Med Biol Eng Comput
DOI 10.1007/s11517-017-1632-z

ORIGINAL ARTICLE

Combining novelty detectors to improve accelerometer-based fall detection

Carlos Medrano[1] · Raúl Igual[1] · Iván García-Magariño[1] · Inmaculada Plaza[1] · Guillermo Azuara[2]

Received: 25 July 2016 / Accepted: 21 February 2017

Abstract Research on body-worn sensors has shown how they can be used for the detection of falls in the elderly, which is a relevant health problem. However, most systems are trained with simulated falls, which differ from those of the target population. In this paper, we tackle the problem of fall detection using a combination of novelty detectors. A novelty detector can be trained only with activities of daily life (ADL), which are true movements recorded in real life. In addition, they allow adapting the system to new users, by recording new movements and retraining the system. The combination of several detectors and features enhances performance. The proposed approach has been compared with a traditional supervised algorithm, a support vector machine, which is trained with both falls and ADL. The combination of novelty detectors shows better performance in a typical cross-validation test and in an experiment that mimics the effect of personalizing the classifiers. The results indicate that it is possible to build a reliable fall detector based only on ADL.

Keywords Fall detection · *P*attern recognition · Mobility · eHealth

✉ Raúl Igual
rigual@unizar.es

[1] Instituto de Investigación Sanitaria Aragón / Department of Electronics Engineering and Communications, Department of Electrical Engineering / EduQTech, EUPT, University of Zaragoza, c/Atarazana 2, 44003 Teruel, Spain

[2] Department of Electronics Engineering and Communications/ CeNITEQ, EUPT, University of Zaragoza, c/Atarazana 2, 44003 Teruel, Spain

1 Introduction

Falls among older adults are an important public health problem. This problem will be aggravated in next years since the percentage of population aged 65 and older continues to increase in many countries. The detection of falls is necessary in the mobility telemonitoring of the elderly in their living environments. Real-time detection of falls would probably reduce their consequences with a quick intervention. However, detection of falls is a challenge. Despite many years of research, there is no widely accepted technique for fall detection and the issue remains open.

There are several techniques for fall detection [12, 24] that can be classified depending on the sensor type: sensors deployed in the environment or body-worn sensors [7, 11, 19, 29]. The current paper focuses on accelerometers worn by the user. The pioneering studies used simple algorithms based on a threshold on the acceleration peak or valley like in [5]. Other features deduced from acceleration can help to decrease the number of false alarms: orientation and its change, velocity or distance [6, 15]. In [15], the best algorithm included acceleration peak detection followed by posture detection. It was concluded that velocity did not improve the results. However, in [6], it was concluded that taking velocity into account decreased the number of false positives in real conditions. In [26], a threshold-based algorithm was proposed but with a more complex pre-processing. First, an averaged shape during falls was obtained. It served as a mother wavelet, from which a coefficient of similarity could be found for any new shape. The system outperformed detectors based on raw peak values. Other studies have tried to solve the problem using more elaborate pattern recognition methods, like neural networks, support vector machine (SVM) or a combination

Published online: 01 March 2017

of AdaBoost and SVM. In [16], a windowing technique was used to extract features (1-s sliding window with 0.5-s overlapping). Several features were extracted up to a total of 28 (acceleration, velocity, position and different time domain features). Six classifiers were compared and the multilayer perceptron outperformed others like SVM, naive Bayes or OneR. Moreover, up to 178 features from time windows of 10 s were extracted in [1]. Several classifiers were compared, with SVM being the best one. The system detected also the type of fall. In [9] an improvement over classical AdaBoost classifiers was proposed by substituting some stages of weak classifiers by SVM. The authors used directly the acceleration values and the signal magnitude area as features. In [3], five threshold-based algorithms and five machine learning algorithms were compared. The overall performance of the machine learning algorithms was greater. In particular, SVM achieved the highest combination of sensitivity and specificity.

Besides finding a proper fall detection technique, research on fall detection faces the lack of enough recorded real falls. For instance, in [4] several algorithms were tested with real data, including 29 falls from people suffering progressive supra-nuclear palsy. The results were clear: the performance of several algorithms decreased dramatically in real life, while they were near perfection in the original studies. In this regard, an interesting approach deviates from traditional supervised classifiers: instead of training a classical classifier, a model of normal activities can be built, since activities of daily living (ADL) can be easily recorded in real-life conditions, providing a large number of exemplars. A fall is detected as a movement that is rare and anomalous with respect to that model. Besides, as a new user carries the detector, it is possible to record new movements and retrain the model. In this way, the detector can be personalized. For instance, in [25] a time-varying linear prediction model for the acceleration signal was built. The decision was based on a hypothesis testing approach, being the null hypothesis (H_o) that the current acceleration sample comes from ADL. The authors of [20] compared several novelty detectors to check the shape of acceleration around acceleration peaks. Detectors based on a nearest neighbour rule were the best. However, an SVM trained with ADL and falls outperformed a nearest neighbour (NN) trained only with ADL in most situations. Only if the device was worn in a body location different from training conditions, NN was superior to SVM. This work was extended in [22] to study the effect of personalization. Although NN is very easy to personalize, SVM can also be partially personalized, achieving still higher performance than NN. In this regard, Micucci et al. [23] compared one-class and two-class configurations and showed that a one-class NN achieved almost the same performance as a two-class SVM using the raw values of the acceleration in a 1-s time window. Overall, it was concluded that falls are not required to design an effective fall detector. In [17] an approach based on a modified Hidden Markov Model (X-Factor HMM) was proposed. This HMM had inflated output covariances, obtained from some outliers of the ADL that served as proxies for the unseen falls. It was shown that a traditional HMM trained with all the ADL did not produce consistent results for outlier detection, while the X-Factor HMM showed good performance but not always higher than an HMM trained with ADL and falls.

This paper presents an original approach to fall detection that is based on a combination of novelty detectors. Each novelty detector takes into account a different feature extracted from acceleration time series. Then, novelty detectors have been combined using a principled approach, which is the key contribution of this work with respect to previous studies in the literature. To check the proposed approach, the combination of novelty detectors has been compared with a traditional supervised SVM (trained with ADL and falls), both in a typical cross-validation experiment and in an experiment simulating several degrees of system personalization. Novelty detectors have been selected because they present two key advantages: they are based only on real data (ADL) and they allow an easy adaptation to different circumstances by recording new data and being retrained on the fly.

2 Methods

In this section, the data set and feature extraction are explained first. Then, several concepts are briefly reviewed in order to build, step by step, the detector proposed in this paper. Thus, the concept of novelty detector is presented focusing on the techniques utilized in this paper. This leads to the posterior problem of combining several detectors, where the solutions selected are explained. Then, a traditional algorithm, SVM, is briefly introduced as the system against which the combination of novelty detectors was compared. For SVM, the use of different features has also been tested. Afterwards, the performance figure of merit is explained. The section ends with a sequence of the experiments performed, directly related to the results presented in the next section.

2.1 Data set

For the evaluation of the algorithms, we used a publicly available data set [20]. This data set has a large number of ADL (7816) and falls (503). It is suitable to build a novelty detector since there are many examples of normal movements recorded in real life. In addition, it has been shown to generalize better than other data sets [13]. It includes data from 10 subjects, ranging from 20 to 42 years old. The data set consists of time windows of 6 s. Each window has 301 three-axis acceleration samples at 50 Hz. The recording system used to extract the data was such that there is always a peak of the acceleration vector norm in the centre of the window. In this paper, the peak is used as the reference time of the window, that is, negative and positives times are associated with samples before and after the peak, respectively.

2.2 Feature extraction

For the novelty detectors, we have used four features: the distance to the nearest neighbour record (d_{NN}), the change in orientation (Δ), the final velocity after the fall (v_f) and the distance of the body's displacement during the fall (D). These values were obtained as follows:

- *Nearest neighbour distance* (d_{NN}) Given acceleration records $\boldsymbol{A} = \{\boldsymbol{a}_i,\ i = 1 \ldots T\}$ and $\boldsymbol{B} = \{\boldsymbol{b}_i,\ i = 1 \ldots T\}$, where i is the time index, and T defines the length of the window (from −0.5 to 0.5 s), the distance between two records was defined as:

$$d(\boldsymbol{A},\boldsymbol{B}) = \sqrt{\sum_i \|\boldsymbol{a}_i - \boldsymbol{b}_i\|^2} \tag{1}$$

Then, given a set of N exemplars from a training set $\{\boldsymbol{B}^j,\ j = 1 \ldots N\}$, the nearest neighbour distance of a new acceleration record, $\boldsymbol{C}$, was defined as:

$$d_{NN}(\boldsymbol{C}) = \min_j d(\boldsymbol{C},\boldsymbol{B}^j) \tag{2}$$

d_{NN} is a measure of how far is the acceleration shape around the peak from a set of exemplars.

- *Change in orientation* (Δ) It was measured from the gravity component of the acceleration, by calculating the cosine of the angle between the gravity at times −1.5 and 3 s. To get gravity (g_i), the initial acceleration (a_i) was low-pass filtered.
- *Final velocity* (v_f) The effect of gravity was removed to obtain the linear acceleration: $\boldsymbol{a}_{l,i} = \boldsymbol{a}_i - \boldsymbol{g}_i$. Then, a three-axis velocity was obtained by integrating the three-axis linear acceleration in the time interval from −0.5 to 0.5 s. The final velocity (v_f) is the module of the vector obtained in this way at 0.5 s.
- *Distance of the body's displacement during the fall (D)*: In the next step, the distance was obtained by integrating the module of the velocity from −0.5 to 0.5 s. This physical distance (D, in metres) is not to be confused with the distance between two acceleration shapes ($d(\boldsymbol{A},\boldsymbol{B})$ and $d_{NN}(\boldsymbol{C})$, in m/s^2).

2.3 Novelty detectors

Fall detection can be casted as a novelty (or anomaly) detection problem [8, 20], in which the system has to discriminate between normal records (ADL) and outliers (falls in our case). The classification problem is more complex due to the absence of a large data set of outliers. Since falls are always simulated, the current paper focuses on the extreme case of having no outliers for training, which is in fact the real scenario since there are no real fall data sets. Thus, the system models only the behaviour of the normal class (ADL).

In the present paper, a single novelty detector was obtained by first modelling the probability density of a given feature, $p(x_i)$, and then thresholding it. Then, for an acceleration time window centred in the peak, the feature x_i was extracted and it was considered that a fall had occurred if:

$$p(x_i) < \theta_i \tag{3}$$

where x_i can be any of the features considered in this paper: $i = d_{NN}$, Δ, v_f or D.

To model densities, the technique of kernel density estimation (KDE) [2] was selected. KDE provides a continuous model. The probability of a feature x_i is written as:

$$p(x_i) = \frac{1}{N\sigma_i} \sum_{j=1}^{N} k\left(\frac{x_i - x_i^j}{\sigma_i}\right) \tag{4}$$

where x_i^j are the training exemplars values, k is the kernel function and σ_i is the bandwidth. In this paper, a Gaussian kernel was selected and the free parameters were σ_i, $i = d_{NN}$, Δ, v_f or D.

2.4 Combination of novelty detectors

Combining classifiers is a common alternative to capture more characteristics of the data. However, novelty detectors considered in this paper have some specific properties. Since information on the outlier data is not available, their "training" neither provides directly a way to classify new data nor models posterior probabilities. This precludes the direct use of techniques such as the majority vote or the naive Bayes combination [18]. In [30] the specific problem of combining one-class classifiers was considered. In this paper, two combination rules proposed in [30] have been selected in order to cover two approaches: one based on probabilities and another based on output labels. First, the product combination rule was followed, that is, the probability outputs of the individual classifiers were multiplied and the decision depended on a threshold on that product ($y_{pp}(x) < \theta_{pp}$), with:

$$y_{pp}(x) = \prod_{i=1}^{N_f} p(x_i) \tag{5}$$

where N_f is the number of features considered in the model. This rule has the advantage of not relying on any parameter (other than the global threshold θ_{pp}). It makes use of the probabilities themselves.

Another possibility is to transform probabilities into labels and then use the mean vote:

$$y_{mv}(x) = \frac{1}{N_f} \sum_{i=1}^{N_f} I(p(x_i) \leq \theta_i) \tag{6}$$

where θ_i is a threshold for method i and $I()$ is the indicator function, which gives 1 if x_i fulfils the condition or 0 otherwise. In this case, several parameters have to be set. The threshold for $y_{mv}(x)$ itself was set to 0.5. Thus, the mean vote was transformed into a majority vote (ties were resolved at random if N_f was even). With respect to θ_i, they were selected so that the fraction of the training set accepted as being ADL was the same, f_T, for any of the individual classifiers:

$$\int I(p(x_i) > \theta_i) p(x_i) \mathrm{d}x_i = f_T \tag{7}$$

This effectively left a single free parameter, f_T, that can be swept from 0 to 1, which allowed drawing a ROC curve and use a common framework to measure performance (see Sect. 2.6).

In order to select which features are suitable, a forward wrapper selection method was adopted [10]. In this method, features are progressively incorporated into larger and larger subsets. That is, first the best classifier using a single feature is found. Then, that best single feature is kept in the set and all the possible classifiers using two features are tested, ending with a selection of the best classifier using a subset of two features. The process continues until there are no more features available (Fig. 1).

It should be pointed out that classifiers using several features were made by combining single feature novelty

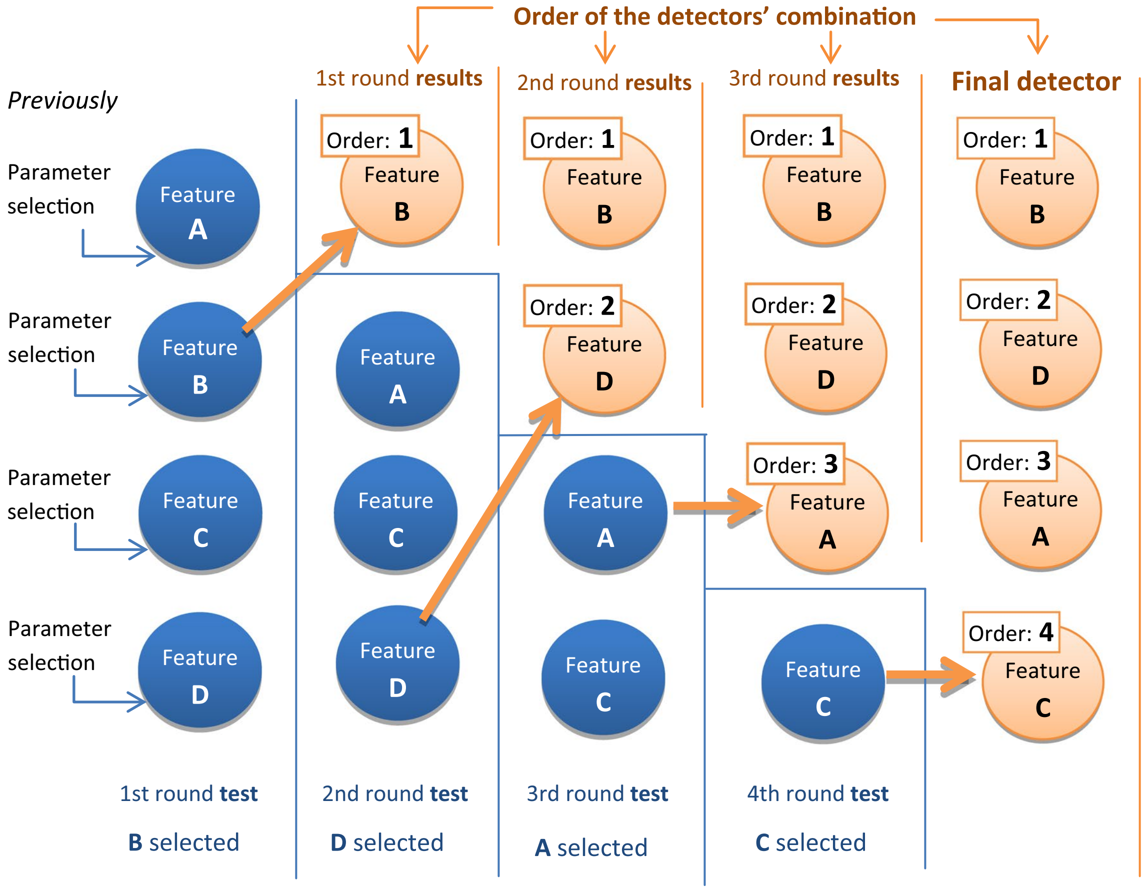

Fig. 1 Schematic view of the forward wrapper selection method

detectors. That is, feature selection is tantamount to detector selection in this paper.

2.5 Support vector machine

The proposed approach based on the combination of novelty detectors was compared with a traditional SVM [2]. Contrary to the novelty detectors presented above, SVM requires both falls and ADL for training. It finds a hyperplane, not necessarily in the original feature space, to separate two classes. Depending on the sign of the distance to it, a record is assigned to either of two classes. It has been successfully applied in many problems. After training, the decision for a new record is based on a few support vectors that lie near the hyperplane. In this paper, we have used the popular Gaussian kernel:

$$K = e^{-\gamma L} \tag{8}$$

where L is the distance between two acceleration records. As for the novelty detectors, several features can be considered. In the most complete case, the distance L takes into account all the features by defining an extended vector for each record: $\left\{ \frac{\boldsymbol{A}}{\sigma_d}, \frac{v_f}{\sigma_v}, \frac{\Delta}{\sigma_\Delta}, \frac{D}{\sigma_D} \right\}$. Each feature was divided by a standard deviation, so that none of them should dominate the others in the numerical optimization process. Thus, L is:

$$L = \sqrt{\frac{d^2(\boldsymbol{A},\boldsymbol{B})}{\sigma_{\mathrm{d}}^2} + \frac{\left(v_{fa} - v_{fb}\right)^2}{\sigma_f^2} + \frac{(\Delta_a - \Delta_b)^2}{\sigma_\Delta^2} + \frac{(D_a - D_b)^2}{\sigma_D^2}} \tag{9}$$

Combination of features was also tested by omitting some of the components in the extended vector in a forward wrapper method.

SVM has two parameters: γ in the exponential and C, which is a trade-off between minimizing training errors and controlling model complexity [2]. They are conventionally found by grid search and cross-validation [27], which is the method we have adopted.

2.6 Performance figure of merit

The kind of novelty detectors used in this work do not have a proper "training algorithm". They are pure data-driven methods based on a set of exemplars (training set), as large as possible. By varying the thresholds θ_{pp} or f_T, the performance can be evaluated on a different set (validation set), which includes ADL and falls. For instance, θ_{pp} can be varied from zero (every record is classified as an ADL) to infinity (every record is detected as a fall). The intermediate regimes are plotted in the receiver operating characteristic (ROC) curve. In this paper, the point in the curve that maximized the geometric mean of the sensitivity (SE) and the specificity (SP) was selected, and that value was used as the figure of merit for performance. This value is not affected by the different size of the ADL and fall validation sets. For SVM, the ROC curve was obtained by varying a threshold on the distance to the hyperplane found during training.

2.7 Experiments performed

The experiments have been performed in the following order:

- To determine the σ_i parameter of KDE, the data set was randomly divided in two sets (90% for training, 10% for validation). Then, performance was plotted as a function of σ_i for each feature and a value was selected (Sect. 3.1). After selecting the set of parameters, they were kept fixed in the remaining experiments.
- To see the effect of combining several detectors, the forward selection method was applied. Performance was obtained similarly to the previous case, plotting the results as a function of the number of features. This allowed us to select the best combination of features (Sect. 3.2), which was kept fixed in the remaining experiments. This was carried out for both, the combination of novelty detectors and the combination of features in SVM.

After determining the optimal parameters and features, the novelty detector and SVM were compared with and without personalization:

- The comparison of novelty detectors against SVM was done on the basis of a leave-one-subject-out cross-validation (LOOCV) (Sect. 3.3). That is, for each run, the models were built with the records of nine subjects and tested on the remaining one. A Wilcoxon test was applied to estimate statistical significance.
- The effect of personalization was studied in the last experiment. In this case, several training conditions were applied. For the nth person, a validation set (1/3 of his/her ADL and all his/her falls) was set aside. Then, the performance with different training conditions was tested, see Table 1. They correspond to a fully customized (C) classifier (training records from the nth subject), a generic (G) detector (training records from the remaining subjects), a mixed (M) detector (training records from the nth subject and the remaining volunteers) and a generic detector with the number of training records restricted (GR) to be the same as in the customized case. The notation with a hyphen will indicate the kind of training. For instance, KDE-C means the fully customized version of the combination based on KDE. The analysis was repeated for all the volunteers ($n = 1$ to 10). Even though the novelty detectors are the most suitable for personalization,

Table 1 Summary of different training sets for the *n*th subject in order to study the effect of personalization in a set of *N* volunteers

Training conditions	Training set
Customized (C)	Only ADL from the *n*th subject (2/3 of his/her ADL)
Mixed (M)	ADL as in the customized case, adding all the ADL of the remaining (N-1) people
Generic (G)	ADL from the remaining (N-1) subjects
Generic with restrictions (GR)	Only some ADL from the remaining (N-1) subjects, so that their number is equal to that of the customized version

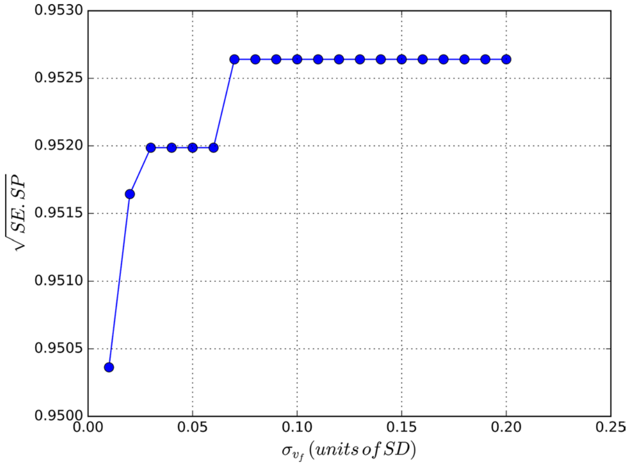

Fig. 2 Performance as a function of σ for a KDE novelty detector based on velocity

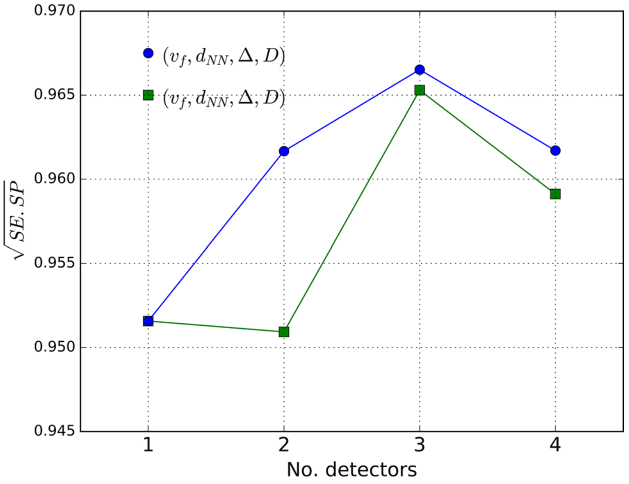

Fig. 3 Performance as a function of the number of novelty detectors being combined. *Blue circles* product probability rule; *green squares* mean vote. The order of the selected features is shown on the *left-top* part of the image (color figure online)

it was easy to include SVM in our offline analysis. The results were globally analysed considering the two factors involved, the classifier algorithm and the training data set. Thus, an ANOVA was carried out considering a design with repeated measures on both factors. Then, pairwise comparisons were performed between some selected pairs using a Wilcoxon test. The pairs were selected according to performance and other aspects, like computation time and ease of personalization, in which the members of the pair can be considered as reasonable competitors.

All the algorithms were implemented in Python using Scipy [14] and Scikit-learn [27]. In particular, models for KDE and SVM were fitted with the tools of the package Scikit-learn [27], while the rest were programmed by the authors. Statistical analyses were performed in R [28].

3 Results

3.1 Parameter selection

In Fig. 2, the performance is shown as a function of σ for the velocity-based KDE detector. The units of the abscissa are standard deviations (SD) of the corresponding feature, obtained in the training set. The performance increases with σ, but there is some saturation. A similar behaviour has been found for the rest of features. Thus, there is a wide range of parameters giving similar performance. For the rest of experiments, we selected the rule of taking σ_i as one SD of the associated feature.

3.2 Feature selection

In Fig. 3, the performance is plotted as a function of the number of features for the combination of KDE-based novelty detectors. It shows the results of the mean vote and the product probability rules. In both cases, the maximum is reached with three features. The product probability rule is better for all the combination of features (except for the case of one feature in which, actually, they are the same detector). The output of the forward wrapper selection algorithm showed that the best features were, in this order: v_f, d_{NN}, Δ and D, regardless of the combination rule. For the remaining of this paper, only the best

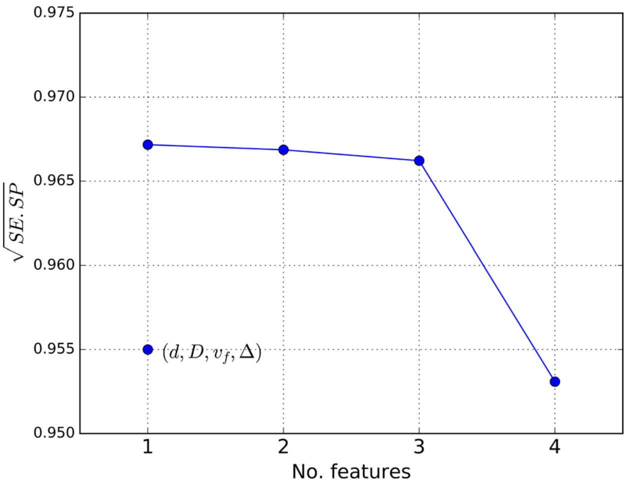

Fig. 4 Performance as a function of the number of features included in the extended vector for SVM. The order of the selected features is shown on the *left-bottom* part of the image

combination of three features is considered for novelty detectors.

Figure 4 shows the same analysis for SVM. In this case, no improvement is observed, being the results very similar for one, two and three features, and falling with four features. So, for SVM only *d* is considered in the remaining of this paper.

3.3 Combination of novelty detectors vs SVM

In this case, we compared the best combination of novelty detectors with a traditional SVM. The comparison is shown in Table 2 as values of SE, SP and the corresponding square root of the product of SE and SP. The results show that the difference with SVM was statistically significant (p value <0.01). The combination of three novelty detectors outperformed a traditional supervised detector. This improvement can also be quantified in terms of the false positive rate (FPR), which is 1-SP. In this regard, SVM roughly triggers four false alarms for each 100 ADL while KDE triggers only three false alarms. This represents an improvement of the 25%.

3.4 Effect of personalization

The effect of personalization is shown in Fig. 5 for different classification algorithms and training sets. Values of SE and SP are given in Table 3. The personalized (KDE-C, SVM-C) and generic with restrictions (KDE-GR, SVM-GR) detectors were trained with an average of 521 records; the generic versions (KDE-G, SVM-G) were trained with an average of 7034 records and a bit more (7034 + 521) for the mixed versions.

Table 2 Performance comparison of the combination of KDE novelty detectors with SVM

	SE	SP	$\sqrt{\text{SE} * \text{SP}}$
KDE	0.979	0.967	0.973
SVM	0.964	0.955	0.959

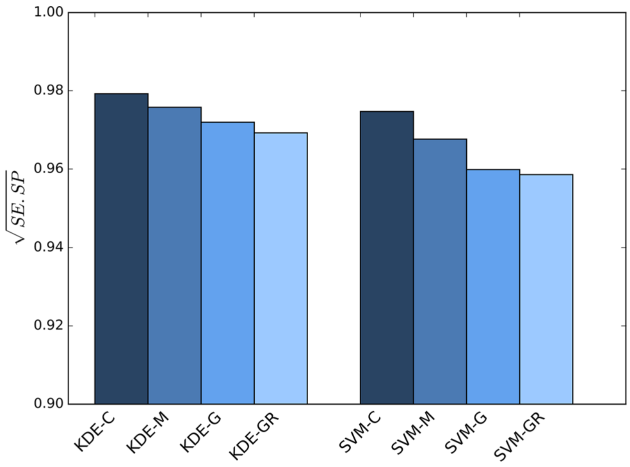

Fig. 5 Effect of personalization on different classifiers

The results of an ANOVA with two factors (classification technique and training conditions) gave us a significant effect of the classification technique (p value = 0.021) and a significant effect of the training set (p value = 0.037), while there was no significant effect of the interaction (p value = 0.619). All the p-values were obtained from Pillai's trace, although the same conclusions were drawn from other common statistics.

In view of Fig. 5, KDE-C would be the selected algorithm. However, other approaches can present different advantages, like computation time. There is no reason to prefer KDE-M, KDE-G or KDE-GR over KDE-C, since all of them have similar or higher computation time. With respect to the comparison with the SVM variants, a rough estimation of processing time in Python was carried out: SVM-C was about 18 times faster than KDE-C. Thus, we decided to check specifically the difference between KDE-C and SVM-C, giving a p value of 0.23. Therefore, the significance is not very high.

4 Discussion

The combination of novelty detectors has shown to provide a high performance. One of the drawbacks of KDE is the selection of parameters. For σ_i in KDE, it suffices to select a value large enough, which can be easily obtained from the statistics of the ADL (the standard deviation). This wide range of values is suitable for real implementation.

Table 3 Sensitivity and specificity for different degrees of personalization

Classifier	Training set	SE	SP
KDE	Custom (C)	0.986	0.972
	Mixed (M)	0.979	0.973
	Generic (G)	0.970	0.974
	Generic with restrictions (GR)	0.981	0.958
SVM	Custom (C)	0.972	0.978
	Mixed (M)	0.962	0.973
	Generic (G)	0.962	0.958
	Generic with restrictions (GR)	0.960	0.958

The combination of three novelty detectors improves performance compared to a single detector. The features selected cover different aspects. The distance to the nearest neighbour, d_{NN}, is an integral measure related to the shape of the acceleration during the fall-like event. The velocity indicates, ideally, the final speed after the movement. It is intuitive that it could be higher for falls, which are unexpected events. The change in orientation comparing the situations before and after the acceleration peak should be also a relevant discriminant feature for falls, at least for those that end with the person lying on the floor. The inclusion of new features was used traditionally to reduce the number of false positives. For instance, the authors of [15] proposed an algorithm based on the detection of impact and posture, without velocity, while in [6] it was concluded that adding velocity helped to reduce the number of false positives in unscripted normal activities. Thus, our findings are in keeping with previous studies, since our ADL data set was taken in free-living conditions. In addition, the proposed procedure is based on a more principled approach by using a clear distinction between training and validation data sets and a single measure of performance, while many previous algorithms based on simple threshold algorithms found them ad hoc in all the data set [6, 15]. Other papers considered several features at the same time, but without any previous feature selection processing to reduce their number [1, 16]. However, adding features did not help to improve SVM performance. The combination of d, Δ and v_f produced almost the same result as using only d. Overall, it seems that these features allowed a better modelling of normal events in the novelty detector, but did not provide additional information in the distance between two records in SVM. Anyway, there are many options to combine classifiers, and this topic deserves further research. Another important conclusion can be extracted from the results shown in Table 2. A combination of novelty detectors outperforms a supervised classifier under general conditions. This is important because the novelty detector is based only on real ADL data, which can be easily recorded. Although the final decision about selecting one system or another must involve real falls and it is out of the scope of this paper, it is possible that a novelty detector does not suffer too much when facing free-living fall data. Thus, this approach is promising and would avoid the burden of recording real fall data. This task would involve many volunteers carrying devices for a long period, which is far from being easily achievable. The present paper contrasts with our previous results [20], in which SVM was shown to be the best option in most situations. There are several improvements over that work. First, we have taken the three-axis acceleration, and not only its module. This enhances performance for all the classifiers (including SVM). Secondly, we have combined several novelty detectors, while in [20] only a single one was used. Thirdly, the performance evaluation has been estimated by LOOCV. Thus, the detectors face data from people out of the training set. This is more realistic. It seems that the performance of SVM decreased in this circumstance more than that of novelty detectors.

With respect to personalization, Fig. 5 suggests some trends. A generic detector with a few training records is worse than a generic detector using all the data. The performance increases if data from the user are added (mixed detector). However, it is even better in the fully personalized detector, despite being trained with a far lower number of records. Thus, using only data from the user seems to be the best option, as it has also been found for activity recognition with smartphones [31]. It is also clear that, for an equal number of training records, the use of personalized movements helps a lot to discriminate ADL from falls (KDE-C vs KDE-GR for instance). The result also agrees with [22]. The analysis of personalization performed in [22] has been improved in several aspects. Firstly, the performance evaluation is based on *SE* and *SP*, which are more practical than the area under the ROC curve. Secondly, different degrees of personalization have been tested, implying different training data set sizes. Thus, we have compared a personalized detector trained with far less records than a generic one, which is a less favourable situation but more realistic. Thirdly, we have performed an ANOVA analysis that allows getting an overall conclusion.

However, the comparison between KDE-C and SVM-C is not clear. There is mild evidence of KDE-C outperforming SVM-C, but SVM-C is faster and computation time can be an issue, especially in portable devices. The anomaly score computation in KDE-C requires the computation of many exponentials, though their number is reduced by utilizing only the ADL from a specific user. SVM-C is a sparse model that takes its decision based on a subset of the training data. This is probably the reason why it shows the smallest computation time. On the other hand, KDE is very easy to be personalized. The reason is that KDE is based on a set of exemplars, and thus, a table of recent ADL can be updated as the user carries the device, a technique used in [21] to personalize a nearest neighbour classifier. A traditional supervised detector, like SVM, could also be personalized, but only with respect to ADL, since it is unlikely that a sufficient number of falls of the same subject could be recorded. However, training algorithms would be far more complex to implement in a portable device and a server would be needed to give support to the retraining process.

5 Conclusions

To sum up, we have shown that a combination of three novelty detectors is better than a single novelty detector and, more important, the combination also outperforms a traditional classifier trained with simulated falls. Personalization further improves performance. This personalization can be easily come into practice in dedicated devices, since it would require just a table of ADL exemplars that can be updated with new user's movements. Nonetheless, all the data we have used come from a moderate number of young or mature people. Although the proposed algorithms offer the key characteristic of adaptability, they are still to be proven effective in a large sample of elderly people. The computation time of KDE-based techniques could also be improved, for instance by approximating the density function with interpolating functions, avoiding the calculation of many exponentials. Besides, the problem of fall detection has been restricted to discriminate acceleration patterns in rather short periods of a few seconds. In principle, this is reasonable, since falls themselves are very short. However, adding longer time information can be useful to avoid false alarms, checking for self-recoveries or testing a more complex approach based on states, like HMM [17].

Acknowledgements We acknowledge the "European Social Fund", the "Departamento de Innovación, Investigación y Universidad del Gobierno de Aragon" and the "TEC2013-50049-EXP Project" of the "Ministry of Economy and Competitiveness" for their support.

References

1. Albert MV, Kording K, Herrmann M, Jayaraman A (2012) Fall classification by machine learning using mobile phones. PLoS ONE 7:e36556
2. Alpaydin E (2010) Introduction to machine learning. The MIT Press, Cambridge, p 537
3. Aziz O, Musngi M, Park EJ, Mori G, Robinovitch SN (2016) A comparison of accuracy of fall detection algorithms (threshold-based vs. machine learning) using waist-mounted tri-axial accelerometer signals from a comprehensive set of falls and non-fall trials. Med Biol Eng Comput. doi:10.1007/s11517-016-1504-y
4. Bagala F, Becker C, Cappello A, Chiari L, Aminian K, Hausdorff JM, Zijlstra W, Klenk J (2012) Evaluation of accelerometer-based fall detection algorithms on real-world falls. PLoS ONE 7:e37062
5. Bourke AK, O'Brien JV, Lyons GM (2007) Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm. Gait Posture 26(2):194–199
6. Bourke AK, van de Ven P, Gamble M, O'Connor R, Murphy K, Bogan E, McQuade E, Finucane P, Olaighin G, Nelson J (2010) Assessment of waist-worn tri-axial accelerometer based fall-detection algorithms using continuous unsupervised activities. In: Conference proceedings: IEEE engineering in medicine and biology society 1, pp 2782–2785
7. Casilari E, Luque R, Morón MJ (2015) Analysis of android device-based solutions for fall detection. Sensors 15:7827–17894
8. Chandola V, Banerjje A, Kumar V (2009) Anomaly detection: a survey. ACM Comput Surv 41(3):15:1–15:58
9. Cheng W, Jhan D (2013) Triaxial accelerometer-based fall detection method using a self-constructing cascade-adaBoost-SVM Classifier. IEEE J Biomed Health Inform 17(2):411–419
10. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182
11. Habib MA, Mohktar MS, Kamaruzzaman SB, Lim KS, Pin TM, Ibrahim F (2014) Smartphone-based solutions for fall detection and prevention: challenges and open issues. Sensors 14(4):7181–7208
12. Igual R, Medrano C, Plaza I (2013) Challenges, issues and trends in fall detection systems. BioMed Engin OnLine 12(1):66
13. Igual R, Medrano C, Plaza I (2015) A comparison of public datasets for acceleration-based fall detection. Med Eng Physics 37:870–878
14. Jones E, Oliphant T (2001) Scipy: open source scientific tools for Python. http://www.scipy.org/. Accessed 18 July 2016
15. Kangas M, Konttila A, Lindgren P, Winblad I, Jämsä T (2008) Comparison of low-complexity fall detection algorithms for body attached accelerometers. Gait Posture 28(2):285–291
16. Kerdegari H, Samsudin K, Ramli A, Mokaram S (2012) Evaluation of fall detection classification approaches. In: Proceedings of the 4th intelligent and advanced systems, Kuala Lumpur 131136
17. Khan S, Karg M, Kulic D, Hoey J (2014) X-factor HMM for detecting falls in the absence of fall-specific training data. Lectures Notes in Computer Science. Ambient Assisted Living and Daily Act 8868, 1–9
18. Kuncheva LI (2004) Combining pattern classifiers: methods and algorithms. Wiley-Interscience, New York. ISBN: 0471210781
19. Luque R, Casilari E, Morón MJ, Redondo G (2014) Comparison and characterization of Android-based fall detection systems. Sensors 14:18543–18574
20. Medrano C, Igual R, Plaza I, Castro M (2014) Detecting falls as novelties in acceleration patterns acquired with smartphones. PLoS ONE 9(4):e94811
21. Medrano C, Igual R, Plaza I, Castro M, Fardoun HM (2014) Personalizable smartphone application for detecting falls. In :IEEE-EMBS international conference on biomedical and health informatics, 169–172
22. Medrano C, Plaza I, Igual R, Sanchez A, Castro M (2016) The effect of personalization on smartphone-based fall detectors. Sensors 16:117
23. Micucci D, Mobilio M, Napoletano P, Tisato F (2015) Falls as anomalies? An experimental evaluation using smartphone accelerometer data. J Amb Int Humanized Comp 1–13

24. Mubashir M, Shao L, Seed L (2013) A survey on fall detection: principles and approaches. Neurocomputing 100:144–152
25. Nathasitsophon Y, Auephanwiriyakul S, Theera-Umpon N (2013) Fall detection algorithm using linear prediction model. In: IEEE international symposium on industrial electronics, 1–6
26. Palmerini L, Bagalà F, Zanetti A, Klenk J, Becker C, Cappello A (2015) A wavelet-based approach to fall detection. Sensors 15(5):11575–11586
27. Pedregosa F et al (2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12:2825–2830
28. R Core Team (2013) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org/. Accessed 18 July 2016
29. Schwickert L, Becker C, Lindemann U, Marechal C, Bourke A, Chiari L, Helbostad JL, Zijlstra W, Aminian K, Todd C, Bandinelli S, Klenk J (2013) Fall detection with body-worn sensors: a systematic review. Z Gerontol Geriatr 46(8):706–719
30. Tax D, Duin R (2001) Combining one-class classifiers. Lectures Notes in Computer Science. Multiple classifier systems 2096: 299–308
31. Weiss GM, Lockhart JW (2012) The impact of personalization on smartphone-based activity recognition. In: AAAI workshop on activity context representation: techniques and languages, 98–104

Carlos Medrano obtained a Ph.D in Physics in 1998. His current interests are focused on the field of pattern recognition and mHealth. He is the author of several papers in fall detection.

Raúl Igual received the Ph.D in Electronics Engineering. One of his research interests is fall detection. He has published several works in journals with impact factor on this topic.

Iván García-Magariño obtained the Ph.D in Computer Science Engineering in 2009. He has published articles in journals with high impact such as “Information Sciences” and “Engineering Applications of Artificial Intelligence”.

Inmaculada Plaza obtained the Ph.D in Electronics Engineering and Communications. She is the head of the R&D&I EduQTech group that has a line of research specially focused on fall detection.

Guillermo Azuara received Ph.D from University of Zaragoza in Information and Comm. Technologies and Mobile Networks. His research interests include security, RFID and traceability related to health.