proven to achieve high effectiveness with competitive performances in many classification problems. This work aims to study 16 Recurrent Neural Networks architectures (using Long Short-Term Memory and Gated Recurrent Units) for falls detection based on accelerometer data, reducing computational requirements of previous research. The architectures have been tested on a labeled version of the publicly available SisFall dataset, achieving a mean F1-score above 0.73 and improving state-of-the-art solutions in terms of network complexity.

Keywords: Fall detection · Deep Learning · Recurrent Neural Networks · Long Short-Term Memory · Gated Recurrent Units · Accelerometer

1 Introduction

According to the World Health Organization [12], unintentional falls are one of the most frequent causes of injuries in people over 65 years. Approximately 28%–35% of this cohort suffer at least one fall per year. This topic is gaining importance due to the progressive elderly population increase. Major injuries pose significant risk for postfall morbidity and mortality. This risk has been shown to be closely correlated to the delay in assist with first aid after the fall [7].

Fall detection systems (FDS) are devices that monitor the user's activity and ideally alert when a fall has occurred. They allow sending an accident notification immediately to medical entities, caregivers and family members for a quick assistance. Among all the different FDS types, wearable devices allow a continuous monitoring without dependence on the environment. This kind of devices usually use accelerometers and different algorithms to distinguish between daily activities and fall events [4]. Although threshold based algorithms show very high performance in terms of detection effectiveness and low computational complexity, they present many difficulties when trying to adapt them to new types of falls and user complexion. Machine learning methods are considered more sophisticated approaches to try to solve this problem. However, traditional supervised classification algorithms are not suitable due to the sequential nature of fall events, that implies a large computational cost, and the scarcity of datasets to study these events [5].

Recurrent Neural Networks (RNN) such as Long Short-Term Memory units (LSTM) and Gated Recurrent Units (GRU) are Deep Learning networks specifically designed to process sequences. Recent studies shed some light on the potential of RNNs for accelerometers [6]. This work focuses on finding a cost-effective RNN architecture in terms of computational complexity and effectiveness for fall detection in real-time.

2 Materials and Methods

2.1 Dataset

SisFall dataset [10] is used in this study. This dataset is composed of several simulated activities mainly classified in falls and activities of daily living (ADL). Each sample contains accelerometer measurements obtained from a device fixed to the user's waist. The measurement's sampling frequency is 200 Hz. The dataset was complemented with labeling criteria proposed in [6] dividing each activity into segments with a width of 256 samples and a stride of 128. Each segment can be classified as Fall, Alert or Background (FALL, ALERT and BKG) considering if that segment recorded part of a fall event, a fall hazard status or an ADL state without danger. A subset of 20% approximately (all activities related to 5 adult subjects and 3 elderly, randomly chosen from each category) was extracted from the total set for evaluating the effectiveness of models trained in this study. The dataset includes 94K samples for training (90K BCK, 1K ALERT, 3K FALL) and 23K for testing (22K BCK, 0.3K ALERT, 0.7K FALL).

2.2 Gated RNN

Gated RNNs introduce some memory-like cells in the architecture that hold information separated from the rest of the neural network. The information is managed through a set of gates. During the training of the network, the cells learn to close or open their gates according to the relevance of the information that comes from the sequence and the information currently stored.

LSTM units [3] contain three gates. Input and forget gates evaluate the addition of new information into memory and the deletion of part of the stored information. The output gate controls what information is provided to the next

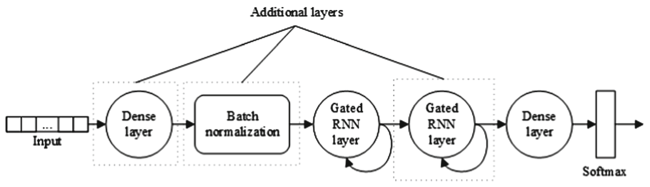

Fig. 1. Scheme representing all architectures trained in the first study stage, differentiated between them by having distinct combinations of the highlighted layers.

step. GRUs [1] are more recent cells similar to LSTM that lack of the output gate, dumping what is stored in the cell's memory during the entire training process. Both alternatives have shown to be similarly effective [2], although GRUs are more economical in terms of computation cost.

2.3 Training and Testing

This work aims to identify the most ideal RNN in order to be implemented in wearable devices. To this end, due to the high computational cost, the architecture should be simplified. Therefore, our study consists of two stages. First one is focused on comparing architectures with different layers combinations (See Fig. 1). The study includes 16 architectures (8 uses LSTM and 8 uses GRU). These are obtained by including/removing the dotted layers in Fig. 1, and are simplified versions of the proposed solution in [6] with the exception of number 8, that consist of the same architecture without dropout. Each architecture was tested by using both GRU and LSTM as RNN layers.

Due to the dataset being highly unbalanced, the overall classification accuracy is not an appropriate way to measure the effectiveness of the system. We compared the effectiveness employing the F1-score [8] for each class and average, which measures the relations between data's positive labels and those given by a classifier through a combination of precision and accuracy. Regarding the architecture complexity, the observed metric was the number of trainable parameters.

The second stage aimed to optimize the architectures that obtained the best results. Firstly, in order to deal with overfitting, dropout technique [9] was applied in the dense and recurrent layers, with the exception of the last dense used to classify the input. It was tested with 0% (without dropout), 20% and 35% values for each layer. Secondly, we used the best results combinations obtained previously to adjust batch size and learning rate hyperparameters by grid search with {32,64} and {0.0015, 0.001, 0.0005} values respectively.

3 Results and Discussion

Main results of first stage are presented in Table 1. F-1 score did not reach 0.33 for the ALERT class. Some reason for this can be the scarcity of this class examples in the dataset [5] and the falling conditions. The application of batch

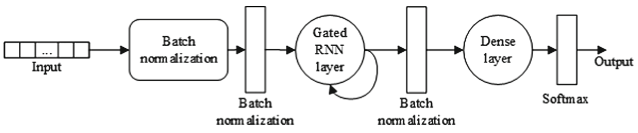

Fig. 2. Scheme representing the architecture with best results obtained in the first stage in terms of number of trainable parameters and mean F1-score.

Table 1. Results obtained with proposed architectures. First 8 rows corresponds to training results using LSTM as RNN layers and second ones using GRU. The *avg* subscript means the average between metrics values obtained. The *prec.* term consists of precision metric. Codes d1, bn and rnn2 indicates the presence in the architecture of first dense layer, batch normalization and second RNN layer respectively.

	Additional layers	Loss	$F1_{BKG}$	$F1_{ALE}$	$F1_{FALL}$	$Prec._{avg}$	$Recall_{avg}$	$F1_{avg}$	Param.
1	-	1.83	0.88	0.09	0.58	0.49	0.76	0.52	4835
2	rnn2	1.56	0.94	0.21	0.69	0.55	0.87	0.61	13283
3	**bn**	**1.04**	**0.95**	**0.22**	**0.83**	**0.61**	**0.91**	**0.66**	**4847**
4	bn, rnn2	1.02	0.96	0.29	0.82	0.63	0.91	0.69	13295
5	d1	2.07	0.85	0.08	0.45	0.45	0.73	0.46	8675
6	d1, rnn2	1.37	0.94	0.20	0.71	0.56	0.86	0.62	17123
7	d1, bn	0.97	0.95	0.25	0.82	0.62	0.91	0.67	8803
8	d1, bn, rnn2 [6]	1.31	0.96	0.26	0.82	0.63	0.89	0.68	17251
9	-	1.75	0.87	0.09	0.64	0.51	0.78	0.53	3651
10	rnn2	1.63	0.95	0.23	0.74	0.58	0.85	0.64	9987
11	**bn**	**1.10**	**0.97**	**0.32**	**0.82**	**0.64**	**0.90**	**0.70**	**3663**
12	bn, rnn2	1.20	0.96	0.29	0.80	0.62	0.90	0.69	9999
13	d1	1.89	0.84	0.09	0.43	0.44	0.75	0.45	6563
14	d1, rnn2	1.11	0.94	0.21	0.71	0.56	0.89	0.62	12899
15	d1, bn	0.97	0.96	0.28	0.87	0.65	0.92	0.70	6691
16	d1, bn, rnn2	1.28	0.97	0.34	0.82	0.64	0.90	0.71	13027

normalization presents the best contribution to performance results. Although the second RNN layer improves the results, the amount of parameters that it adds to the network implies a higher computational cost. Due to the low number of parameters and good results in comparison with the rest of the models, it was considered to optimize results of both architectures 3 and 11 (Fig. 2).

Table 2 shows the best results obtained after the parameter optimization. Effectiveness results are very similar to [6], of which is estimated an F1-score of 0.71 considering that they had used approximately the same number of samples in each class as in this study. However, the model obtained in this study has a much lower complexity since it lacks a dense layer and an additional LSTM layer. This implies a direct impact on the energy saving of the real-time capable device in which the model would be integrated, being able to have an autonomy greater than 20 h of [11].

Table 2. Results obtained after grid search optimization.

RNN type	Learn. rate	Batch size	Input drop	RNN drop	Param	$F1_{BKG}$	$F1_{ALE}$	$F1_{FALL}$	$Prec_{avg}$	$Recall_{avg}$	$F1_{avg}$
lstm	0.001	32	0.35	0	4847	0.98	0.42	0.85	0.69	0.88	0.75
gru	0.001	64	0.35	0.2	3663	0.98	0.37	0.85	0.69	0.88	0.73

4 Conclusions

In this paper, we present the use of Gated RNN based in LSTM and GRU layers as a method implementable in wearable devices with accelerometer to detect falls in real-time. After a study of 16 architectures, the best results in terms of computational complexity and classification are formed by a batch normalization layer receiving the input, a RNN layer and a dense end layer for the classification of the event. FALL and BKG classes are well classified, with F1-scores above 0.98 and 0.85 respectively. Mean F1-score obtained was 0.75 and 0.73 for LSTM and GRU versions, respectively. In future studies the authors will implement this model in a hardware device suitable for use as a wearable FDS.

Acknowledgment. This work was supported by the excellence project from the Spanish government grant (with support from the European Regional Development Fund) COFNET (TEC2016-77785-P).

References

1. Cho, K., et al.: On the properties of neural machine translation: encoder-decoder approaches. arXiv preprint arXiv:1409.1259 (2014)
2. Chung, J., et al.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
3. Hochreiter, S., et al.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
4. Igual, R., et al.: Challenges, issues and trends in fall detection systems. Biomed. Eng. Online **12**(1), 66 (2013)
5. Khan, S.S., et al.: Review of fall detection techniques: a data availability perspective. Med. Eng. Phys. **39**, 12–22 (2017)
6. Musci, M., et al.: Online fall detection using recurrent neural networks. arXiv preprint arXiv:1804.04976 (2018)
7. Noury, N., et al.: Fall detection-principles and methods. In: 2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS 2007, pp. 1663–1666. IEEE (2007)
8. Sokolova, M., et al.: A systematic analysis of performance measures for classification tasks. Inf. Process. Manag. **45**(4), 427–437 (2009)
9. Srivastava, N., et al.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
10. Sucerquia, A., et al.: SisFall: a fall and movement dataset. Sensors **17**(1), 198 (2017)

11. Torti, E., et al.: Embedded real-time fall detection with deep learning on wearable devices. In: 2018 21st Euromicro Conference on Digital System Design (DSD), pp. 405–412. IEEE (2018)
12. World Health Organization: Ageing; life course unit. Who global report on falls prevention in older age. World Health Organization (2008)