SOFT FALL DETECTION USING MACHINE LEARNING in WEARABLE DEVICES

Dominique Genoud, Vincent Cuendet
Institute of Information Systems,
University of Applied Sciences Western Switzerland
(HES-SO), Sierre, Switzerland
Email: dominique.genoud@hes-so.ch,vincent.cuendet@alumni.hes-so.ch

Julien Torrent
FST,
Fondation Suisse pour les Téléthèses
Neuchâtel, Switzerland
Email: torrent@fst.ch

***Abstract*—Wearable watches provide very useful linear acceleration information that can be use to detect falls. However falls not from a standing position are difficult to spot among other normal activities. This paper describes methods, based on pattern recognition using machine learning, to improve the detection of "soft falls". The values of the linear accelerometers are combined in a robust vector that will be presented as input to the algorithms. The performance of these different machine learning algorithms is discussed and then, based on the best scoring method, the size of the time window fed to the system is studied. The best experiments lead to results showing more than 0.9 AUC on a real dataset. In a second part, a prototype implementation on an Android platform using the best results obtained during the experiments is described.**

I. Introduction

Wearable devices have the ability to follow people wherever they go and record their movements. For disabled or elderly people this monitoring can make a crucial difference as it will allow them to stay at home and keep their independence as long as possible. For example correctly detecting people falling has a huge impact on public health which triggered a very active research in this area these last 15 years. According to the WHO (World Health Organization) between 28% and 35% of people older than 65 years old fall at least once a year and these accidents are responsible of about 40% of the total recorded injuries [1]. To address this problem many approaches are proposed, some require to install video cameras and movement detectors to follow people and detect abnormal changes in their movements, others require to wear multiple sensors in the chest and along the legs. All those solutions are pretty expensive, invasive and require a lot of tunning to give results in the field.

Fortunately, with the rise of relatively cheap and accurate wearable sensors like smart watches, real-time reactivity (i.e. a couple of seconds) can be considered for production applications.

The problem addressed here will be to detect falls among normal daily activities using continuously monitored sensor's signals. There are many ways to regroup fall detection systems, some are based on the identification of different phases [7] other on variations of the acceleration [8] and some on the different kind of sensors used [9]. The experiments described here will focus on the the data provided by wearables as they look the most promising nowadays.

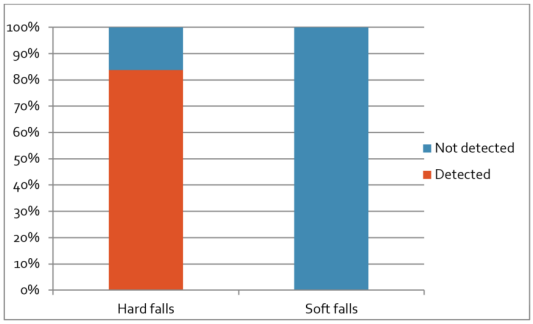

Fig. 1. Previous fall detection on the FST dataset using Threshold Based Measures (TBM)

In the current state-of-the-art of wearable devices used to detect falls, the most common features are based on the information provided by one or more accelerometers [11]–[13], [16]–[18] that will detect rapid change in movements. In addition more and more often data from gyroscopes or altimeters [19], [20] are available to precise the position of the user. To detect events on the signal of these devices the main approaches are either to fix a threshold (TBM, threshold based measure) on sensor values or to construct a machine learning model based on the signal analysis [10], [16]. Mixed approaches exists also [12] and are a good compromise in case of limited resources.

The work presented here is based on experiments recorded on real cases of falls [2] and completes previous studies [3] that used TBM (figure 1) to distinguish between hard falls (falling from the standing position) and normal activities (walking, sitting,climbing steps, freestyle). Other research team obtained similar results on hard falls like Kau & al. [14] or Albert & al. [16]. However, the previous results (see figure 1) show that it was very difficult to detect **soft falls**

Name	Format	Possible values	Type
Fall type	Alphanumeric, 2 symbols	B1, B2, B3, M1, M2, M3, NO, FR	Category
Subject	Alphabetic, 2 characters	AA, BA, BO, FA, GU, KU, LA ; NI, PI, TO, TR, UN, VA	Category
Age	Numeric, 2 digits	[22, 93]	Continuous
Sex	Alphabetic, 1 character	F, M	Category
Auxiliary mean	Alphanumeric, 2 symbols	00, CA, DR, DS	Category
Linear acceleration, X axis.	Numeric	[-34.86532974243164, 33.24461364746094]	Continuous
Linear acceleration, Y axis.	Numeric	[-44.05729675292969, 42.206565856933594]	Continuous
Linear acceleration, Z axis.	Numeric	[-29.68331527709961, 35.54317855834961]	Continuous
Acceleration, X axis	Numeric	[0,0]	Continuous
Acceleration, Y axis	Numeric	[-39.08319854736328, 39.369998931884766]	Continuous
Acceleration, X axis	Numeric	[-38.91899871826172, 39.82899856567383]	Continuous
Gyroscope, X axis	Numeric	[-38.91109848022461, 39.54209899902344]	Continuous
Gyroscope, Y axis	Numeric	[-39.08319854736328, 39.369998931884766]	Continuous
Gyroscope, Z axis	Numeric	[-13.002599716186523, 14.506699562072754]	Continuous

Fig. 2. This table describe the raw input features produced by the sensors. The experiments described here will only use the 3 axes linear accelerations and the activities labels.

(defined as falls from other starting point than the standing position, like for exemple sliding from a chair).
The experiments described here will address the detection of these soft falls by using machine learning algorithms (MLa). After a description of the data used for the experiments in section III, the performances of different MLa will be compared in section IV and improvement of the input data will be described in the section V.
Finally the last part of this paper will describe a prototype solution based on our machine learning system and implemented as real time system on a wearable devices running on Android OS (section VI).

II. Methodology

Classically machine learning systems operate in two phases: First a ML algorithm is trained with real data and a model of the observed data is constructed. This phase is often called the **training phase**. Then a **test phase** will use the model trained previously to validate its performance. To assess properly the quality of the model the test data should not have been seen during the training process.
The dataset available to construct the system is accordingly split into 2 distinct parts: the training set and the test set. There are multiple popular ways to split the training and test set, all of them have some advantages and drawbacks [4]. In this work we will use a randomly sampled 50% of the full dataset to learn our task and 50% to test the learned models. In order to increase the statistical relevance of the results, the train/test process is repeated 20 times with a new random selection for each algorithm. Moreover as for some experiments in the second part of this work we have few data for the soft fall category, a leave-one-out scoring [4] was used, proved more robust in such cases.

As the task on this paper will focus on how to separate the **soft falls** from other activities that are not falls, this implies that the hard falls (from a standing position) are removed from the dataset (see section III). Thus, the task that will be considered here will be to **classify** [4] the data in 2 categories (classes): **soft falls** against **other activities**.

To measure the separation of 2 classes we will use a well accepted scoring method that is called AUC (Area Under the Curve) [4] of a ROC curve (Receiver Opreating Characteristic) [4]. The AUC has the advantage of being independent of an a priori threshold but can be easily related to other measures like specificity and sensitivity [24].

TABLE I
Dataset used in the soft fall experiments, the mean duration of a record is around 24.5 seconds. The samples are taken every 20ms

Labels	Nr Records	Nr Samples
Soft Fall	475	145'865
Other activities	2651	12'990'095

III. Data preparation and dataset

The data used in the following experiments come from real recordings made on two different smart watches (LG-G [21], Moto360 [22]) that are producing the features described in the figure 2 every 20ms (50Hz). An Android application [2] collects and processes the data from the wearable devices. Multiple sessions involving 16 persons performing diverse activities like soft falls with or without residual activities, sitting, walking, running, climbing steps up or down were recorded. Then, following Wang & Shi [23] we kept only the 3 linear accelerometers as they determined that the linear accelerometers provide convincing results and limit the processing time required to detect the falls. The table I shows the statistics of dataset extracted from the raw data and available for the experiments. The linear accelerometers values are accelerations without the influence of the gravity. The figure 3 shows an example of a rear soft fall and of a climbing up stairs activity and how similar they can be.

Data pre-processing of the accelerometers

For stability purpose we didn't use the raw values of the 3 linear acceleration sensors but, following the equation III, the norm of the resulting vector V_n :

$$V_n(i) = \sqrt{(a_x(i))^2 + (a_y(i))^2 + (a_z(i))^2}$$

with i the sample number. and $a_{x,y,z}$ are the linear acceleration along the three axes x, y, z

After a first inspection of the soft falls pattern we decided to feed the inputs of the MLa classifiers with records of 200 vectors V_n which represent a time window of

$$20 * 200 = 4000ms$$

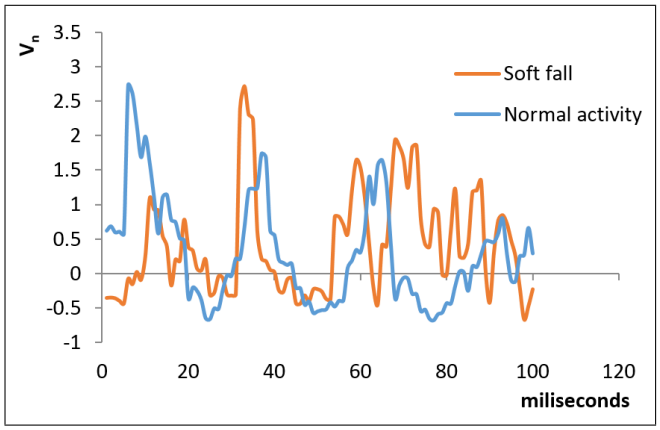

Fig. 3. Example of soft fall signal vs other activity

TABLE II
PERFORMANCE OF POPULAR MACHINE LEARNING ALGORITHMS, AUC (AREA UNDER CURVE) AND CONFIDENCE INTERVALE (CI) AT $\alpha = 95\%$ RANDOM REFERS TO RANDOM INPUT SCORING AND LOO TO LEAVE-ONE-OUT SCORING

ML Algorithm	AUC/random	AUC/loo
Decision Tree	0.63 $\pm$ 0.02	0.68 $\pm$ 0.02
DT Forest	**0.90 $\pm$ 0.01**	**0.90 $\pm$ 0.01**
KNN	0.82 $\pm$ 0.02	0.81 $\pm$ 0.02
MLP	0.72 $\pm$ 0.02	0.71 $\pm$ 0.02

IV. BEST ALGORITHM SELECTION

Following the methodology described in section II we split the dataset in 2 parts by randomly taking half the records to construct the training set and the other half for the test set. We also ran a leave-on-out experiment to cross check the obtained results. As many MLa have some convergence problem when the ratio of samples between classes is not equilibrated we use a simple bootstrap sampling method [15] to balance the training data. Moreover as Neural Network like the Multi-Layer Perceptrons (MLP) have problems if the input features are not normalized, we do compute some z-score normalizations [4] of the input features on the training records and we reuse the same normalization on the test set.
The machine learning algorithms compared here are implemented on Knime [5], a widely used data mining platform (see figure 4). The detailed mathematical equations and convergences of the algorithms used can be found in [4]:

1) *Decision Tree*: The decision trees (also called C4.5) use a gini index computation on each feature of the input data to find the best separation of the classes.
2) *Decision Tree Ensemble (DT Ensemble)*: this algorithm will learn an ensemble of decision trees (also called random forest), for the experiments we used a maximum of 300 trees and an information gain ratio as distance measure.

TABLE III
PERFORMANCE OF THE FOREST TREES ACCORDING TO THE ANALYSIS WINDOW DURATION, AUC (AREA UNDER CURVE) AND CONFIDENCE INTERVALE AT $\alpha = 95\%$. THE SOFT FALLS% IS THE QUANTITY OF SOFT FALLS IN THE DATASET

Window [ms]	Nr records	soft fall%	DTE-AUC
100	1012	46.64	0.78 $\pm$ 0.01
200	1012	46.64	0.77 $\pm$ 0.01
2000	926	43.30	0.91 $\pm$ 0.01
4000	637	23.70	0.90 $\pm$ 0.01
8000	527	7.78	0.87 $\pm$ 0.02

3) *K Nearest neighbors (KNN)*: the nearest neighbor is configured with a maximum of 50 neighbors.
4) *Neural networks, multilayer perceptrons (MLP)*: the values fed to the input neurons are z-score normalized and the neural network is configured with one hidden layer of 10 neurons.

Results

The results obtained with this setup are shown on the table II. We can observe that the Decision Tree Ensemble scores clearly the best and outperform the results obtained by the other algorithms like the MLP cited very often in the litterature [12], [14], [16], [19], [20]. By observing the classification errors made by the MLa it was possible to determine that some activities like climbing down stairs were often detected as soft falls. Among many way to solve this confusion, like adding frequency components for example, a possibility was also to better capture the details of the fall pattern by adapting the analysis time window. This approach will be described in the section V. Interestingly the leave-one-out scoring does not give significantly different results which is a good sign of the robustess of the dataset. The confidence intervale on the AUC is computed according to Hanley & McNeil [6].

V. IMPROVE THE DETECTION WINDOW

The duration of the record that will contain the shape of a soft fall was determined empirically in our first experiments. So, in order to optimize the time window that will give us the best classification results, we have setup different record time windows. Then, we have built a system using the Decision Tree Ensemble, our best performing algorithm (see section IV) and fed it with time windows varying from 100 ms to 8 seconds. As the number of records available varies greatly upon the duration of the window, we used a leave-on-out scoring (see section II) to have the same kind of scoring for all the experiments.

Results

The results can be seen on table III. There is a small difference in performance by using 4000 ms over 2000 ms

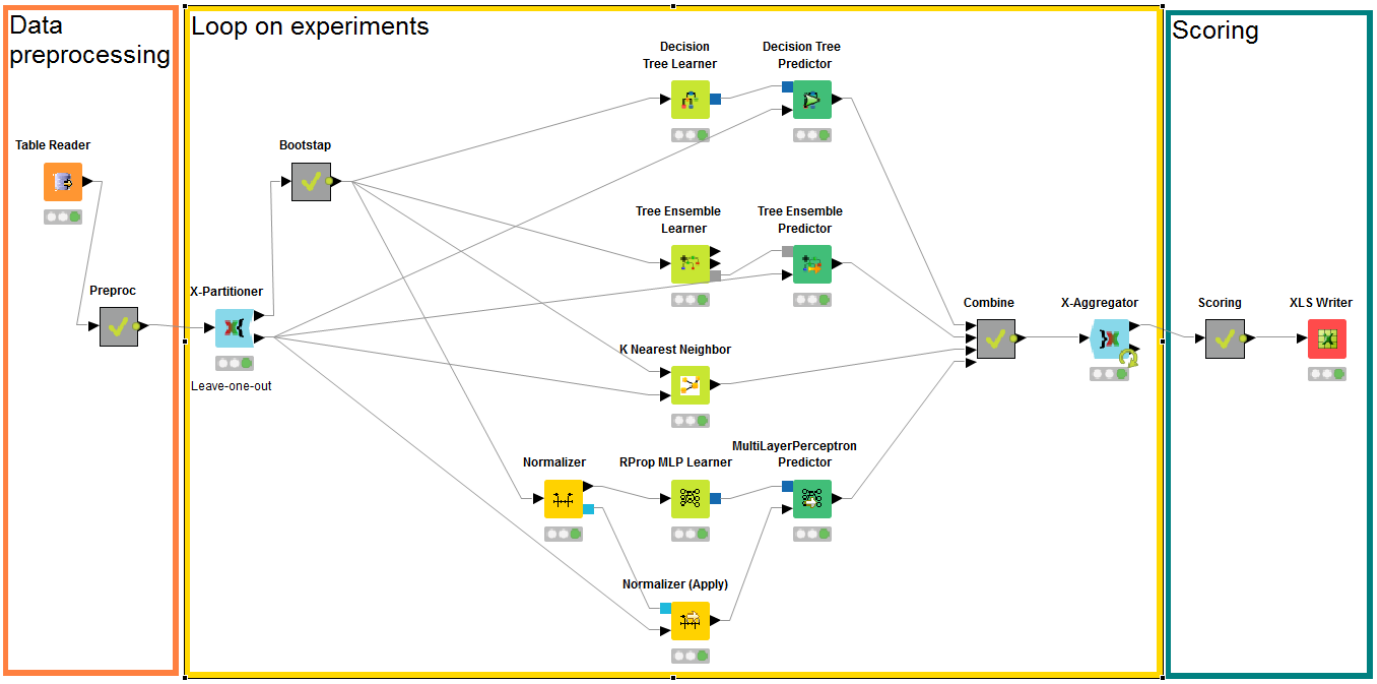

Fig. 4. Example of Knime workflow that shows the methodology used: Data preparation, algorithm selection and scoring

but is not significant if we look at the confidence interval. However, dividing by 2 the analysis window has a great impact on the resources needed to process the signal, a major concern when deploying a system on a mobile device. Thus, this result will be exploited during the implementation phase (see section VI) and tuned in a way that the results stay robust.

VI. Implementation

The implementation of the experiments in a real-life system is the next, logical, step to capitalize on demonstrated elements. This step pursue several objectives: first, target an existing and affordable hardware, second being able to convey usability, especially for elderly people, and third offer confidence through robustness and precision. We chose an Android platform because of its large user base on mobile devices that is spreading from entry level to specialized terminals. The export in a shareable form of the produced model is done using the PMML (Predictive Model Markup Format) format, generated from Knime platform. The methodology to get a consumable XML file differs from the precedent experiments as the data separation between train and test datasets is done using a stratified sampling with a random seed [4]. This choice is driven by the necessity to reduce the prediction model size and its footprint on production system. Then, the consumption of the model is done by the open-source library jpmml-evaluator. The selection of this code base was motivated by its full support of the latest PMML standards without incurring additional costs. The precision of the system is guaranteed by the use of the MLa approach and the selected options described in the section V.

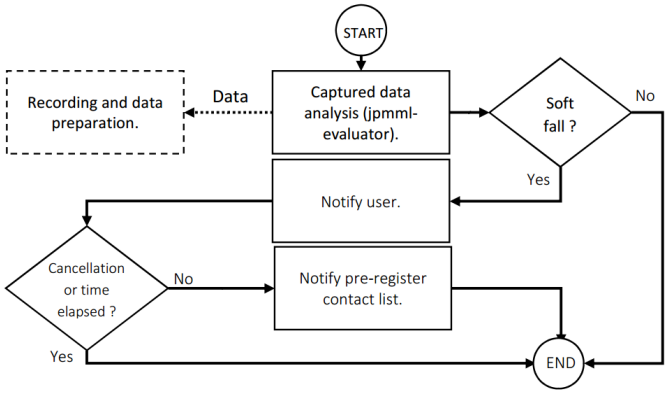

Fig. 5. Implementation scheme for the soft falls detections.

The system work-flow, shown in figure 5, is composed by daemon collecting data from device sensors which stores them in a memory buffer and from a process which evaluates the stored content every second [2]. It interacts with the end-user and finally triggers an alarm if necessary. A detection flow analyzes the data contained in the buffer every second, then the decision tree model evaluates the 100 input samples, and finally if no fall is detected the process ends. In case of soft fall detection, a message is displayed to the user who can decide to cancel the flow. In that case the process ends, otherwise, after a parametrized time-frame, for example 15 seconds, the alarm is relayed by e-mail or SMS to list of predefined contacts [2].

VII. Conclusion

The experiments releted in this paper show that it is possible to obtain robust results to detect **soft falls** at more than 0.9 AUC by using machine learning algorithms on 2 seconds recording windows. The best algorithm is the Decision Tree Ensemble that was constantly better than others. We also confirmed that using the norm V_n of the linear accelerations on the 3 axes x, y, z gives better results than multiple others tests (like the use of frequency components or the angle of the accelerations). Future work to improve the detection of soft falls will be to add new input features like gyroscopic and compass informations for example. As the system implemented is good enough to be used in real production, the main improvement will certainly be to analyze the errors happening in special cases and retrain the machine learning algorithms able to capture this kind of problems in their modeling.

Finally, the architecture presented in this work combined with the already efficient threshold detection for hard falls respond to the objectives of a real application. Indeed, it allows a deployment on a widely used platform, uses standards, keep the interaction with the end-user as simple as possible and yet offers the necessary control and detection capacity to be efficient, thus minimizing wrong alarms and enforcing confidence in the system.

Acknowledgment

The authors would like to thank the Fondation Suisse pour les Téléthèses for its support and its very important help during the cleanup of the dataset of the project.

References

[1] World Healt Organization. *WHO Global Report on Falls Prevention in Older Age*. World Health Organization, 2007.

[2] Torrent, J., Triki, M., Khner, D., Nicolet, M., Sieber, G., Vallat, M., Chassot, C., Edelman, R., Aebi, R., & Nedjmeddin, B. *Développement d'un dispositif innovant de détection de chute à l'aide d'une smartwatch Android* FST - Fondation Suisse pour les Téléthèses, To be published Neuchtel 2016

[3] Kostopoulos, P., Nunes, T., Slavi, K., Deriaz, M., & Torrent, J. *Increased Fall Detection Accuracy in an Accelerometer-Based Algorithm Considering Residual Movement*. fstlab.ch, 2014.

[4] Berthold, M., Borgelt, C., Hppner, F. & Klawonn, F. *Guide to Intelligent Data Analysis, How to Intelligently Make Sense of Real Data*. Springer, ISBN 978-1-84882-259-7, 2010.

[5] Berthold, M., Cebron, N., Dill, F., Gabriel, T., Kötter, T., Meinl, T., Ohl, P., Sieb, C., Thiel, K., & Wiswedel,B. *KNIME: The Konstanz Information Miner* SPRINGER, Studies in Classification, Data Analysis, and Knowledge Organization,ISBN 978-3-540-78239-1, GfKL 2007.

[6] Hanley JA & McNeil BJ. *The meaning and use of the area under a receiver operating characteristic (ROC) curve.* RADIOLOGY, Vol 143, No1, pp 29-36, April 1982

[7] Noury, N., Rumeau, P., Bourke, A. K., OLaighin, G.,& Lundy, J. E. *A proposal for the classification and evaluation of fall detectors.* IRBM, pp. 340-349, 2008.

[8] Mubashr, M., Ling, S., & Luke, S. *A survey on fall detection: Principles and approaches.* Neurocomputing, pp. 144-152, 2013.

[9] Perry, J. T., Kellog, S., Vaidya, S. M., Jong-Hoon, Y., Ali, H., & Sharif, H. *Survey and evaluation of real-time fall detection approaches.* Proceedings of the 6th International Symposium High-Capacity Optical Networks and Enabling Technologies (pp. 158-164) 2009. Alexandria: Institute of Electrical and Electronics Engineers.

[10] Igual, R., Medrano, C., & Plaza, I. *Challenges, issues and trends in fall detection systems* BioMedical OnLine, 12:66, 2013.

[11] Li, Yanjun; Chen, Gan; Shen, Yueyun; Zhu, Yihua; Cheng, Zhen *Accelerometer-based fall detection sensor system for the elderly* IEEE International Conference on Cloud Computing and Intelligence Systems. pp 1216-1220. 2012.

[12] Aguiar, Bruno; Tiago, Rocha; Joana, Silva; Inłs, Sousa *Accelerometer-based fall detection for smartphones* IEEE International Symposium on Medical Measurements and Applications (MeMeA), Lisboa. p1-6. 2014.

[13] Bagal, Fabio; Becker, Clemens; Capello, Angelo; Chiari, Lorenzo; Aminian, Kamiar; Hausdorf, Jeffrey M.; Ziljlstra, Wiebren; Klenk, Jochen *Evaluation of Accelerometer-Based Fall Detection Algorithms on Real-World Falls* PLoS ONE 1-9. 2012.

[14] Kau, L.-J., & Chen, C.-S. *A Smart Phone-Based Pocket Fall Accident Detection, Positioning, and Rescue System.* IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS, VOL 19, NO 1, 44-56.2015.

[15] Haibo He & Edwardo A. Garcia *Learning from Imbalanced Data* IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 21, NO. 9, SEPTEMBER 2009

[16] Albert, M. V., Kording, K., Herrman, M., & Jayaraman, A. *Fall Classification by Machine Learning Using Mobile Phones.* U. H. Christian Lovis, Ed. PLos ONE, pp. 1-6. 2012.

[17] Karantonis, Dean M.; Narayanan, Michael R.; Mathie, Merryn; Lovell, Nigel H.; Celler, Branko G. *Implementation of a Real-Time Human Movement Classifier Using a Triaxial Accelerometer for Ambulatory MonitoringIEEE* IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE 2006 pp 156-167

[18] Abbate, S., Avvenuti, M., Bonatesta, F., Cola, G., Corsini, P., & Vecchio, A. *A smartphone-based fall detection system.* Pervasive and Mobile Computing, 883-899. 2012.

[19] He, J., Hu, C., & Li, Y. *An Autonomous Fall Detection and Alerting System based on Mobile and Ubiquitous Computing.* International Conference on Ubiquitous Intelligence & Computing and 2013 IEEE 10th International Conference on Autonomic & Trusted Computing (p. 539). IEEE. 2013

[20] Ojetola, Olunkule; Gaura, Elena I.; Brusey, James *Fall Detection with Wearable SensorsSAFE (SmArt Fall dEtection)* IEEE 2011 Seventh International Conference on Intelligent Environments

[21] Smartphone LG-G *https://support.google.com/androidwear/answer/6056447?hl=en*

[22] Smartphone Motorola 360 *https://support.google.com/androidwear/answer/6088867?hl=en*

[23] Wang, Ye; Bai, Xiang-yu *Research of Fall Detection and Alarm Applications for the Elderly* IEEE 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC) Shenyang, China

[24] Bossuyt PM, Reitsma JB, Bruns DE, Gatsonis CA, Glasziou PP, Irwig LM & al. *Towards complete and accurate reporting of studies of diagnostic accuracy: the STARD initiative* Clin Chem 2003;49:1-6.

[25] Zhu, Yunyue *High Performance Data Mining in Time Series: Techniques and Case Studies* PhD thesis, Department of Computer Science, New York University 2004

[26] Morent, D., Stathatos, K., Lin, W., Berthold, M. *Comprehensive PMML Preprocessing in KNIME* ACM Proceedings of the 2011 workshop on Predictive markup language modeling, pp28-31. 2011