Transfer learning approach for fall detection with the FARSEEING real-world dataset and simulated falls

Joana Silva¹, Inês Sousa¹ and Jaime Cardoso²

Abstract—Falls are very rare and extremely difficult to acquire in free living conditions. Due to this, most of prior work on fall detection has focused on simulated datasets acquired in scenarios that mimic the real-world context, however, the validation of systems trained with simulated falls remains unclear. This work presents a transfer learning approach for combining a dataset of simulated falls and non-falls, obtained from young volunteers, with the real-world FARSEEING dataset, in order to train a set of supervised classifiers for discriminating between falls and non-falls events. The objective is to analyze if a combination of simulated and real falls could enrich the model. In the real-world, falls are a sporadic event, which results in imbalanced datasets. In this work, several methods for imbalance learning were employed: SMOTE, Balance Cascade and Ranking models. The Balance Cascade obtained less misclassifications in the validation set. There was an improvement when mixing the real falls and simulated non-falls compared to the case when only simulated falls were used for training. When testing with a mixed set with real falls and simulated non-falls, it is even more important to train with a mixed set. Moreover, it was possible to conclude that a model trained with simulated falls generalize better when tested with real falls, than the opposite. The overall accuracy obtained for the combination of different datasets were above 95%.

I. INTRODUCTION

Falls are one of the main causes of hospitalization and loss of independence in the elderly population. Prompt assistance could decrease the negative effects of a fall event. Automatic fall detection systems have been developed in the past years and rely mostly on wearable or smartphones with integrated inertial sensors and location capabilities that facilitate the detection and triggering of a fall alert. Prior reported methods used accelerometer signal processing for the development of a supervised algorithm for fall detection with a dataset of simulated falls and activities of daily living (ADLs), considered as non-falls, acquired from young volunteers [1]. The occurrence of a fall event is very rare, compared to the amount of daily living activities, and the annotation process involved in creating a real-world falls dataset is very time and resources consuming, which makes the availability of such datasets extremely scarce. Even though, there are some research groups that have managed to collect data from real-world falls [2].

This paper describes an approach based on a dataset of accelerometer readings from simulated falls and daily living activities recorded by Fraunhofer Portugal AICOS (FhP), in laboratory facilities, and a real-world dataset provided by FARSEEING project, with 23 examples of real falls acquired from elderly patients in hospital settings. The research objective is to train a supervised model capable of correctly identifying fall events, with a minimum number of false positives during activities of daily living in real-world conditions. A transfer learning mechanism was employed in order to train a model that could be both validated with real and simulated falls and activities of daily living, in order to extend the validation of supervised models to real-world scenarios. Window-based methods were used to segment the accelerometer signals into falls and non-falls events, for the implementation of feature extraction methods. In total, 16 temporal features were extracted for each time-window in a dataset with 23 real falls, 650 simulated falls, 877 real-world non-falls and 410 simulated non-falls events.

Fall detection has been tackled with several approaches: cameras, floor pressure sensors, infrared sensors, inertial sensors, heart rate sensors and microphones. According to [3], fall detectors can broadly be divided into context-aware systems and wearable devices. Most of the approaches based on wearable devices rely on inertial sensors to discriminate between falls and ADLs. Moreover, a high percentage of these studies have used datasets of simulated falls to develop and validate the fall detector. Previous studies on this topic have reported the comparison between simulated and real-world falls, as the work of [4] and [5]. The work of [6] has also focused on the comparison between accidental falls in elderly and simulated falls of younger volunteers. Both studies have highlighted that the limitations of fall simulations should be taken into consideration and the protocol should be adapted to better match real-world falls. The fall detectors should consider the imbalance in real and simulated datasets, by employing imbalance learning methods on the trainset and evaluate the trained models on imbalance conditions that mimic real-world scenarios. Machine learning approaches have successful been applied to discriminate between falls and ADLs, based on inertial sensors data. High sensitivities and specificities have been achieved for simulated falls datasets, however the validation of these approaches has been questioned for real-world conditions [7][6]. More recently, [8] has reported that a decision tree classifier trained with real-world falls is capable of discriminate between falls from ADLs with accuracies comparable with previous studies that have used simulated falls datasets. However, an hybrid
approach that considers both simulated and real-world falls for the development and validation of a fall detector could be of utmost interest, given that real world falls datasets are rare and difficult to achieve, the limited number of available samples could impact the development of machine learning algorithms. Given that young volunteers have potentially more false positives than more sedentary older volunteers, non-fall events acquired from young volunteers should also be incorporated in the validation datasets. The FARSEEING real-world dataset was acquired from hospitalized patients, which may not be representative of a broad population. This way, the combination of this real-world dataset with simulated non-falls events could enrich the validation set, since more diversity data is used to validate the models. The naive use of real data in models learned from simulated data may face difficulties due to the differences between both settings. However, the use of domain adaptation/transfer learning techniques has the potential to leverage the benefits of both.

II. DATA ACQUISITION AND PROCESSING

A. Simulated falls dataset

Simulated falls and non-falls were collected using a smartphone inside the trousers’ pocket, following the protocol described by Noury [9], adapted with additional non-fall activities. The dataset was collected from 7 volunteers that performed 8 types of falls (backward, forward, lateral falls) and 8 types of ADLs (sit-to-stand and stand-to-lay transitions, walk, run, bend, drop phone on a table, walk and sit, sit with rotations), repeated three times [10]. This dataset comprises 650 falls and 410 non-falls.

B. FARSEEING real-world fall database

In the scope of the FARSEEING project, a dataset with real falls were recorded from three different wearable devices: MiniMod and Hybrid were located in the lower back, sampled at 100Hz, and ActivPAL3 was placed on the thigh, sampled at 20Hz. The authors made available 22 accelerometer files from a set of 100 files and 1908 sequences of ADLs [2]. Each person, of a group of 15 persons, used a wearable device to collect falls and sequences of ADLs, at the Geriatric Rehabilitation Unit in Robert Bosch Hospital, Stuttgart. The available dataset comprises 22 files with 20 minutes of wearable sensor data. From these files, 21 have only one fall event and one file has two fall events (with a short duration of approximately 10 seconds). The sensor data prior to the fall event were considered as non-fall events.

1) Resampling: From the 22 samples, 7 were collected at 20Hz and the remaining were collected at 100Hz. Original files were resampled in order to uniformize the sampling frequency to 100Hz. First, the timestamp was converted to seconds: an artificial timestamp was created for all files in order to have unique timestamps. Second, the 20Hz files were upsampled to 100Hz to ensure concordance with the remaining of the dataset. The method used to upsample was based on replicating samples in order to have 100 samples in each second elapsed.

2) Segmentation: Centered in the timestamp of the fall (previously annotated by the FARSEEING group), a window with 7.5 seconds was defined as the fall period. The non-fall period was considered as the period until 10 seconds before the fall timestamp, yielding a total of 9 minutes and 50 seconds for the non-fall period. Then, this period was sequentially divided into 7.5 seconds windows. Since the amount of samples of the non-fall period is considerably higher than the amount of samples in the fall period, several imbalance approaches were used to overcome this difference.

C. Comparison between datasets

Two datasets were used for the development and validation of fall detection approaches based on machine learning techniques: a simulated dataset acquired by FhP’s young volunteers in simulated conditions and a set of FARSEEING real-world falls acquired with elderly patients in a hospital. The major differences between the two datasets are: different inertial sensors placement (simulated falls were acquired with the smartphone on the trousers’ pocket and real falls were acquired with wearable sensors on the lower back and thigh), different sampling rates (simulated falls were acquired at 100Hz and some real falls were acquired at 20Hz), different inertial sensors specifications concerning the accelerometer amplitude range (certain sensors involved in the real-world data collection were limited to 2G range). In addition, the context of the fall is different: the elderly does not stand in front of the mattress to fall as in simulated conditions. The acceleration of the fall impact is also different since the impact is on the ground and not on the mattress.

III. MACHINE LEARNING PIPELINE

A. Pipeline overview

Using Python’s scikit-learn package (v.0.19.1), a classification pipeline was designed:

1) Input signal: the accelerometer magnitude was computed, at 100Hz. The signal was divided into windows with 7.5 seconds (750 samples), without overlap. Windows with a low signal standard deviation were removed, in order to discard samples where the signal was mainly stationary, and were considered useless for the train and test sets.

2) Feature extraction: a set of time-domain features were extracted for each time-window and include mean, standard deviation, median, median deviation, maximum (max), minimum, energy, root mean square (rms), inter quartile range (iqr), histogram (10 bins), skewness and kurtosis [10].

3) Feature selection: features with correlation higher than 0.90 were removed. The removed features were energy, iqr, max, median, rms, yielding a feature vector with 16 features.

4) Feature standardization: features were standardized by removing the mean and scaling to unit variance, in order to ensure that all features are standard normally distributed.

5) Train and test split: for the real-world dataset, there were 23 falls and 877 non-falls (after removing stationary windows). The stratified split train and test, with 30% test size was used, which yield a train set with 630 samples (14 falls and 616 non falls) and a validation test with 270 samples (9 Falls and 261 non falls).
6) **Supervised classifiers**: Nearest Neighbors, Decision Tree, Random Forest, Multi-layer Perceptron and AdaBoost.

7) **Hyper-parameters**: of each classifier were optimized, within a given interval, using Grid Search with 10 folds cross-validation, for the train set (630 samples). The stratified split train and test, with 30% test size, was once more applied in order to divide the train set into train (9 falls and 413 non falls) and test (5 falls and 203 non falls) sets, for hyperparameters tuning. This process was repeated 40 times to ensure statistical significance.

**B. Imbalance learning**

A dataset is considered imbalanced when the classification classes are not approximately equally represented. Often real-world datasets are mainly composed of normal events with only a small percentage of abnormal or of interest events [11]. For the real-world fall dataset, the amount of non-fall events is considerably higher than the amount of fall events (due to its rare and sudden nature). To overcome the class imbalance, several approaches were considered, in order to ensure that the imbalance dataset does not have an impact on the classifiers performance:

1) **Synthetic Minority Over-sampling Technique** (SMOTE) [11] was used to oversample real-world samples in the train set. Using this approach, the minority class is oversampled by creating synthetic examples rather than by oversampling with replacement.

2) **Balance Cascade**: creates an ensemble of balanced sets by iteratively undersample the imbalanced dataset using an estimator [12]. SMOTE and Balance Cascade are implemented in Python’s imbalanced-learn (v0.2.1) package [13].

3) **Ranking Models**: were used for tackling class imbalance with ranking. Models tested with features extracted from real-world falls were: Adaboost, Balanced linear SVC, Linear SVC, Rankboost and Rank SVM [14].

**C. Transfer learning**

We propose the use of domain adaptation/transfer learning techniques to cope with differences between simulated and real falls datasets. For means of comparison with a setup that only uses the real-world dataset, an initial test was made for train with real samples and test with real samples.

Given that the real-world falls dataset is highly imbalanced, 23 falls and 877 non falls, imbalance methods were also applied in order to overcome this disproportion, namely the SMOTE algorithm. Moreover, given that the real-world dataset was collected with hospitalized patients, it could be expected that the activities the users undertook do not include high accelerometer variations, as expected when running or jumping or other activities with higher impacts. Due to this, a dataset of simulated non-falls, that includes samples with high accelerometer variations, were mixed with the real-world dataset (mixed set) in order to challenge the train and test sets (case 1, 2 and 3). This approach was evaluated for three different cases: 1) train with a mixed set and test with real samples and 2) train with real samples and test with mixed set and 3) train with mixed set and test with the remaining mixed set. These tests were meant to analyze the number of false positives achieved on a mixed set when the model is trained only with real samples (case 2) or when the model is trained with a mixed set (case 3). The opposite was also tested in order to assess the added value of training a model with a mixed set (case 1) comparatively to train only with the real-world dataset (case 6). The latter two cases were also compared with the case of training a model exclusively with simulated falls and non-falls and test it with real-world samples (case 4) and the opposite (case 5).

**IV. RESULTS**

**A. Imbalance learning**

1) **SMOTE**: was applied for each classifier, using a stratified train and test split for 40 splits. Overall, the SMOTE algorithm obtained high accuracy for the set of five classifiers tested using a balanced train set with 413 falls and 413 non-falls. When evaluated in an imbalanced validation set, the MLP classifier had the higher area under the curve and achieved only one false positive and one false negative.

2) **Balance Cascade**: divided the dataset into 53 sets and the classifiers were trained for each set. The model with the higher area under the curve was the MLP Classifier. Using the Balance Cascade, the accuracy with the training set was also very high. Despite for the Random Forest, all remaining classifiers obtained an accuracy, precision and recall above 90%. For the imbalanced validation set, the best trained model obtained only one false positive.

3) **Ranking Models**: were trained for 10 sets using the stratified split. The trained model with highest area under the curve was tested with the validation set. The results obtained for the ranking models were even higher than for the latter cases, however when tested with the validation set, the best model obtained a higher number of false positives (six non-falls were misclassified as falls).

**B. Transfer learning**

For the cases 1 and 6 (Table 1), when testing only with real data, if we include simulated non-falls in the training (case 1), the number of false negatives (FN, actual falls predicted as non falls) will not change, as expected, however the number of false positives (FP, actual non falls predicted as falls) will increase, because the train set has more variability and also more samples. Even though, this model is expected to be more robust against potential non-falls events, since it was trained with non-falls simulated in conditions that usually trigger more false positives. Comparing these two cases with case 4, when training only with simulated falls and testing with real falls, there is a improvement when mixing the real and simulated falls (case 1) compared to the case when only simulated falls were used for training (case 4).

For the cases 2 and 3, when testing with a mixed set with real falls and non-falls and simulated non-falls, it is indeed more important to train with a mixed set (case 3) than train only with real data (case 2), because less false positives were found. However, this model is also more prone to false negatives. In order to conclude which combination is better,
it should be considered the cost of misclassifying a fall and the cost of detecting a false fall.

Comparing case 4 and 5, it was possible to conclude that a model trained with simulated falls (Simul.) generalizes better when tested with real falls (case 4), than the opposite (case 5), a model trained with real falls and tested with simulated falls. Moreover, these two cases, that do not include any mixed samples, were the ones with lower accuracy (Acc) (expecting for case 6), highlighting the fact the mixing real and simulated non-falls improves the results.

<table>
<thead>
<tr>
<th></th>
<th>Train</th>
<th>Test</th>
<th>Acc</th>
<th>Prec</th>
<th>Rec</th>
<th>FP</th>
<th>FN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mixed</td>
<td>Real</td>
<td>96</td>
<td>97</td>
<td>99</td>
<td>96</td>
<td>21</td>
</tr>
<tr>
<td>2</td>
<td>Real</td>
<td>Mixed</td>
<td>97</td>
<td>98</td>
<td>99</td>
<td>97</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>Mixed</td>
<td>Mixed</td>
<td>99</td>
<td>97</td>
<td>99</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>Simul.</td>
<td>Real</td>
<td>92</td>
<td>91</td>
<td>98</td>
<td>92</td>
<td>144</td>
</tr>
<tr>
<td>5</td>
<td>Real</td>
<td>Simul.</td>
<td>71</td>
<td>66</td>
<td>71</td>
<td>71</td>
<td>73</td>
</tr>
<tr>
<td>6</td>
<td>Real</td>
<td>Real</td>
<td>99</td>
<td>99</td>
<td>100</td>
<td>99</td>
<td>3</td>
</tr>
</tbody>
</table>

**TABLE I**

TRANSFER LEARNING RESULTS FOR THE COMBINATION OF SIMULATED AND REAL FALLS.

V. CONCLUSIONS

Detecting a fall in non-restricted nor simulated scenarios has been accomplished in most of the past works using wearable inertial sensors. Comparatively to camera-based approaches, wearable sensors avoid the need for environment adaptations and fixed placement, allowing the monitored device to follow the user continuously. Since fall events are very rare and extremely difficult to acquire in free living conditions, most of the prior work has focused on simulated datasets acquired in scenarios that mimic the real context. Even though, the validation of the systems that were trained with simulated falls remains unclear. This approach presented for combining two datasets: one with real falls and non-falls, from the FARSEEING real-world dataset, and another with simulated falls and non fall events, acquired with younger and more active volunteers. Both datasets were used with different combinations for training and validation, in order to obtain a fitted supervised classifier that better generalizes to new fall events. In the real-world, falls are a sporadic and rare event, which results in imbalanced datasets. In this work, several methods for imbalance learning were employed, to deal with this dataset, namely SMOTE algorithm, Balance Cascade and Ranking Models. Among the three approaches, the accuracies obtained for a set of different classifiers were very high, but the Balance Cascade obtained less misclassifications in the validation set. Combined sets of simulated and real falls presented advantages comparatively to using only simulated falls. There is an improvement when mixing real falls and simulated non-falls compared to the case when only simulated falls were used for training. When testing with a mixed set containing real falls and simulated non-falls, it is indeed more important to train with a mixed set. Moreover, it was possible to conclude that a model trained with simulated falls generalize better when tested with real falls, than the opposite. Compared to previous works that have used the FRASEEING real-world dataset, the results obtained with this approach overcome the ones obtained by [8] using a decision tree classifier by 10%. Moreover, few samples of falls were used in this work for train and validation, highlighting the need to employ imbalance learning and transfer learning.

ACKNOWLEDGMENT

We thank all participating men and women in the FARSEEING project, as well as all FARSEEING research scientists, study and data managers and clinical and administrative staff who make the study possible.

REFERENCES


