Lessons Learned From the Development of Mobile Applications for Fall Detection

Authors: Italo L. de Araújo[1], Rossana M. C. Andrade[1], Evilasio C. Junior[1], Pedro Almir Oliveira[1], Breno Oliveira[1] and Paulo Aguilar[1]

Presenter: Evilasio Costa Junior

Email: evilasiojunior@great.ufc.br

[1]Group of Computer Networks, Software Engineering and Systems (GREat)
Federal University of Ceará (UFC) - Fortaleza - CE - Brasil

UNIVERSIDADE FEDERAL DO CEARÁ

MDCC Master and Doctorate Program in Computer Science

GREat Group of computer network Software engeenering and System

IARIA

Presenter Profile

Evilasio Costa Junior has graduation at Computer Science by State University of Ceará (2010) and master's at Academic Master in Computer Science by State University of Ceará (2015). Acted as substitute professor of the undergraduate course of Science Compute at the State University of Ceará, between 2015 and 2017. Currently is Doctor Student of Federal University of Ceará since 2017. Has experience in the area of Computer Science. Focused, mainly, in the subjects: Self-Adaptive Systems, Self-Adaptive Service-based Systems, Adaptation Cycle, Frameworks, Internet of Things (IoT), and IoT Health

Research Area

We are part of the Computer Networks, Software and Systems Engineering Group (GREat) at Federal University of Ceará. The authors of this work work on research projects in the areas of Internet of Things for health (IoT Health), focusing on:

- Automatic fall detection
- Detection of possible falling causes
- Monitoring and identification of health problems based on movement data (such as gait, speed, posture and location)
- Quality of life monitoring and improvement
- Ambient Assited Living (AAL)

Source: https://www.flaticon.com/free-icon/smartwatch_2622363

Contextualization

- About 28-35% of the people aged 65 years and over fall every year, and this number increases when the person's age is over 70 years old, achieving 32-42%[1]
- Fall detection systems provide a way to quickly identifying older adults falls[2]
- There are many ways to detect falls
  - Smartphone and Wearables[3]
  - Computer Vision[4]

[1] W. H. Organization, “Global report on falls prevention in older age" http://www.who.int/ageing/publications/Fallsprevention7March.pdf, October, 2017
[2] I. L. de Ara´ujo, L. Dourado, L. Fernandes, R. M. d. C. Andrade, and P. A. C. Aguilar, “An algorithm for fall detection using data from smartwatch,” in 2018 13th Annual Conference on System of Systems Engineering (SoSE), June 2018, pp. 124–131
[3] R. L. Almeida, A. A. Macedo, Í. L. de Araújo, P. A. Aguilar, and R. M. Andrade, “Watchalert:An evolution of the fAlert app for detecting falls on smartwatches,” in Extended Proceedings of the XXII Brazilian Symposium on Multimedia and Web Systems. SBC, 2016, pp. 124–127
[4] R. Cucchiara, A. Prati, and R. Vezzani, “A multi-camera vision system for fall detection and alarm generation,” Expert Systems, vol. 24, no. 5,2007, pp. 334–345

Motivation

- Although several studies propose solutions for automatic fall detection in the literature, a few them highlighting the difficulties of developing these systems
- Since 2016, our research group has been developing and evolving fall detection solutions using mobile devices and wearables
- Throughout our experience, we have identified several points of improvement and lessons that we believe are valuable for the development of this type of applications

Goal

1. Report a development and evolution experience of two fall detection solutions using smartphones and wearables

2. Compile and discuss the top six lessons learned while developing fall detection solutions to facilitate future research

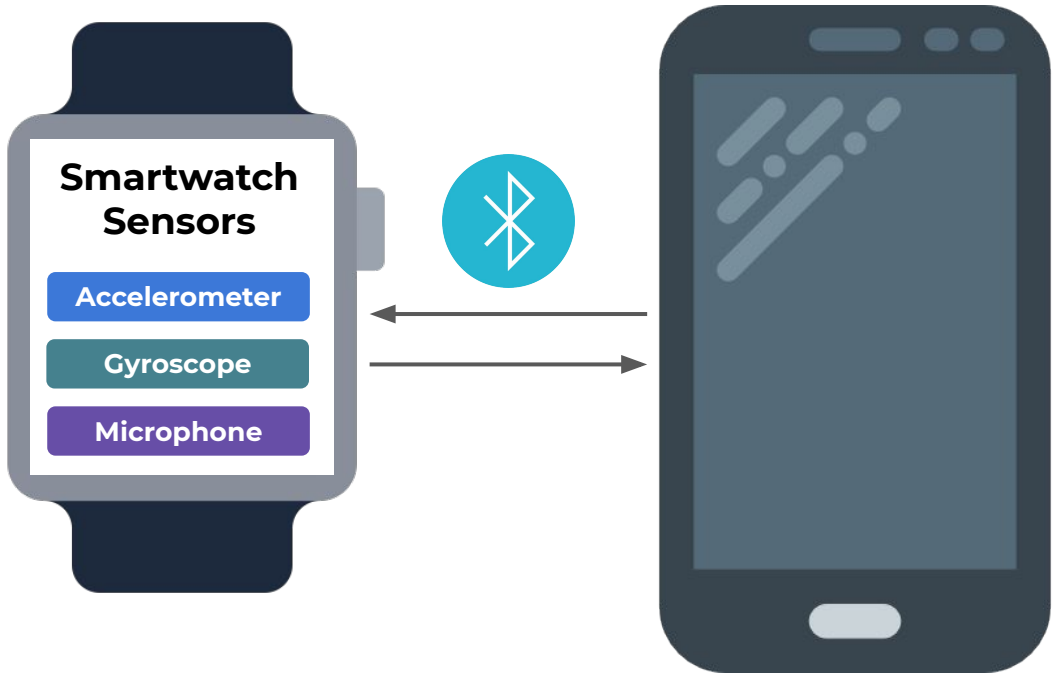
Related Works

Paper	Goal	Algorithms	Devices/Sensors	Lessons Learned
Peñaforte-Asturiano et al. 2018	Multimodal database to detect falls	-	Infrared, ECG, cameras, wearables	Database consolidation
Liu and Cheng 2012	System to detect a fall	SVM	Accelerometer	No
Andò et al. 2016	System to detect a fall	Threshold	Accelerometer, Gyroscope, Magnetometer	No
Dziak et al. 2017	Healthcare monitoring system based on wearables	Thresholds and ML	Wearables	No
Khojasteh et al. 2018	System to detect a fall	Thresholds and Optimization	Accelerometer	No

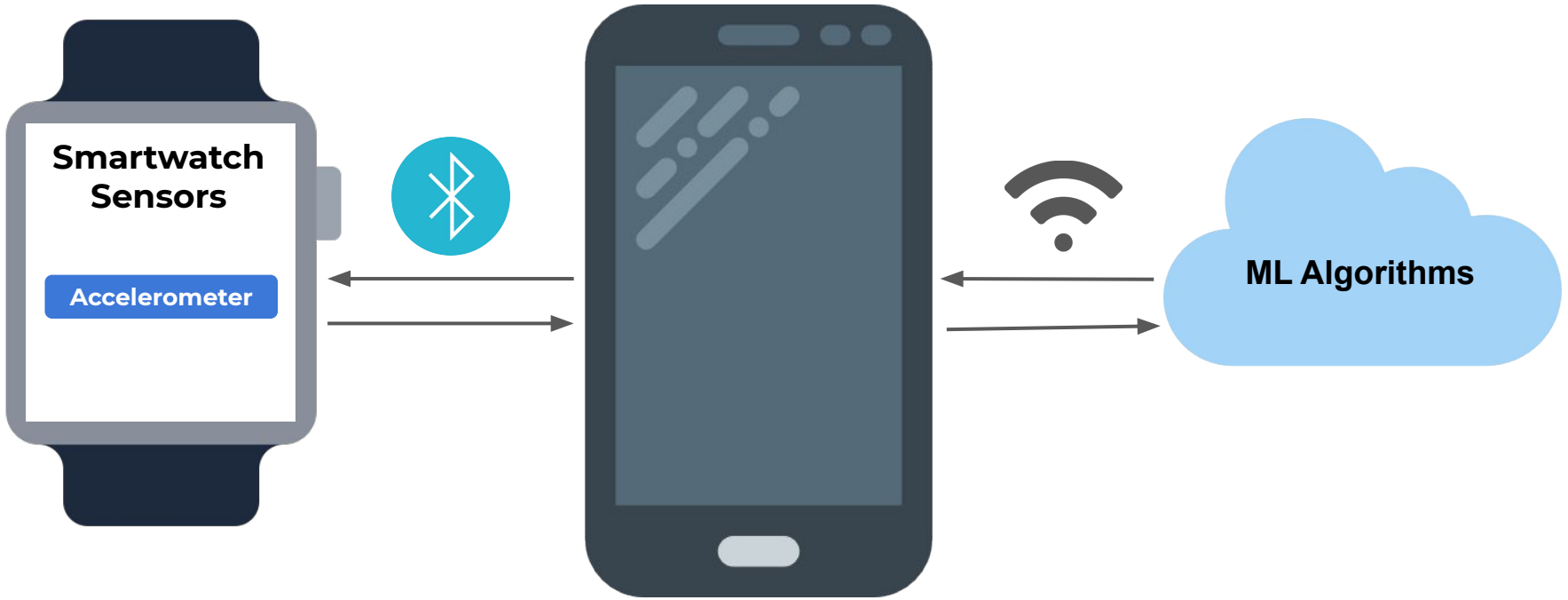
fAlert

- Application to detect a fall from smartphone data
- Allocated in user's chest
- In this version, the app uses accelerometer, magnetometer and microphone

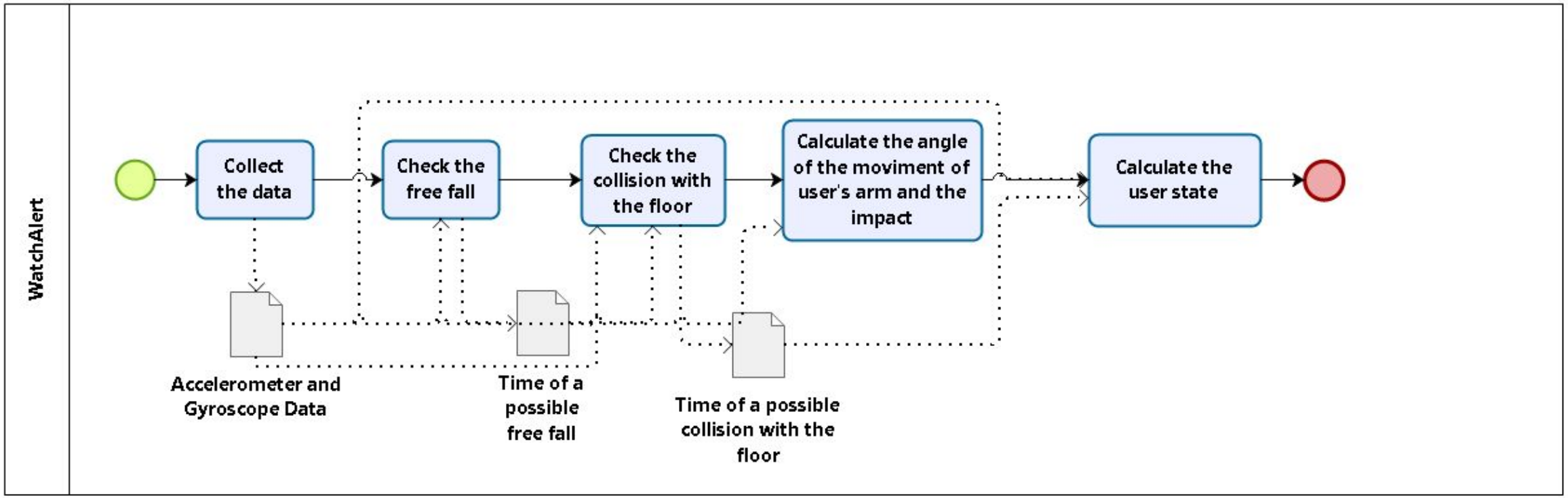
First Version



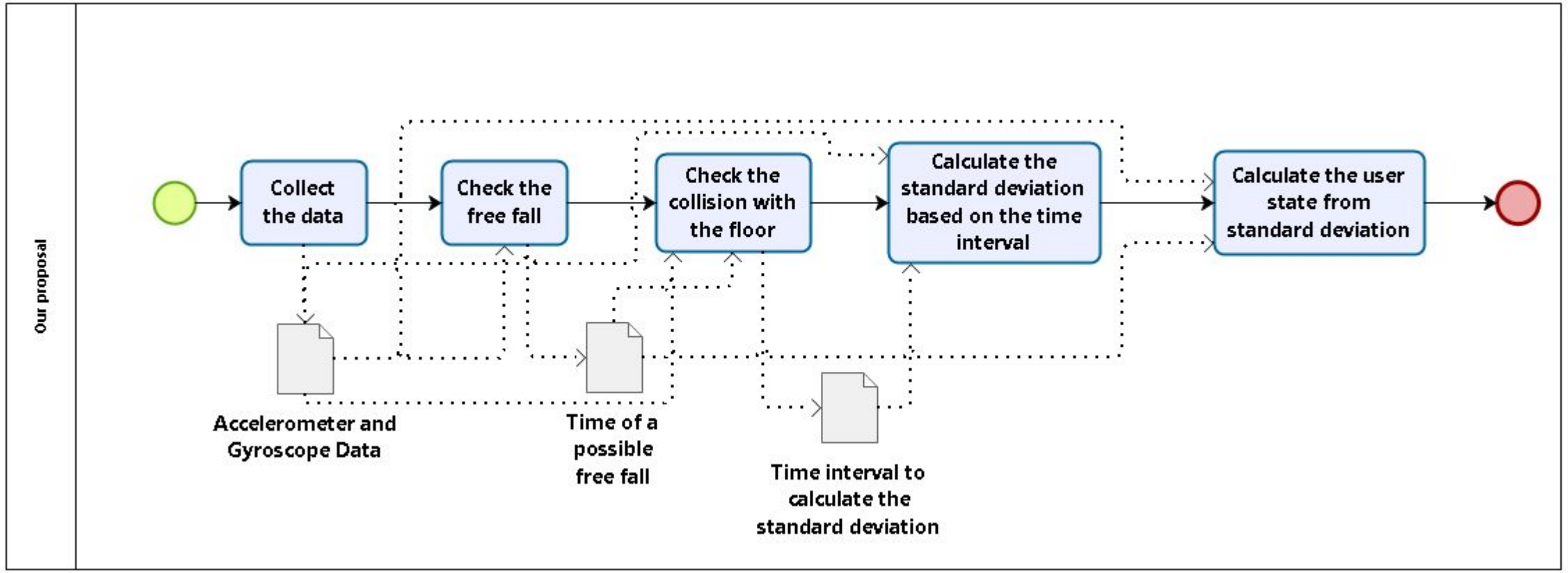
Second Version



First Threshold Approach



Second Threshold Approach

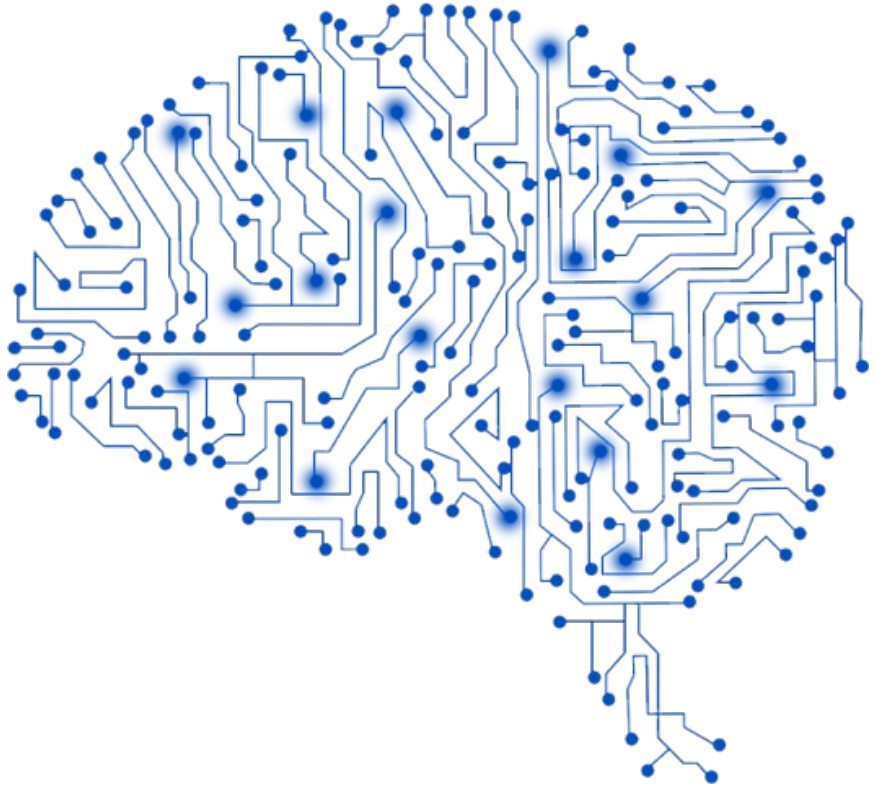


Threshold Evaluation

Algorithm	Sensibility	Specificity
WatchAlert First Version	93,8%	95,5%
WatchAlert Second Version (without gyroscope)	92,9%	95,5%

Machine Learning Approach

- In the third version of WatchAlert, we use supervised Machine Learning to detect falls
- This approach allows for greater accuracy than the threshold approach but requires more processing power to function, so it needs access to the cloud
- For evaluations, we use the Weka Tool[5]



Source: https://www.pngwing.com/

[5] M. Hall et al., "The WEKA data mining software:an update," ACM SIGKDD explorations newsletter, vol. 11, no. 1, 2009, pp. 10–18

Feature Evaluation

- Standard deviation and the average are strongly correlated with the RMS, so it was decided to remove these features

	STD	MEAN	MAX	MIN	RMS	Kurtosis
STD	1	0.4575	0.87859	-0.74941	0.76955	-0.31087
MEAN	0.4575	1	0.4587	-0.00775	0.88088	-0.33049
MAX	0.87859	0.4587	1	-0.63796	0.70221	0.0596
MIN	-0.74941	-0.00775	-0.63796	1	-0.3261	0.07504
RMS	0.76955	0.88088	0.70221	-0.3261	1	-0.40499
Kurtosis	-0.31087	-0.33049	0.0596	0.07504	-0.40499	1

Algorithm Evaluation - Original Dataset

Classifier	Classifier Detail	Accuracy (%)	F. Positive (%)	Time Training (s)	Time Testing (s)
Attribute Selected Classifier	Random Forest (Info Gain)	93.85	25.09	0.11	0.00093
	SVM - Linear Kernel	93.46	20.24	0.20	0.00076
	SVM - RBF Kernel	91.27	41.42	0.06	0.00155
	SVM- Polynomial Kernel	87.21	23.06	53.49	0.00064
Cost Sensitive Classifier	Attribute Selected: Random Forest	93.43	19.09	0.10	0.00078
	Attribute Selected: SVM - Linear Kernel	88.83	15.76	0.41	0.00063
	SVM - Linear Kernel	88.64	15.65	0.43	0.00203
	SVM - Polynomial Kernel	83.10	20.75	53.59	0.00141
	SVM - RBF Kerne	87.77	30.90	0.02	0.00173

Algorithm Evaluation - Original Dataset + SMOTE

Classifier	Classifier Detail	Accuracy (%)	F. Positive (%)	Time Training (s)	Time Testing (s)
Attribute Selected Classifier	Random Forest (Info Gain)	95.13	5.20	0.17	0.00175
	SVM - Linear Kernel	88.47	12.35	0.31	0.00110
	SVM - RBF Kernel	92.09	8.19	0.06	0.00277
	SVM- Polynomial Kernel	88.49	11.81	36.29	0.00189
Cost Sensitive Classifier	Attribute Selected: Random Forest	93.11	5.97	0.15	0.00064
	Attribute Selected: SVM - Linear Kernel	76.48	19.45	0.33	0.00139
	SVM - Linear Kernel	76.40	19.56	0.29	0.00295
	SVM - Polynomial Kernel	87.52	11.45	5.51	0.00109
	SVM - RBF Kerne	89.81	8.43	0.03	0.00281

Algorithm Evaluation - Original Dataset + SMOTE

Classifier	Classifier Detail	Accuracy (%)	F. Positive (%)	Time Training (s)	Time Testing (s)
Attribute Selected Classifier	Random Forest (Info Gain)	95.13	5.20	0.17	0.00175
	SVM - Linear Kernel	88.47	12.35	0.31	0.00110
	SVM - RBF Kernel	92.09	8.19	0.06	0.00277
	SVM- Polynomial Kernel	88.49	11.81	36.29	0.00189
Cost Sensitive Classifier	Attribute Selected: Random Forest	93.11	5.97	0.15	0.00064
	Attribute Selected: SVM - Linear Kernel	76.48	19.45	0.33	0.00139
	SVM - Linear Kernel	76.40	19.56	0.29	0.00295
	SVM - Polynomial Kernel	87.52	11.45	5.51	0.00109
	SVM - RBF Kerne	89.81	8.43	0.03	0.00281

Best Result

LL01: Use Flow-based Programming

- **Problem:** Considering the sending of batch data, one problem occurs when the free-fall occurs in a batch, and the impact with the floor and the user state are in another. Then, the fall detection behavior can be compromised and the emergence service or caregivers cannot be contacted. How to deal with this situation?

- **Lesson:** We suggest the developers of the fall detection solutions use flow-based programming because it allows the sequenced data to avoid the improvement of the false-negative rate. This strategy is especially important in these situations because the fall detection solution deal with real-time data and needs of the precise and correct result to avoid the severe sequelae with the user.

LL02: Use Service-Oriented Architecture (SOA)

- **Problem:** As each device perform several actions, for instance, the smartwatch collects the data and interact with the user to receive the answer of the real status or the smartphone, which gets the data, process it and sends the message to caregivers or emergence service, it is important to deal with this. How can the developers of the fall detection solutions maintain several functionalities without one impact in another?

- **Lesson:** We should develop an application using Service-Oriented Architecture, dividing each functionality in service. For instance, we can run the data collection and check the user's health in the smartwatch. This avoids the increase of the false-negative rates, turning the application more accurate.

LL03: Divide the code in different devices

- **Problem:** Wearables devices have constraint hardware, which allows the use of a few services and algorithms. Then, these devices cannot collect and process the data and call the user's caregivers. How we deal with this situation without to impact on the result?

- **Lesson:** The division of the code between different devices according to each hardware configuration. This strategy is possible due to the IoT solutions, which are composed of several devices and sensors with different configurations. Then, with this lesson, the developers of fall detection applications can better use the devices and do not suffer any impact on the accuracy of the algorithm neither in any other important service of the application.

LL04: Use offline methods of fall detection integrated with ML algorithms

- **Problem:** Health applications are critical, and the information should always be available. However, if the application uses only algorithms in the cloud, the solution can have a problem when the connectivity with the Internet and the service is inaccessible. Then, how can we assure the high availability of the service and a self-adaptive detection model?

- **Lesson:** We suggest putting at least another algorithm in the smartphone to run in the situations of Internet problems. Therefore, it is possible to ensure high service availability even that the application lost the internet connection.

LL05: Analysis and Selection of the best features for the ML algorithm

- **Problem:** When using machine learning for fall detection, we can use several features and not all features contribute to improve the result. Moreover, the greater the number of features may consequently make the algorithm processing slower. Then, how can we select better features for the machine learning algorithm?

- **Lesson:** We recommend using correlation analysis and attribute analysis algorithms, such as Information Gain and SVM Attribute Evaluation. In the fall detection scenario, we suggest using the maximum, minimum values, RMS, and kurtosis, when use only accelerometer data.

LL06: Choose the most suitable ML algorithm

- **Problem:** Many times, we see works that use ML algorithms for fall detection but do not explain the chosen. However, some factors impact the algorithm results, e.g., the feature type, the amount of data, and how this data is organized. Then, what algorithm should we choose?

- **Lesson:** Choose the ML algorithm for your fall detection applications based on the data available. In the fall detection scenario, we suggest at least evaluate variations of SVM algorithm and tree-based algorithms when work with only the accelerometer data.

Conclusion

- This work presented two solutions using a smartphone and smartwatch for automatic fall detection and its evolution over time
- We present and discuss the main lessons learned were throughout the development and evolution of the applications presented
- Finally, we believe that by following these guidelines, it is possible to create a robust, seamless, and easy-to-maintain fall detection service using sensors present on smartphones or smartwatches

Future Works

- Evaluate battery consumption and solutions latency
- Compare the threshold approach and machine learning algorithms

References

[6] C. J. Peñafort-Asturiano, N. Santiago, J. P. N ú ñez-Martínez, H. Ponce,and L. Mart ínez-Villaseñor, “Challenges in data acquisition systems:lessons learned from fall detection to nanosensors,” in 2018 Nano technology for Instrumentation and Measurement (NANOfIM).IEEE,2018, pp. 1–8

[7] S. H. Liu and W. C. Cheng, “Fall detection with the support vector machine during scripted and continuous unscripted activities,” Sensors,vol. 12, no. 9, 2012, pp. 12 301–12 316

[8] B. Andò, S. Baglio, C. O. Lombardo, and V. Marletta, “A multisensor data-fusion approach for adl and fall classification,” IEEE Transactions on Instrumentation and Measurement, vol. 65, no. 9, 2016, pp. 1960–1967

[9] D. Dziak, B. Jachimczyk, and W. Kulesza, “Iot-based information system for healthcare application: design methodology approach,” Applied Sciences, vol. 7, no. 6, 2017, p. 596

[10] S. Khojasteh, J. Villar, C. Chira, V. Gonz ález, and E. De La Cal,“Improving fall detection using an on-wrist wearable accelerometer,” Sensors, vol. 18, no. 5, 2018, p. 1350

UNIVERSIDADE FEDERAL DO CEARÁ

MDCC
Mestrado e Doutorado em Ciência da Computação

GREat
Grupo de Redes de Computadores
Engenharia de Software
e Sistemas

CAPES

IARIA

Thanks!

Evilasio C. Junior
evilasiojunior@great.ufc.br