Daniel Jorge Loureiro Fidalgo do Vale Rodrigues

Licenciado em Ciências da Engenharia Eletrotécnica e de Computadores

Risk Assessment for Alzheimer Patients, using GPS and Accelerometers with a Machine Learning Approach

Dissertação para obtenção do Grau de Mestre em

Engenharia Electrotécnica e de Computadores

Orientador: Ricardo Jardim Gonçalves, Full Professor, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa

Co-orientador: Fernando Luís Ferreira, Invited Professor, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa

Júri

Presidente:	Prof. José Manuel Matos Ribeira da Fonseca - FCT-UNL
Arguente:	Prof. Tiago Oliveira Machado de Figueiredo Cardoso - FCT-UNL
Vogal:	Prof. Fernando Luís Ferreira - FCT-UNL

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

May, 2019

Risk Assessment for Alzheimer Patients, using GPS and Accelerometers with a Machine Learning Approach

Este documento foi gerado utilizando o processador (pdf)LaTeX, com base no template "novathesis" [1] desenvolvido no Dep. Informática da FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

Abstract

Alzheimer is a pathology with an increasing incidence as people age. The epidemiology of the Alzheimer's disease crosses every country at later stages in life and, for that motive, has become a major concern as expectancy of life is raising in developed world.

The problem, as the disease progresses, becomes, the continuity of the person's life as normal as possible and the insurance of safety and security for that person. It is known that Alzheimer patients tend to forget about important things such as their identity and location and for that it is important to provide that they become aware of such relevant information for them and for those who could provide them support. It is also known that people with Alzheimer tend to wander and, when that happens, they can get lost and become exposed to danger.

The aim of this work is to assess if the person is getting away from usual paths and to monitor if the person falls which becomes riskier while wandering out of usual paths. The usage of GPS makes it possible to keep track of routes and, with the detection of possible deviations, it becomes possible to act accordingly, either issuing warnings for that person and later to carers and family. On the other hand, using machine learning to evaluate usual movements, it is possible to determine if a person falls or endures an excessively quiet position. With those strategies working together it is aimed to ensure safety of a person and request for assistance when high risk is assessed by the technological setup. To address these cases, the proposded setup will be based on a *smartphone*, together with a *smartwatch*, both carried by that person, as such devices already provide some needed sensors and GPS receiver while providing processing capabilities.

Keywords: Alzheimer, location, monitor, danger, GPS, paths, wander, falls, safety, *smartphone*, processing capabilites.

Resumo

Alzheimer é uma patologia com uma incidência crescente à medida que as pessoas envelhecem. A epidemiologia da doença de Alzheimer é transversal a todos os países em fases posteriores da vida e, por esse motivo, tornou-se uma grande preocupação à medida que a expectativa de vida aumenta no mundo desenvolvido.

O problema, à medida que a doença progride, torna-se a continuidade da vida da pessoa tão normal quanto possível e a segurança para essa pessoa. Sabe-se que os pacientes com Alzheimer tendem a esquecer-se de coisas importantes como a sua identidade e a sua localização e, para isso, é importante fornecer-lhes informações relevantes e também àqueles que poderão fornecer apoio. Sabe-se também que as pessoas com Alzheimer tendem a divagar e, quando isso acontece, elas podem-se perder e ficar expostas ao perigo.

O objetivo deste trabalho é avaliar se a pessoa está a afastar-se dos caminhos usuais, monitorizando se a pessoa cai, um risco acrescido quando a pessoa vagueia por caminhos fora dos usuais. O uso de GPS possibilita acompanhar rotas e detectar possiveis desvios, permitindo reagir de acordo com as circunstâncias, emitindo alertas para a pessoa e depois para os seus cuidadores e familiares. Por outro lado, usando *machine learning* para avaliar os movimentos usuais, é possível determinar se uma pessoa cai ou se se encontra numa posição excessivamente imóvel. Com essas estratégias a trabalhar em conjunto, o objetivo é garantir a segurança de uma pessoa e solicitar assistência quando diferentes etapas do processo de avaliação apontam para um alto risco. De forma a endereçar esses casos, a configuração proposta será baseada num *smartphone*, juntamente com um *smartwatch*, transportados por essa pessoa, os quais já fornecem sensores de movimento e o receptor de GPS, enquanto também providenciam recursos de processamento.

Palavras-chave: Alzheimer, localização, monitorizar, perigo, GPS, caminhos, divagar, quedas, segurança, *smartphone*, recursos de processamento.

Contents

List of Figures

List of Tables

Glossary

accelerometer	A device that measures proper acceleration. That is, the rate of change of velocity of a body, in its own instantaneous rest frame.
Android	A mobile operating system, developed by Google.
database	An organized collection of data, generally stored and/or accessed from a computer.
GPS	Short for Global Positioning System. It is a satellite positioning system that sends to certain mobile devices, their location, as long as the device is within the field of view of, at least, three satellites.
gyroscope	A device used for measuring or maintaining orientation and angular velocity.
machine learning	A subset of AI (artificial inteligence) that uses statistical techniques to give computers the ability to learn, by giving them "data" of a certain subject.
Python	An interpreted, high-level, general-purpose programming language.
sensor	A device, module, or subsystem whose purpose is to detect events or changes in its environment and send the information to other electronics, frequently a computer processor.
smart device	An electronic device that can connect to different types of wireless protocols such as Bluetooth, WiFi, 3G, 4G, NFC, etc., that can operate, to some extent, interactively and autonomously.
smartphone	A class of mobile phones, with strong hardware and software capabilities.
smartwatch	A wearable computer in a form of a wristwatch.

Acronyms

ANN	Artificial Neural Networks.
API	Application Programming Interface.
FN	false negative.
FP	false positive.
FS	false negative.
GNU	GNU is Not Unix.
GPS	global positioning system.
GUI	Graphical User Interface.
KNN	K-Nearest Neighbors.
LSTM	Long-Short-Term Memory.
SVM	support vector machine.
TP	true positive.
VSA	vector sum of acceleration.
VST	vector sum of tilt.

CHAPTER 1

Introduction

1.1 Motivation

Alzheimer's disease (AD) has emerged as a serious public health concern, as it places an immense burden on the individual, family, community, and health care resources.

AD most frequently presents its' patients with episodic memory impairment as the earliest and most prominent feature, with additional deficits in language, semantic memory, executive functioning, visuospatial abilities, and functional impairment that emerge over the disease course [10] [6].

The advances in the health domain with new diagnosis tools and new treatments lead to an improvement in quality of life and extended life span. However, as in many other aspects of life, there is a downside and that is the emergence and prevalence of age associated pathologies. Such are the cases of people with Dementia where most of the cases refer to Alzheimer patients which have reduced ability to perform mental tasks such as memorizing, reasoning and decision making. After entering the sixth decade of life, the risk of dementia increases immensely taking the form, in most of the cases, of the Alzheimer pathology. The progression of Alzheimer is different from person to person and it usually starts to manifest in episodes of loss of memory, confusion and loss of references and even self-awareness. A major consequence of such episodes is that, as they can happen anywhere at any time without previous symptoms, it may involve some type of danger for the person. Among those risks, some of major concern may be related to wandering episodes, getting lost, falling on the ground or to some dangerous place (e.g. river, cliff), being caught by a car or by another vehicle or equipment. In fact around 40–60% of falls lead to injuries (Masud & Morris 2001).

Associated to those risks, variations of the heartbeat are sometimes identified, before or as a consequence of such events. Those measurements are outside normal boundaries,

for a typical heartbeat at that age, or outside clinical values for that person. In order to address such problems several types of devices and services have been developed for people in less severe stages of Alzheimer. The existing solutions are sometimes expensive as they require buying specific equipment and paying a monthly fee to a provider of a service that will monitor and take actions after a risky situation is identified.

1.2 Objectives

The main goal of this research work, reported in this dissertation, is to analyze and present a viable solution for monitoring Alzheymer patients, keeping them safe and secure.

The adopted methodology takes as reference the fact that smartphones become pervasive objects that each individual carry for communication and other services. The design is based on using the sensor capacities to enable the detection of position and risk for a given person. In this case the internal accelerometers are used to determine if there was an event of the type that generates an increased acceleration, typically a falling person.

There are some other works where equipment was used to detect falls at home but those need a specific architecture and a deployment of specifically designed sensors (Biros et al. 2014). Some other works try to assess the most precise definition of the kind of detected event (Anjum & Ilyas 2013).

Another type of event that can be classified as risky is wandering. Most of today's smartphones carry a GPS sensor, which, together with specific algorithms can be used to detect if a certain person is outside of certain, pre-determined, areas. These can be classified as zones where the patient isn't near any dangerous sites, or places where there is some type of surveillance.

In the present work, the objective is to keep a simple solution that makes the detection of events without seeking for a precise characterization of the type of event but rather trying to get the confirmation of risky situation from the user, or by default on user's inaction.

1.3 Research questions

- How can we help improve an Alzheimer patients' sense of security, trust and autonomy?
- Can a technological solution help Alzheimer patients' relatives keep track of the patient, helping minimize the consequences of accidental falling and wandering?

1.4 Hyphothesis

By collecting sensor data on a smart-device and using certain machine learning and location algorithms to interpret the data, we can detect if an Alzheimer patient is in a risk situation, in particular, in a risk of falling or wandering.

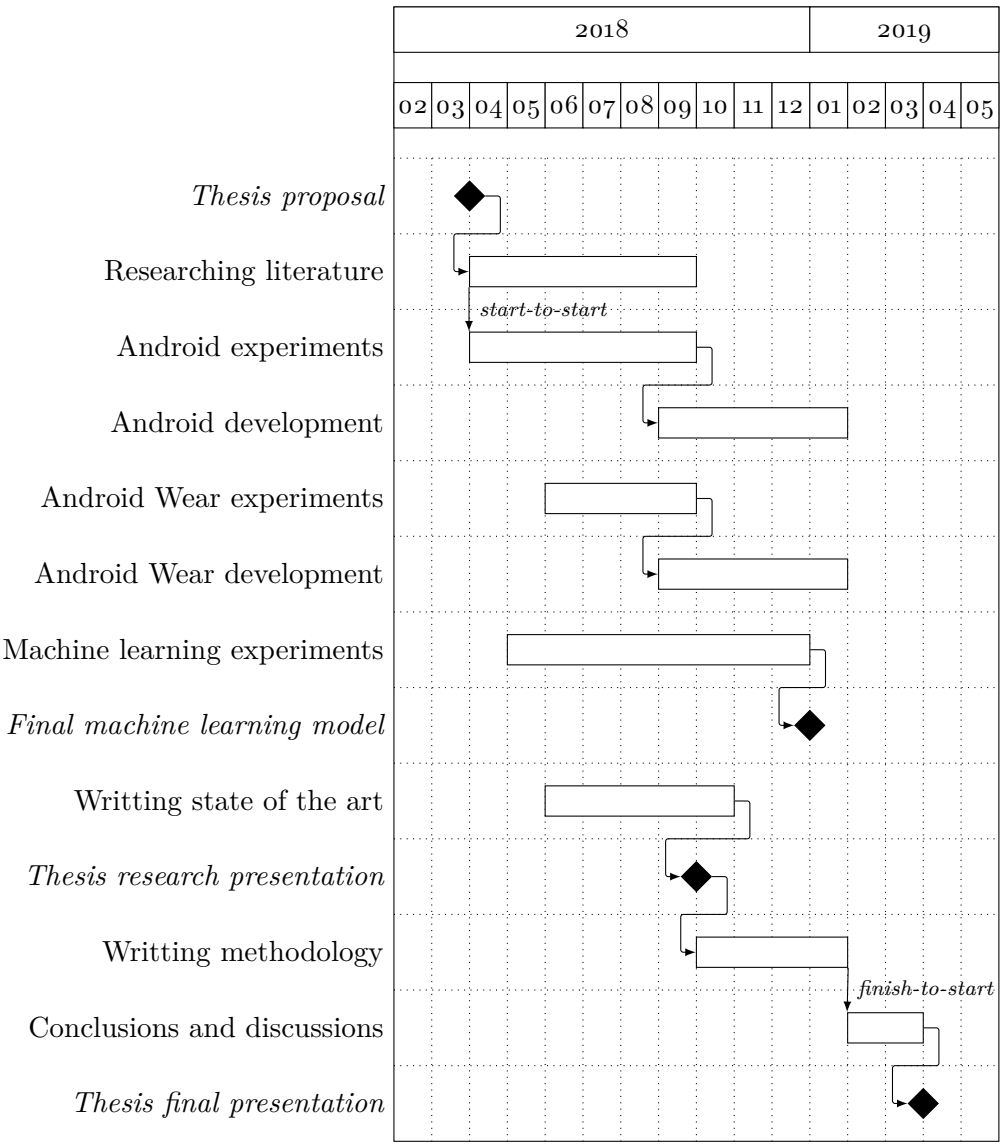
1.5 Reporting Research and Dissertation outline

1.5.1 Structure

This disstertation is divided into 6 sections, which are described bellow:

- Section 1 - Introduction: describes the main objectives of this thesis, as well as the motivation behind the project.
- Section 2 - State of the art: this section is used to present the information that is necessary to build the hypothesis. Available products are shown and described, as well as different tools that may prove useful for the thesis, as for example, machine learning algorithms and location systems.
- Section 3 - Research method: In this short chapter, the research question, hyphothesis and dessertation outline are exposed, with the thesis chronogram.
- Section 4 - Methodology and implementation: This is the practical chapter, in which the proposed method is shown, as well as the prototypes developed based on it.
- Section 5 - Results analysis: Where we discuss and analyse the results from the machine learning model created, as well as from the developed *software* and *hardware* prototypes.
- Section 6 - Conclusions: What have we learned? Do the developed prototypes help answer our research questions? What were the main problems and achievements of the thesis?

1.5.2 Chronogram



Chapter 2

State of the Art

2.1 Introduction

This chapter aims at presenting the state of the art in the scope of the dissertation. These include some of the monitoring products for Alzheimer patients (or other elderly with cognitive impairments). The objective is to identify the strongest points of each product and try to develop something around those.

It is also necessary to find some technical solutions that exist that may help us achieve what we are looking for. This work will identify some of the most important and used tools and describe in detail their way of functioning.

Finally, the dissertation will consider the relevant questions and problems of privacy, regarding monitoring systems, such as the one we are trying to develop.

2.2 Wandering

Six in 10 Alzheimer patients will wonder [40]. Even when a person with Alzheimer is within familiar places, wandering can still occur.

Some common reasons for wandering with Alzheimer patients are **confusion** (for example, when the patient doesn't realize he is at home, when he actually is, leading him to try and "find"his home), **escape from a real or perceived threat** (noise, for example, can frighten an Alzheimer patient) and **agitation** (which can occur by boredom, restlessness and a lack of exercise) [16].

Wandering can therefore occur when internal discomfort, especially when coupled with external demands (e.g., a noisy environment), exceeds the individual's threshold [22]. Also, an individual with the need for toileting assistance or the need to find a familiar safe place, may be more prone to wandering [22].

Many studies have been made on wandering patterns within Alzheimer patients, however, until now, conclusions about these travel patterns remain quite vague [25] [22].

This means it's imperative that some type of knowledge about a certain patients' routes is obtained. There are two main steps in achieving a good wandering detection system according to each patient stage of the disease:

1. For patients in early stages, where wandering becomes a risk, the learning of the usual routs is needed, so that, afterwards, if there has been a deviation, it can be detected;

2. On a more advanced stage, identifying the patients' house and/or care-giving institution perimeter.

2.3 Falling

The ocurrance of falls in Alzheimer patients is quite frequent. Cognitive impairment in one of the main risk factors [11] [19] [4].

A study involving 109 older people with cognitive impariment in residential care found several risk factors when comparing fallers to non-fallers [41].

Fall prevention in an Azlheimer patient can involve multiple preventive measures. However, prolonged restraint of patients is not one of them, as it can weaken them and predispose them to falls. Alzheimer patients should maintain physical activity for as long as possible [5]. This brings a problem: as a group more proun to falling, they stil need some sort of monitoring, as to, while maintaining some physical activity, there is still some "layer"of security.

Considering the consequences of falls, which include functional dependency, progression of the disease and death [19], this is a very serious problem, which needs to have a proper solution.

2.4 Availabe products

2.4.1 *Angelsense*

AngelSense is a child GPS Tracker and an app for the parent, designed for children with special needs. It aims at making the parents gain peace of mind knowing exactly where the child is at every moment.

It alerts for unexpected places (parents receive a text message when the child arrives and departs from a specific place and unexpected place), has live 10-second updates, listens-in (to sense the child's situation and make sure he or she is safe) and has a late departure warning (gets instant alerts if the child hasn't left school or any regularly visited place on time).

However, it's not exactly designed for the Alzheimer patient, has no fall detection and is very proprietary (needs a special device attached to the person) [14].

Figure 2.1: The *smartphone* App for *Angelsense*.

2.4.2 *GPS SmartSole*

GPS SmartSole® is a smartphone hidden and sealed in a trim-able shoe insole. It uses GPS and cellular technology, is charged about every day, and requires a service plan.

It is very discreet (the tracking device is inside the shoe sole), very unlikely to forget to bring it (who forgets to bring their shoes?), sends an alert when the battery is getting low and has geozone alerts (alerts when the person gets out of a specific perimeter, which is a must for Alzheimer patients).

But it has some negatives, such as not being waterproof, having no fall detection and the battery life being just between one and two day [13].

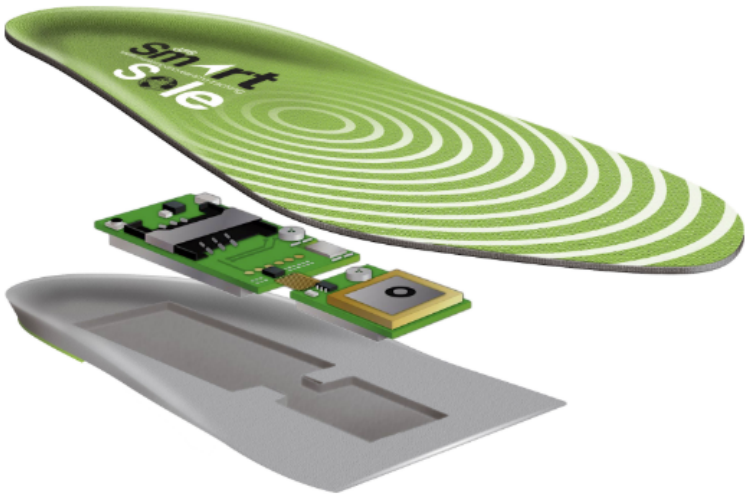


Figure 2.2: The *Smartsole*

2.4.3 *iTraq*

It's the world's smallest all-in-one tracking device that features global tracking and up to 2-months battery life on one charge.

So, it has extremely good battery life, a dedicated SOS button an accelerometer to detect different buttons (can be tuned to detect falls), a temperature sensor (can be tuned to report when it reaches a certain threshold), wireless charging, it's water and dust resistant and multiple users can track one device.

It has, however, limited applications (the device is just a simple white box with a stainless steel ring to attach it to things) and can easily be forgotten [17].

Figure 2.3: The *iTraq*

2.4.4 *Mindme*

Mindme is not a product. It's a company with a range of products and services. It was the first company to create a bespoke GPS tracking device (Mindme Locate) to locate people with dementia or learning difficulties. They introduced satellite mapping as soon as it became available because it realised that street-level mapping is inadequate when locating people off-road.

They have two products: the *Mindme Locate* and the *Mindme Alarm*. The Locate is small and simple, as it has no buttons. Carers can look up the location of the user on Mindme's website at any time. In an emergency, the carers can contact a special service that can locate the wearer and help organise somebody to collect them. The Alarm is very similar, however, it also incorporates an SOS button which, through a two way voice communication service, connects to a 24/7 Response Centre [9].

Figure 2.4: The two *Mindmne* products: the *Locate* and the *Alarm*.

However, there is no fall detection whatsoever and they're products which are highly dependent on the brand's service and "Response Centre".

2.4.5 *Pocketfinder*

Pocketfinder is a personal tracker that transmits information via Global GSM wireless networks.

You can quickly locate, monitor, and track a person by logging into an account on their website or on the free mobile app. Multiple users can see where all the devices are on Google Maps and receive alerts whenever they arrive and leave any of the unlimited number of custom geo-fence zones that can be created.

On the upside, it stores a 60-day backup of all the location data, which can be useful on some situations; has a customizable "time between locates"(the default is 2 minutes but it can go as low as 10 seconds, on the so-called "track mode"); has a "military grade"privacy

protection (a subject which which we are going to talk about on another section); and it is small, rugged and water resistant up to about 1 meter of depth.

However, and again, it doesn't have any type of fall detection system [29].

Figure 2.5: The *Pocketfinder*.

2.4.6 Products comparison

As we can see, none of the existing solutions represent a viable solution towards solving our problem. Their main problems are:

- On almost all of them, the hardware wouldn't work on those patients. It's very easy for them to forget to bring a small box or even their smartphone's;
- The solutions that do have a good device are extremely expensive. And all of them need a paid subscription;
- Only some offer fall detection, which is something we should value.

Table 2.1: The five analyzed products.

	Angelsense	*GPS SmartSole*	*iTraq*	*Mindme*	*Pocketfinder*
Fall detection	NO	NO	YES	NO	NO
GPS Geofencing	YES	YES	YES	NO	YES
Battery life (up to)	24 hours	48 hours	2 months	48 hours	96 hours
Non Proprietary	NO	NO	NO	NO	NO
Non Forgetable	NO	YES	NO	NO	NO
Monthly Fee	33$	25$	N/A	16.50$	N/A
Device Price	79$	300$	119$	80$	159$

2.5 Smart device sensors for fall detection

A smart device (be it a smartphone or a smartwatch, for example) consists, among other things, of a number of various sensors, which can be used to monitor various aspects of the outside world. These sensors can be the basis of a monitoring system that help people monitor other people, making their life easier.

In this section, we explain how the various types of sensors work in real life and how they can provide useful information for a monitoring system.

2.5.1 The accelerometer

A smart devices' accelerometer measures the rate of change in velocity with respect to a direction. It reports three accelerometer values, in a canonical form: an X-axis, a Y-axis and a Z-axis.

The accelerometer can be used to detect changes in a person's movement. For example, when a person rapidly sits down and the smartphone is in their pocket, the Y-axis might suddenly change in value.

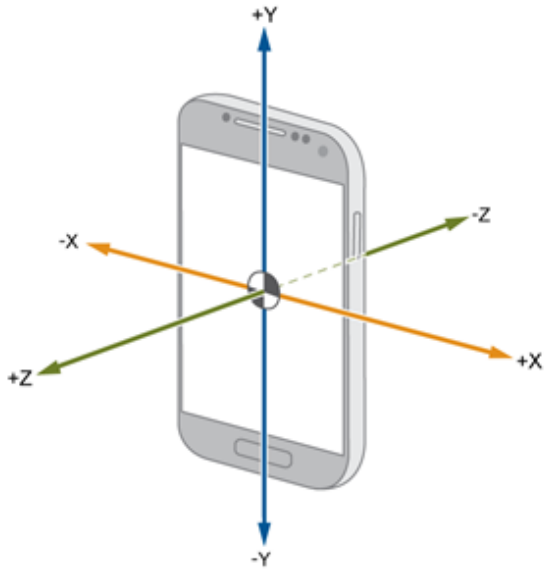


Figure 2.6: The accelerometer axes on a smartphone.

2.5.2 The gyroscope

Even tough the accelerometer can detect changes in a person's movements, some of the movements might appear very similar to the accelerometer (for example, sitting rapidly on a chair and falling from a chair).

One device that can complement the accelerometer can be the gyroscope. It can detect changes on the orientation (or tilt) of the device and can, therefore, differentiate between, for example, a sit down (which theoretically causes no change on the devices orientation) and a fall (which might cause many disruptions on the orientation).

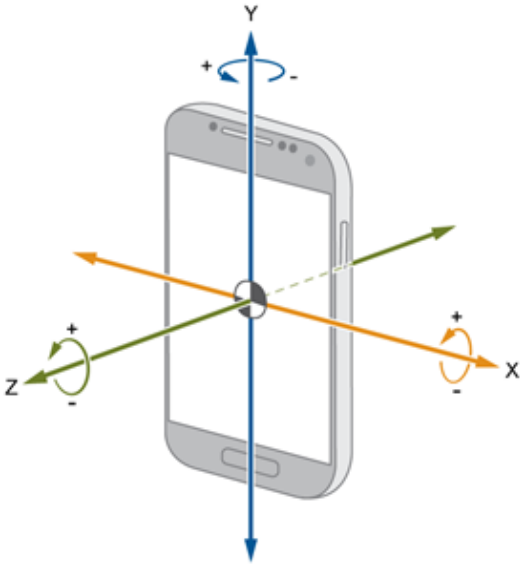


Figure 2.7: The gyroscope axes on a smartphone.

2.5.3 The heart rate sensor

A health monitoring system wouldn't be complete without some sort of health sensor. One very simple and useful sensor is the heart rate sensor. Nowadays, many can be found on certain smartphones and, especially, on smartwatches and smart-bracelets.

These sensors can detect when a person might be on a stressful situation. That heart rate climb can be a consequence of a simple run or scare, but it can also be the follow up of a fall or a more dangerous situation.

Figure 2.8: A smartwatch with a heart rate sensor.

2.6 Fall detection systems

Fall detection is extremely important on the subject of these patients. The detection can be made using a simple accelerometer, incorporated into a smartphone, but the accuracy might suffer a lot if a limited number of sensors is used.

To complement the sensors, we also need a good algorithm that can retrieve the sensors' data, correlate all of them and decide whether there has been a fall or not.

The research made presents the various advantages and disadvantages in using these algorithms, so that, later, we can decide which one could the best, taking into account our objectives.

All of these algorithms have four possible types of output, as is presented in table 2.2.

Table 2.2: The possible results of a fall detection system.

Event	A fall occurs	A fall does not occur
A fall is detected	True positive, TP	False positive, FP
A fall is not detected	False negative, FN	True negative, TN

To evaluate the performance of the system, the are some indexes defined as:

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{2.1}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{2.2}$$

$$\text{False Positive Rate} = \frac{FP}{TN + FP} \tag{2.3}$$

The performance of fall detection systems shows to be a trade-off between sensitivity and specificity [27].

2.6.1 Accelerometer only based detection

Many existing solutions take advantage of the accelerometer. But not many use it as a solo sensor to detect the falls. These solutions [12][7], usually take advantage of the vector sum of acceleration (VSA), which represents the total acceleration on all three directions:

$$VSA = \sqrt{A_x^2 + A_y^2 + A_z^2} \tag{2.4}$$

On the upside, it's an extremely simple method and one that requires very little tools, be it at a hardware level and a software level.

On the downside, because there is only one sensor used, it's very difficult to use a different technique other than the TBM one (threshold-based method), in which we define a threshold to the VSA value and, when that value reaches that threshold, we say there has been a fall.

Unfortunately, there are some critical flaws to this method. In reality, some ADL's (activities of daily living), which do not relate to falls, might provoke a spike on the VSA value. This is not good, as it can lead to many false-positives. Those can be, for example, a quick sit down or a jump. These are not falls and might the interpreted as being one, by this algorithm.

2.6.2 Accelerometer and gyroscope based detection

Another, less practical but more accurate solution might be to complement the use of the accelerometer with the gyroscope [31][26].

In addition to (2.4) we also need another formula which represents the total tilt on all three directions. We shall call it the VST (vector sum of tilt):

$$VST = \sqrt{G_x^2 + G_y^2 + G_z^2} \tag{2.5}$$

The tilt is used to measure the posture, whether a person is standing or falling with the assumption that those who fell will be on the floor (lying or facing up). It can be used to determine whether the user actually fell or stood up quickly.

This type of algorithm usually yields good results as observed by other studies and can be used as a good basis for our system.

2.6.3 Fall detection with machine learning

A machine learning fall detection system thoroughly analyzes the movement of a human to detect if a fall event has occurred, or is occurring. The purpose of fall detection systems is to reduce the severity caused by the impact of the fall, by notifying medical care or a close relative [27].

Today, these algortihms are used among a variety of areas and are increasingly being used in the medical and monitoring field.

There are many types of algorithms that use machine learning to classification. There are also many tools that use these algorithms. We will concentrate on the two most used and known tools: *Weka* and *Tensorflow*.

There are many different types of algorithms for machine learning. We will take a look at some of them:

- Support Vector Machines
- Markov models
- Naïve Bayes and Gaussian mixture models
- K-Nearest Neighbour
- Decision Trees
- Artificial neural networks

"Support Vector Machine"(SVM) is a supervised machine learning algorithm, which means it uses inputs and outputs to learn the mapping function. Even tough it can be used for both classification and regression, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular

coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well, as can be seen in figure 2.9 [39].

Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

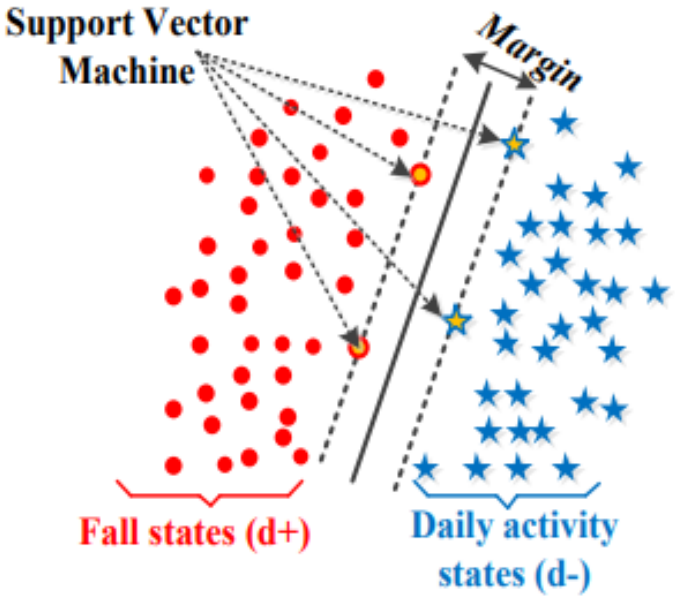


Figure 2.9: Description of the SVM algorithm with hyperplane (Nguyen Thanh Hai1 / Trinh Hoai An 2016).

"Naive Bayes" is a classification technique that assumes that the presence of a particular feature in a class is unrelated to the presence of any onther feature [1].

It is easy to build and known to outperform other more sophisticated classification methods.

This thereom is based upon the calculation of a probability called "Posterior probability"(also as the **"Bayes' Theorem"**), shown in equation (2.6).

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \tag{2.6}$$

Where:

- $P(c|x)$ is the posterior probability of *class* $(c, target)$ given *predictor* $(x, atributes)$;
- $P(c)$ is the prior probability of *class*;
- $P(x|c)$ is the likelihood which is the probability of *predictor* given *class*;
- $P(x)$ is the prior probability of *predictor*.

The posterior probabilty can be calculated by constructing a frequency table for each atribute against the target. Aftwerwords, we transform that table to a likelihood table and use equation (2.6) to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome prediction.

On the upside this method is easy to implement, very fast and needs very little training data to perform well. On the downside, if a categorical variable has a category (in the test dataset), which was not observed in the training dataset, the model will assume a 0 (zero) probability and will be unable to make a prediction.

"K-Nearest Neighbors", or abreviated KNN, is an aproach to data classification that estimates how likely a data point is to be a member of one group or the other, depending on what group the data points close to it are in.

It is considered an "lazy-learner"algorithm. That's because it doesn't build a model using the training set until a query of the data set is performed.

The "distance"from the other points is a problem with various solutions, as shown in equation (2.7) through (2.9).

Eucladian distance on d dimensions:

$$J_e[k,l] = \sqrt{\sum_{i=1}^{d}(x_{ik} - x_{il})^2} \tag{2.7}$$

"City-block"distance:

$$J_{cb}[k,l] = \sum_{i=1}^{d}|x_{ik} - x_{il}| \tag{2.8}$$

Minkowsky distance:

$$J_M[k,l] = \left[\sum_{i=1}^{d}(|x_{ik} - x_{il}|)^s\right]^{1/s} \tag{2.9}$$

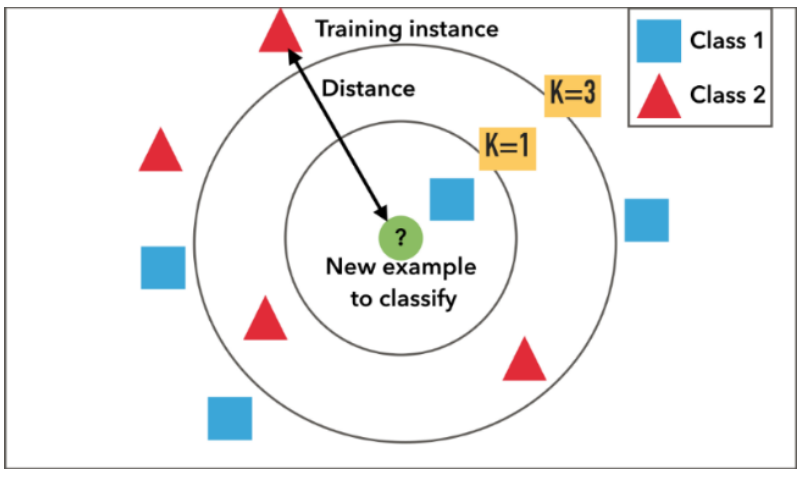


Figure 2.10: Description of the KNN algorithm (Adi Bronshtein 2017).

One of the more common type of algorithms for machine learning are the **Artificial Neural Networks**, or, for short, ANN's. They are an information processing system inspired by bioogical neural networks [33].

Each neural net has a large number of simple processing units called neurons, cells or nodes. Each neuron is connected to another one by directed links which have an associated weight to it. When a signal passes through, it gets multiplied by their respective weights. A neuron is connected with multiple neurons, all the signals are added up and then there is an activation function which checks if the added signal is greater than some threshold value, if it is greater then the signal is fired otherwise it is not [34].

Figure 2.11 shows a generic artificial neural network, with three layers: an input layer, a hidden layer and an output layer.

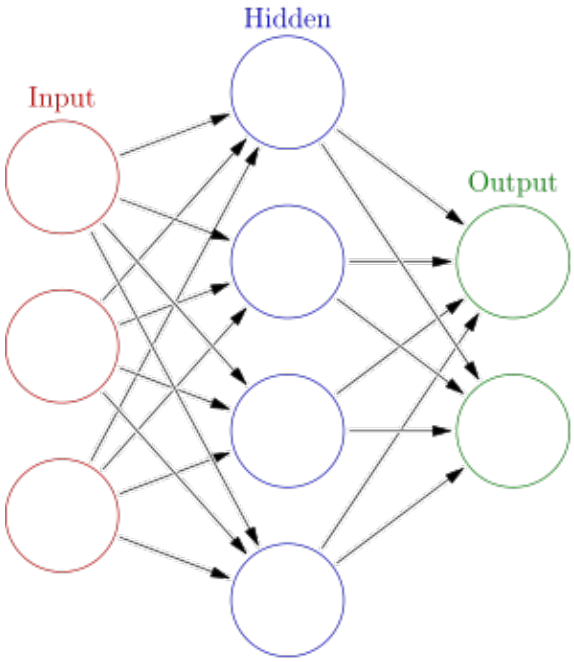


Figure 2.11: An artificial neural network (ANN).

For example, let's consider a neuron B that receives inputs from A1, A2, A3 and A4. Also, the weights of connection from A to B are w1, w2, w3, w4 respectively as shown in the diagram from figure 2.12. Activation function is the logistic sigmoid functio (S-shaped curve), shown in equation 2.10.

The signal after passing through activation function then it passes to another neuron C1, C2 via v1, v2 weights respectively. Training of neural network deals with finding optimal values for weights of connection links.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.10}$$

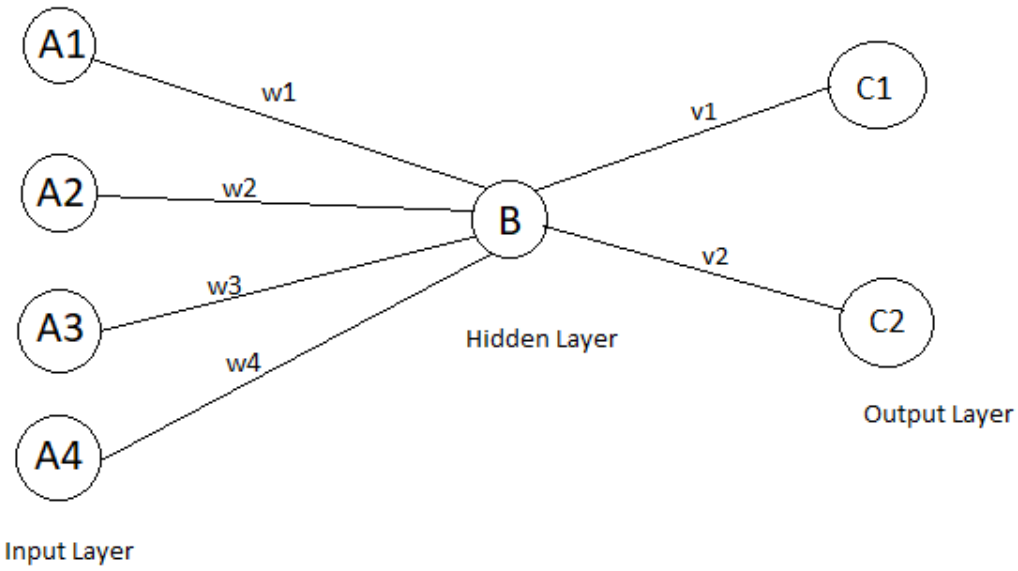


Figure 2.12: An artificial neural network (ANN) example.

LSTM neural networks (short for **Long Short-Term Memory** neural networks) are a type of recurrent neural networks that are getting quite popular within machine learning.

LSTM networks have a type of cells which can, theoretically, store memory as long-term or short-term memory cells (hence the name, "Long Short-Term Memory"). The ouput is then modulated by the state of these cells [36].

For example, imagine you want to classify what kind of event is happening at every point in a movie. It's unclear how a traditional neural network could use its reasoning about previous events in the film to inform later ones [38]. Recurrent neural networks address this issue.

In an extrememlly short sentence: LSTM networks are networks with loops in them, allowing information to persist [38].

An LSTM is well-suited to classify, process and predict time series given time lags of unknown size and duration between important events [38].

Their many usages are:

- Text generation;
- Handwriting recognition;
- Handwriting generation;
- Music generation;
- Language translation;
- Image captioning.

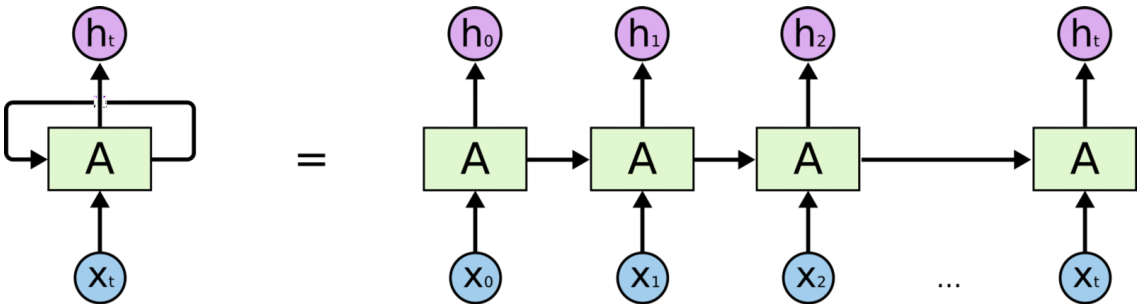


Figure 2.13: An LSTM loop.

2.6.4 Machine learning tools

2.6.4.1 *Weka*

Weka is a GUI (Graphical User Interface) based machine learning program that contains many types of algorithms for classifying datasets. It's extremelly intuitive and easy to use.

It provides many regression and classification tools, can read many types of files (including databases) and is free under the *GNU General Public License*.

It can, however, be a little constrained on the functionality side. Also, it wasn't designed for very large or very complicated datasets.

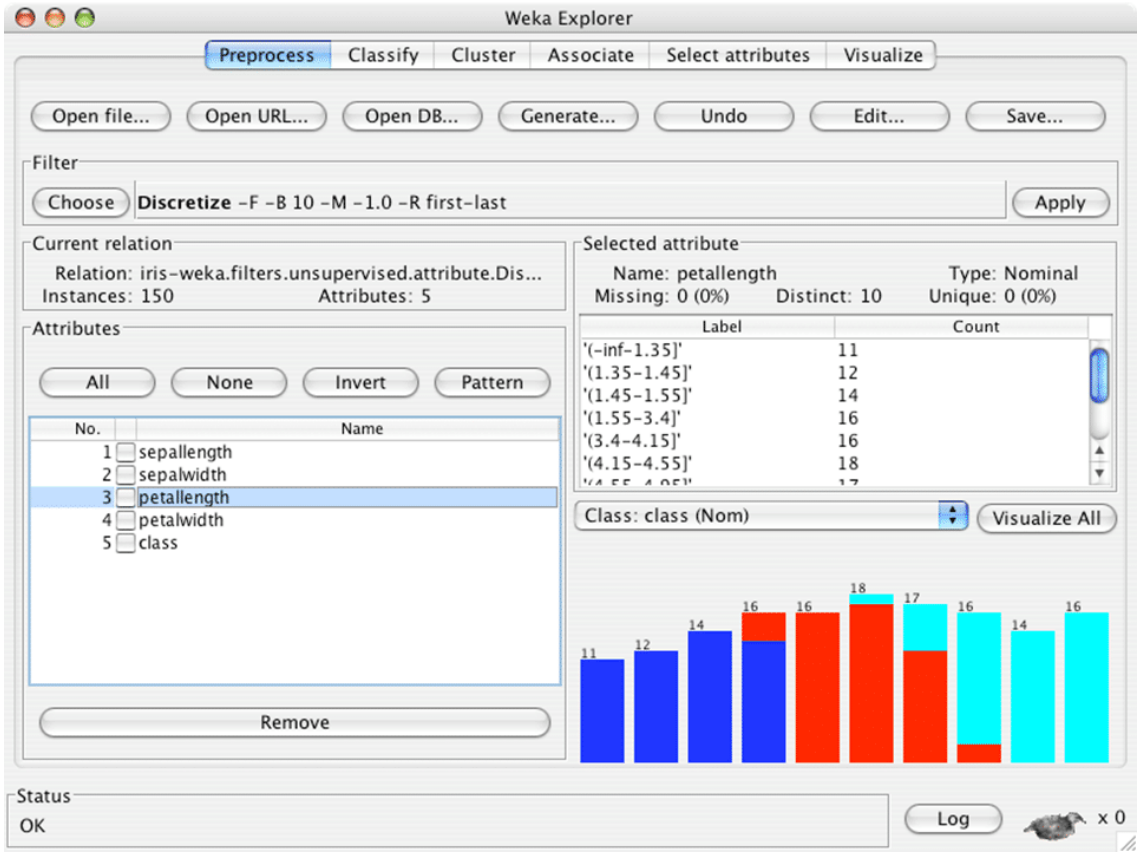


Figure 2.14: The Weka GUI.

2.6.4.2 *Tensorflow*

Tensorflow differs quite a lot from Weka. It's a programatic based machine learning tool. Actually it's more of a library for the Python programing language. It's highly modular and modifiable and it's commonly used for more complicated algorithms and datasets [3][2].

We need to store all the input data, output data and data for all the variables (like the weights) on some type of arrays. Tensorflow uses tensors.

Tensors are the data units which flow through the computational model of a neural Network. A tensor is just any array of data which can take up any number of dimensions. Therefore, in a tensor, we can store any number of attributes in a very organized manner. The number of dimensions of a tensor is known as it's rank [35].

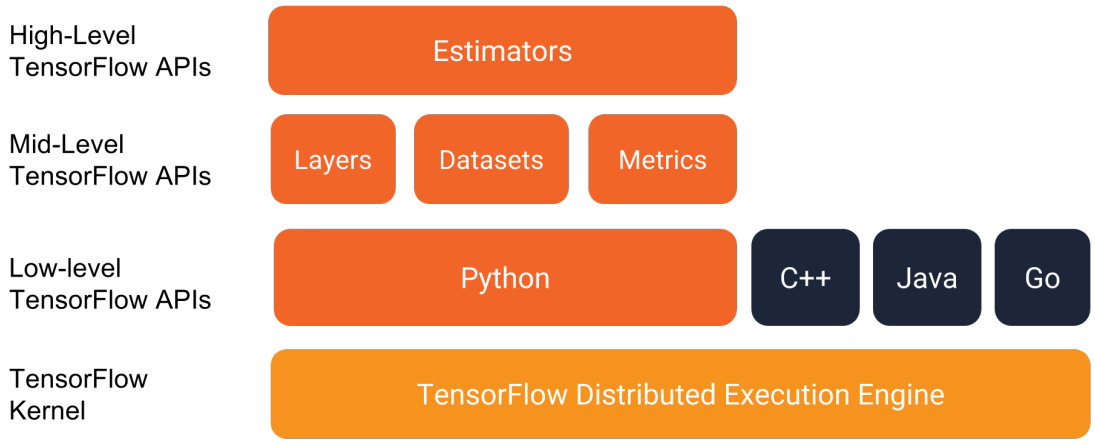


Figure 2.15: Tensorflows' API layers (Tensorflow 2018).

For classification we need an **Estimator**, which is, essencially, Tensorflows' representation of the entire model. It handles initialization, saving, restoring and many other features. An Estimator is any class derived from `tf.estimator.Estimator` and Tensorflow already has many pre-made ones [30].

We then need to create input functions to supply data for training, evaluating and predicting.

The dataset is stored in a `tf.data.Dataset` object which has two elements:

- *The features* - A Python dictionary in which each key contains the name of the feature and and value is an array containing all of that feature's values.
- *The label* - An array containing the values of the label for every example.

When we build an estimator model, we pass it a list of feature columns that describes each of the features you want the model to use. The `tf.feature_column` module provides many options for representing data to the model [30].

Tensorflow provides several premade classifier estimators, like, for example:

- `tf.estimator.DNNClassifier` - designed for deep models that perform multi-class classification.
- `tf.estimator.DNNLinearCombinedClassifier` - for wide and deep models.
- `tf.estimator.LinearClassifier` - for classifiers based on linear models.

Training of the model begins by calling the `classifier.train` method.

Then, we evalute the model by calling the `classifier.evaluate` method, which can evaluate the exact accuracy of it.

After creating and evaluating our model, we can save (and therefore, later restore) the model through checkpoints, which are versions of the model created during training [8].

2.7 Location monitoring

When researching about tracking tools for elderly, several types of application domains are found. Some focus on the tracking and monitoring at their homes, others on a specific part of the home and others simply do about monitoring their health. As we discussed in the Introduction section, one of the main conditions of the Alzheimer disease is confusion and loosing notion of time or place. Therefore, we focus on monitoring people outside their houses as dangerous situations can occur if they forget where they are or how much time has passed since they left their houses.

There are many ways we can make the location monitoring for an Alzheimer patient possible. We can either develop an intirely new device from scrath or use an existing one, from which we can develop.

Probably the most sensible choice is to use a smartphone for this purpose. The choice for this equipment was based on its GPS functionality, its low cost and also on the possibility to develop an application that can easily provide the ability to configure the system's usage. By carrying a mobile device, the Alzheimer patient has a way to be monitored as its position is acquired through GPS satellites and sent to a Web based application and/or the respective App. Caregivers or healthcare professionals can then access the Web based application and/or the respective App, anywhere and anytime, to know where the patient is at the current moment or to check past locations [28].

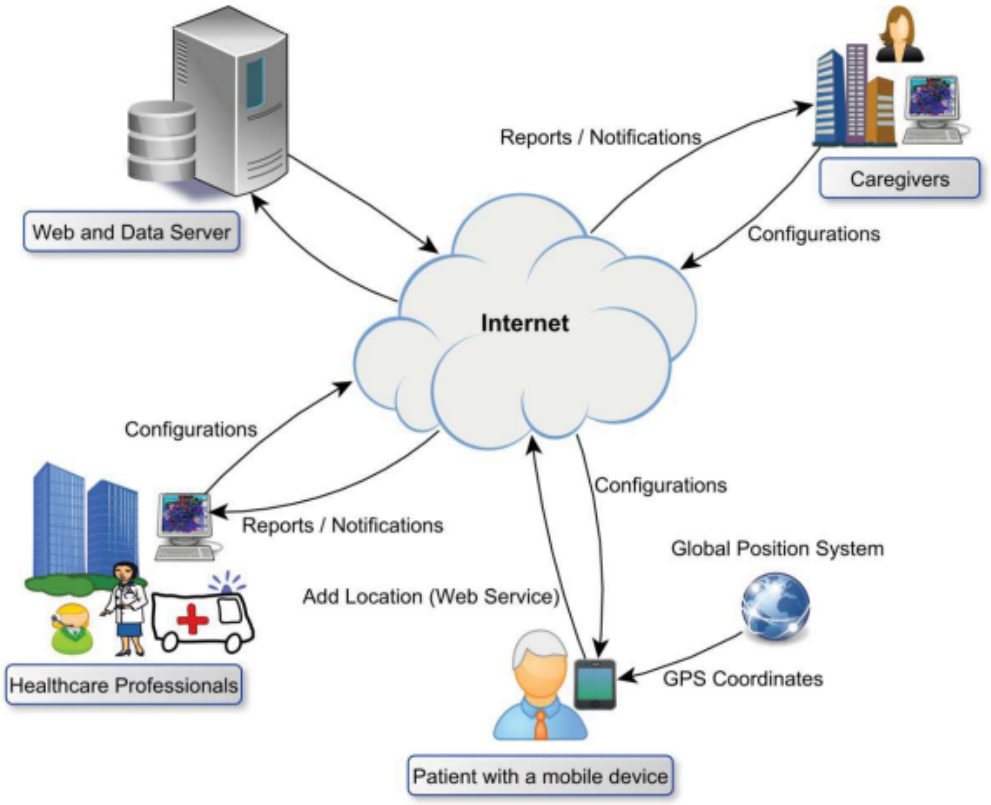


Figure 2.16: An architecture for a GPS tracking solution (Sara Paiva and Carlos Abreu / Procedia Technology 5 2012).

2.7.1 The GPS sensor

The GPS is a satellite based navigation system developed by the U.S. Department of Defense for military purposes [15]. It allows locating mobile devices at any place, any time on Earth using trilateration with range measurements between an observer and a few visible satellites [15]. Even tough the military GPS version is able to provide more accurate positioning data, its public version is limited to an accuracy of upto 5 to 10 meters [21]. This accuracy level can also be achieved with low-cost GPS receivers, such as the ones integrated in smartphones, when locked onto the minimum number of satellites [32]. The functionality of GPS is, though, limited to outdoor positioning as line of sight to the satellites is required [21]. Although the prevalence of GPS receivers has dramatically driven down cost, size, and power requirements [15], GPS sensing is, compared to other location technologies, rather resource intensive [18]. This means that an absolute accurate location monitoring of someone is impossible. We need some auxiliary tools to help us in this mater.

2.7.2 Geofencing

One way we can mitigate some of the accuracy problems is through **geofencing**. Through geofencing we can detect if a person enters or leaves some pre-determined areas that are called geofences, which are pre-defined geographic areas.

To achieve this, the user needs to be continuously tracked in the background, that is, even when the mobile device is idle or while it executes other applications. Geofencing and background tracking thus enable a broad range of new applications, especially in the area of information relevance. However, this method poses a considerable problem, as their introduction can lead to an increase in battery consumption and privacy concerns of the user [20].

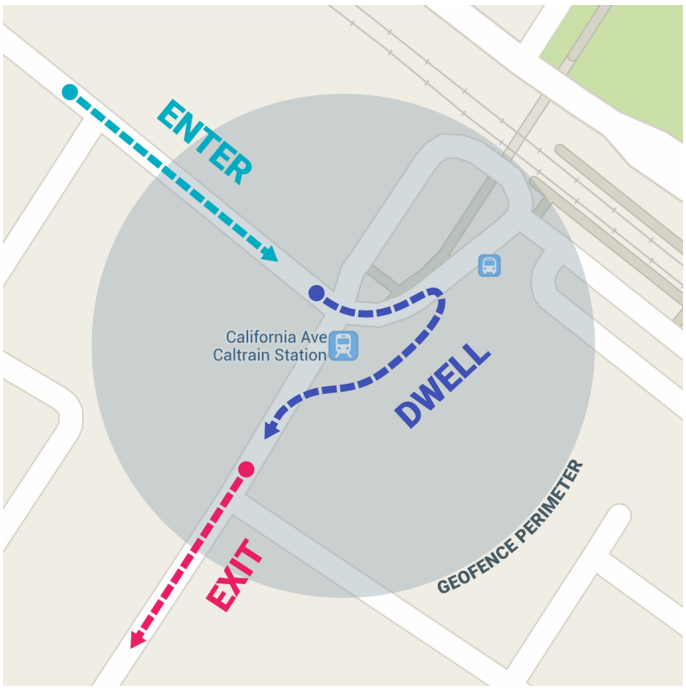


Figure 2.17: A geofence.

2.8 Privacy

Privacy has become a central topic, these days, especially in the tech and software domains (EC, DG Internal Market, Industry 2018). People don't like to be excessively monitored and observed. For Alzheimer patients, the same holds true in particular for people with mild conditions, in early stages, when a free and normal life is desired. It becomes imperative that sensitive or private data is not disclosed without the consensus of the person or unless the system detects an almost certain risk to the patient. That is a central aspect regarded in the present work taking privacy and ethical concern as central in the development of the Application. This platform has an operational difference when regarding to many solutions on the market that depend on a subscription service and, additionally, a proprietary device

to be bought and that is not open source and can only be adopted for use with the seller company or subscription provider (ITraq 2018) (AngelSense 2018).

We can consider, for instance, common applications to be used on a daily basis, like Google Navigation. If the user wants to go somewhere using this app to a calendar appointed place, the app has to access the current GPS location and calendar, potentially tracking movement and monitoring position all the time. Such data, when disclosed or shared to other users becomes a privacy issue for a person running a normal life.

At all times, we should use encrypted authentication, both on the patient and the "person who monitors" side and only disclose location data once the person is outside usual paths of displacement, experiences a fall or a persistent abnormal HR is detected, even in those cases actions are executed to track the person only if there is no active and rational reaction from the user side. It is ensured that no other data is sent to the database, or to other individuals.

Processing and decision should be taken locally without the need of a cloud service or other remote services for data analysis and decision taking [24].

CHAPTER 3

Implementation

3.1 Architecture

In this chapter, the general architecture of the system within this dissertation is presented. The present architecture was defined as a support and demonstration for the scientific results developed in the present research work. We first need to acknowledge the main elements of our system, which are:

- The smartphone (which contains the various sensors);
- The smartwatch (for notifications to the patient);
- A remote database (to store users and some data).

3.1.1 Fall detection

For this dissertation, it was decided to use a machine learning algorithm for the detection of the falls. It was chosen for its accuracy and ability to differentiate between falls and other situations.

Due to us having a relatively small number of fall cases, it was decided to use an Artificial Neural Network (ANN, for short). They work well with a relatively low number of examples and inputs. An LSTM network was considered, but deemed a bit overkill to develop what was necessary.

The tool that will be used to create and train the model is Tensorflow, which was discussed in the State of the Art chapter. It is a very powerful tool, extremely customizable and easily implemented on Android.

3.1.2 Location monitoring

To detect and monitor the patients' location, it was decided to use the cellphones' GPS sensor in conjunction with, at least, the Geofencing algorithm to limit the patients' permitted area/s.

3.1.3 Interfaces

The whole project will be developed using an Android smartphone and an Android Wear smartwatch. As described on the State of the Art section, a smartphone (and smartwatch) already possesses a lot of sensors that are very useful for the project. Unless some future work is done to optimize (and/or commercialize) the system here developed, there is no need to use custom hardware.

3.1.4 Database

All the user data, sensor data and event data will be stored in a database that allows us to send it to various devices and alert pre-determined users of some events, be it a fall or a wandering alert.

It was decided to use a Firebase database, as it is very easily and effectively implemented using a wide range of devices (Android devices, Web pages, etc.).

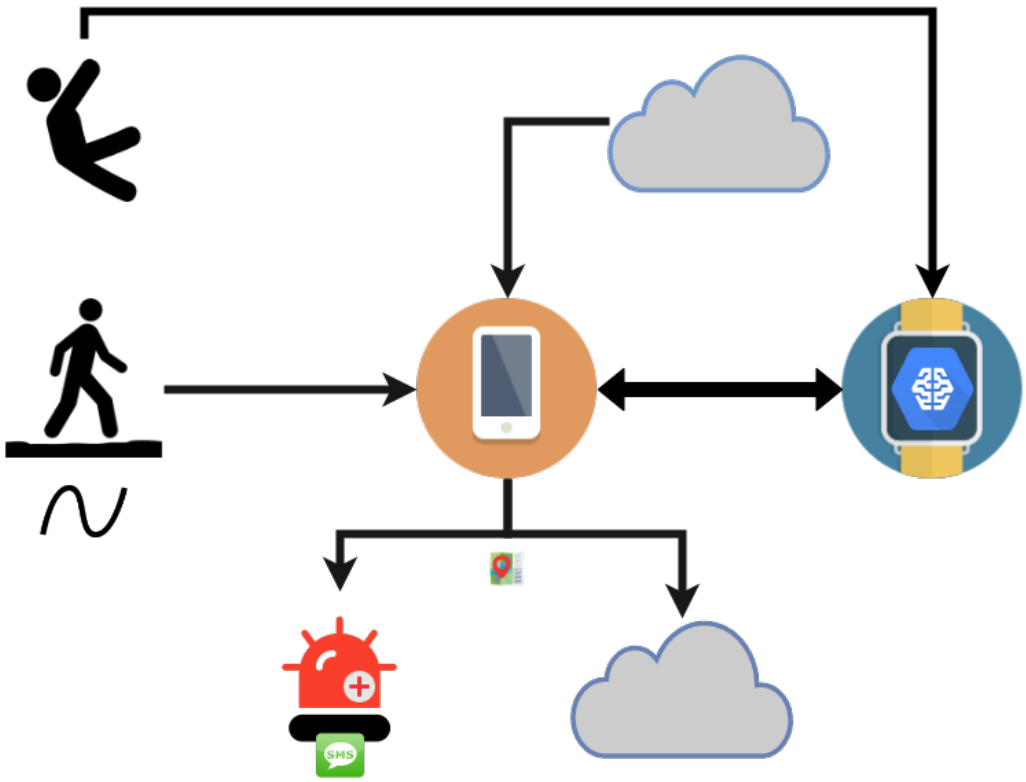


Figure 3.1: The proposed architecture for the dissertation project.

3.2 Requirements and functionalities

The requirements and functionalities were established in order to define and implement an architecture for deploying solutions for the person with Dementia applying the scientific results presented earlier in the previous chapters of this document. These are described bellow:

Requirements:

- **The architecture of the system must take privacy into account:** privacy is a very important matter nowadays. People don't want to be excessively monitored and want a free and normal life. Sensitive or private data mustn't be disclosed without the consensus of the person unless the system detects an almost certain risk to the patient.
- **The architecture should be scalable:** As more users want to interact with the system, the applications and databases should be able to accommodate those needs. Also, it should make possible that, in the future, more platforms adopt the same system with relative ease.
- **The architecture must have fail-safe guards:** Even tough today's technology for sensors and data acquisition has increasingly higher levels of accuracy and reliability, it is still possible that sometimes those mechanisms don't work as expected. Safeguard scenarios and systems must be implemented.

Functionalities:

- **Database should only be accessed by allowed devices:** not only encryption is required but also, only devices within the ecosystem can access selected users data.
- **GPS location algorithms must take into account battery usage:** as location algorithms have a tendency to use a considerable amount of power, their activation and deactivation should take into account not only the actual location of the person but also the type of patient that is using the system (medical history, for example).
- **The patient should be able to indicate if they are feeling safe:** Even tough the patient is being monitored, not all critical detected events should be considered, as there are always some false-positives. In these cases, the patient should be able to interact with the system, by indicating there is no action needed by the caregiver or family member.

- **The smartphone must be the one to interact with the data services:** to ensure that the system works with a variety of devices, only the smartphone (and not the smartwatch) must interact with the data services, as only some (and more expensive) smartwatches contain data plans.

3.3 ANN machine learning model

A machine learning model is created in about five phases:

1. Dataset gathering;
2. Enumeration of characteristics;
3. Definition of a model;
4. Training the model;
5. Model evaluation.

3.3.1 Dataset gathering

The ANN model needs some examples for two purposes: (1), training it; and (2), its' final evaluation.

As there were some fall datasets available online, it was decided to use an existing one, rather than create one from scratch, which, using simple smartphone sensors in a non-controlled environment, could also lead to poor results.

The dataset chosen was the *UR Fall Detection Dataset*, by Bogdan Kwolek and Michal Kepski, for the *Human fall detection on embedded platform using depth maps and wireless accelerometer* paper [23]. This dataset consists of 30 fall examples in a variety of situations, using specific accelerometer devices.

These examples are given with the 3 axis coordinates of the accelerometer, but also the sum of acceleration (as seen in equation 3.1), which we will use to determine certain characteristics.

$$VSA = \sqrt{A_x^2 + A_y^2 + A_z^2} \tag{3.1}$$

A fall-like event has commonly an acceleration peak of magnitude greater than 3g. The threshold of 3g has been proven that it is small enough to avoid false negatives. Next in Figure 3.2 it is presented a graphical representation of acceleration in a fall.

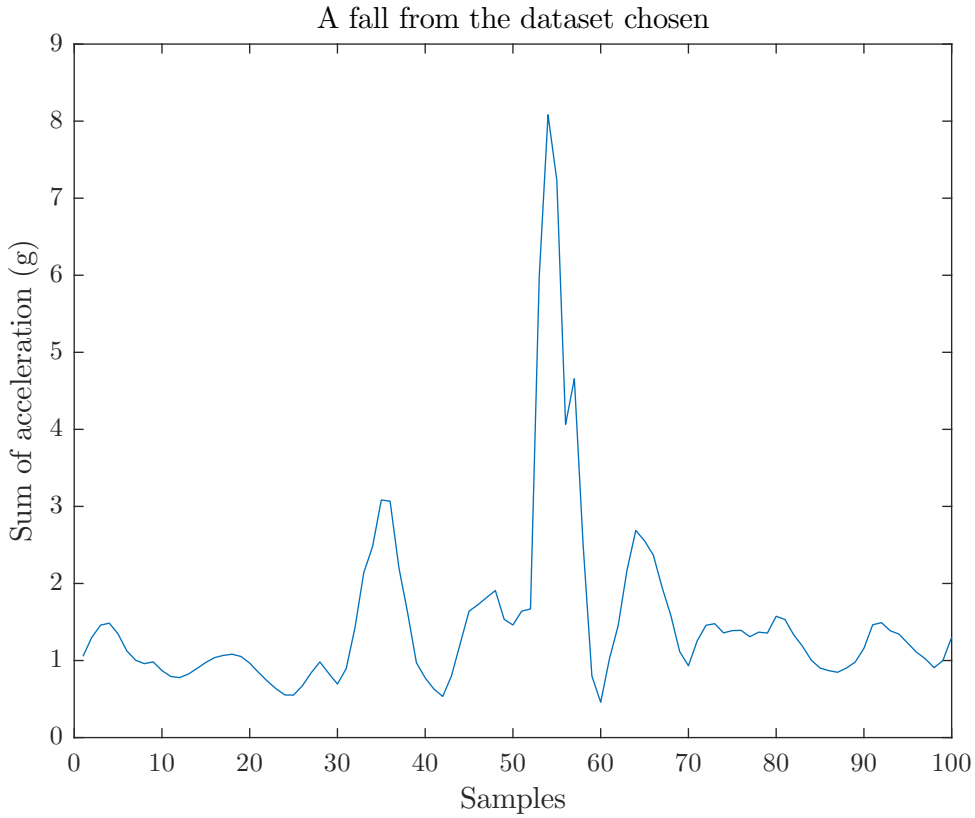


Figure 3.2: A fall from the dataset chosen for this dissertation.

3.3.2 Feature retrieval

Even tough we already have some fall examples, we need to more accurately define what really is a fall.

A possible path to train the machine for fall detection would be to send, as inputs, all 3 axes of the accelerometer, with all time-step points. That would provide, for each training/testing example, a matrix of about 200 points by 3 features (x, y and z). The problem with such approach is the fact that it requires very large amounts of data that will correspond to a very long time to train the machine. On the other side, if we only have a limited amount of training examples, there is a risk of overfitting.

Another reasonable approach would be to try to obtain certain key characteristics of the fall/non-fall and try to find a pattern based on those selected features [37].

As we can see from figure 3.2, we can note some aspects of the fall. The one that stands out the most is peak that occurs around sample number 35. This peak tells us, with a certain degree of certainty, the exact moment of the fall.

However, just this is not enough. A simple drop of a cellphone could lead to the same peak. We need more characteristics.

As a person falls, there are other, smaller peaks, that could also be of interest. Also, for a brief moment, when the person is still falling, the value of the sum of acceleration

might drop slightly below the unity value.

First, we take two auxiliary features:

- **FAM** - the First Aceleration Magnitude value, which indicates the time of the first acceleration that exceeded 2.5 G's, but not the PEAK threshold of 3.5 G's;
- **LAM** - the Last Acceleration Magnitude which indicates the time of the last acceleration that exceeded 2.5 G's, but not the PEAK threshold of 3.5 G's.

With some experimentation, it was decided to use the following characteristics for the definition of a fall. The model will be trained and tested whith these features:

- **PEAK** - the maximum value of the sum of acceleration during all the samples retrieved from the fall;
- **DT** - duration, in time, between the FAM and LAM;
- **DP** - number of intervals of time in which the value of acceleration exceeded the PEAK threhold, which is 3.5 G's.

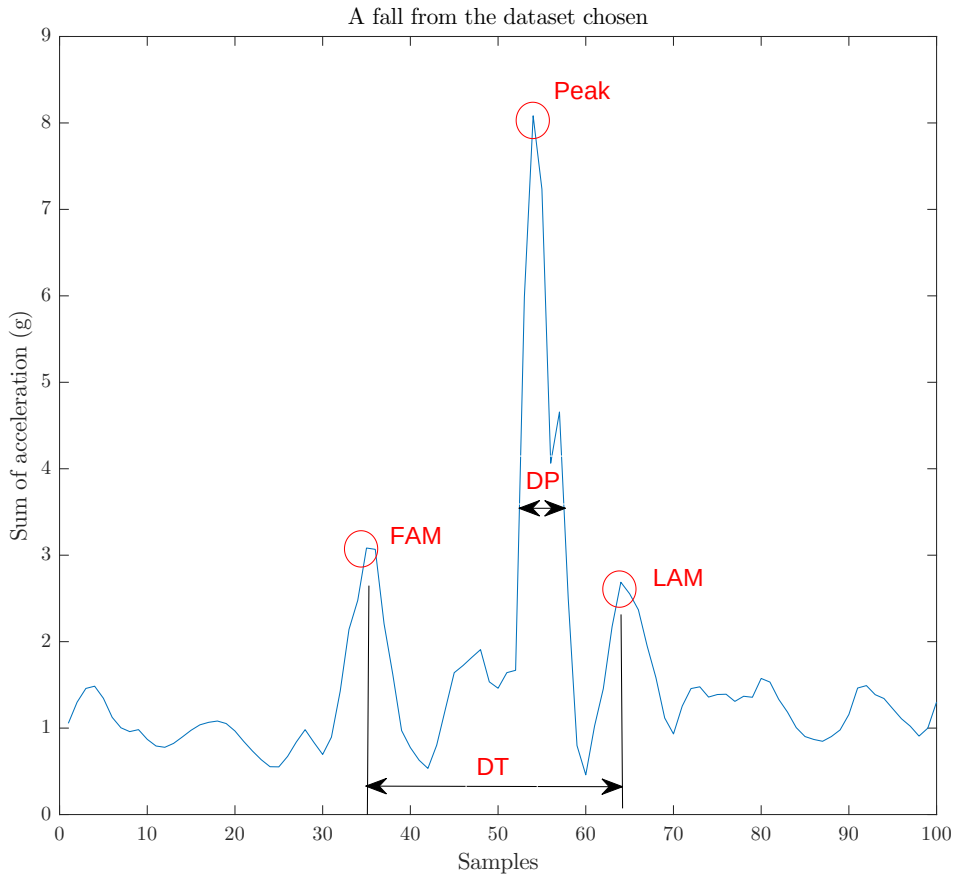


Figure 3.3: A fall from the dataset chosen for this thesis with the characteristics.

As the dataset chosen doesn't have these features for each fall, we must determine them, create a new dataset and then use that one to train and test the model. To do that, it was decided to use *Matlab*.

The peak of a fall is retrieved in a rather simple manner:

```
[PEAK, peak_t] = max(csv_actual(1:size(csv_actual),2));
```

Then, we need to calculate the DP value. For each iteration of the time sequence of the fall, we must check if the value of acceleration exceeds the PEAK threshold of 3.5 G's, then increment the DP value:

```
if csv_actual(i,2) > PEAK_THRESHOLD
        DP = DP + 1;
end
```

To calculate the FAM value, we need to ensure the current value is above 2.5 G's, below 3.5 G's and that it's the first value to fulfill those requirements:

```
if csv_actual(i,2) > LAMFAM_THRESHOLD && csv_actual(i,2) <= PEAK_THRESHOLD
    && FAM_CHECK == 0
        FAM = csv_actual(i,2);
        FAM_CHECK = 1;
        FAM_t = i;
    end
```

For the LAM value, as it's the time of the last value that exceeds 2.5 G's, we jus invert the condition, the calculate DT, by subtracting the LAM and FAM times:

```
if csv_actual(i,2) > LAMFAM_THRESHOLD && csv_actual(i,2) <= PEAK_THRESHOLD
    && FAM_CHECK == 1
        LAM = csv_actual(i,2);
        LAM_t = i;
        DT = LAM_t - FAM_t;
end
```

3.3.3 Model definition

The machine learning model will be defined using Tensorflow.

First, we need to import the dataset:

```
data = pd.read_csv('all.csv', sep=",", names=["PEAK", "DT", "DP", "CLASS"])
```

Then, we will shuffle the data, to make sure the test and training data have mixed examples:

```
data = data.sample(frac=1).reset_index(drop=True)
```

Before defining the model itself, it is necessary to tell the model what are the features and what are the classes:

```
all_x = data[["PEAK", "DT", "DP"]]
min_max_scaler = preprocessing.MinMaxScaler()
all_x = min_max_scaler.fit_transform(all_x)
all_y = pd.get_dummies(data.CLASS)
```

It's also necessary to split between test and train data. After some testing it was decided to use one third of the data for testing purpuses:

```
train_x, test_x, train_y, test_y = train_test_split(all_x, all_y, test_size
    =1 / 3)
```

The model will have 3 hidden layers, each with a certain number of neurons. The values were obtained by extensive testing. The learning rate was decided on 0.01, as it made the training relatively fast and didn't make the final values undefined.

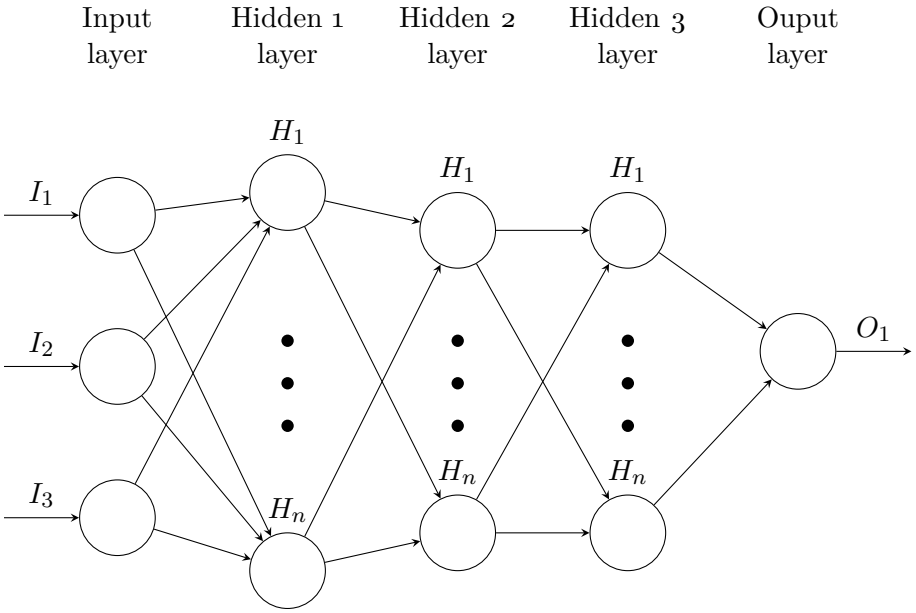


Figure 3.4: The neural network.

Where:

- I1 is the **PEAK** input;
- I2 is the **DT** input;
- I3 is the **DP** input.
- The n value for the **first hidden layer** is 10;
- The n value for the **second hidden layer** is 20;
- The n value for the **third hidden layer** is 10;

3.3.4 Model training

The model training will be done in 6000 epochs, as it resulted in very good results and didn't take very long to finish. The training loss and accuracy is also registered, to better understand the evolution throughout the epochs.

```
training_epochs = 6000

history = dict(train_loss=[],
                       train_acc=[],
                       test_loss=[],
                       test_acc=[])

for epoch in range(training_epochs):
    _, acc_train, loss_train = sess.run([optimizer, accuracy, cost],
    feed_dict={x: train_x, y: train_y})
    _, acc_test, loss_test = sess.run([optimizer, accuracy, cost], feed_dict
    ={x: test_x, y: test_y})

    history['train_loss'].append(loss_train)
    history['train_acc'].append(acc_train)
    history['test_loss'].append(loss_test)
    history['test_acc'].append(acc_test)
```

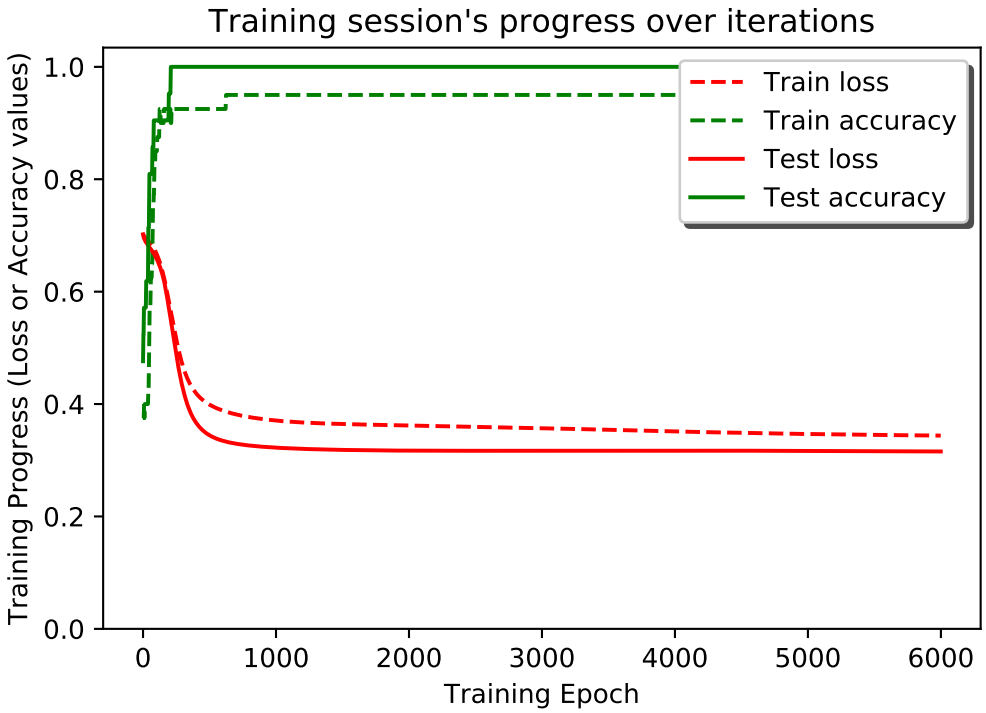


Figure 3.5: The training progress.

3.3.5 Model evaluation

Figure 3.6 presents the confusion matrix during the testing of the model. Again, the results seem to fit the purposed objectives, with a theoretical accuracy of 100%. In terms of research this kind of values live researchers with doubts as certainty is mostly associated with lack of intensive testing. That is the case for the present research where public datasets are scarce and some of them are aimed to other specific purposes.

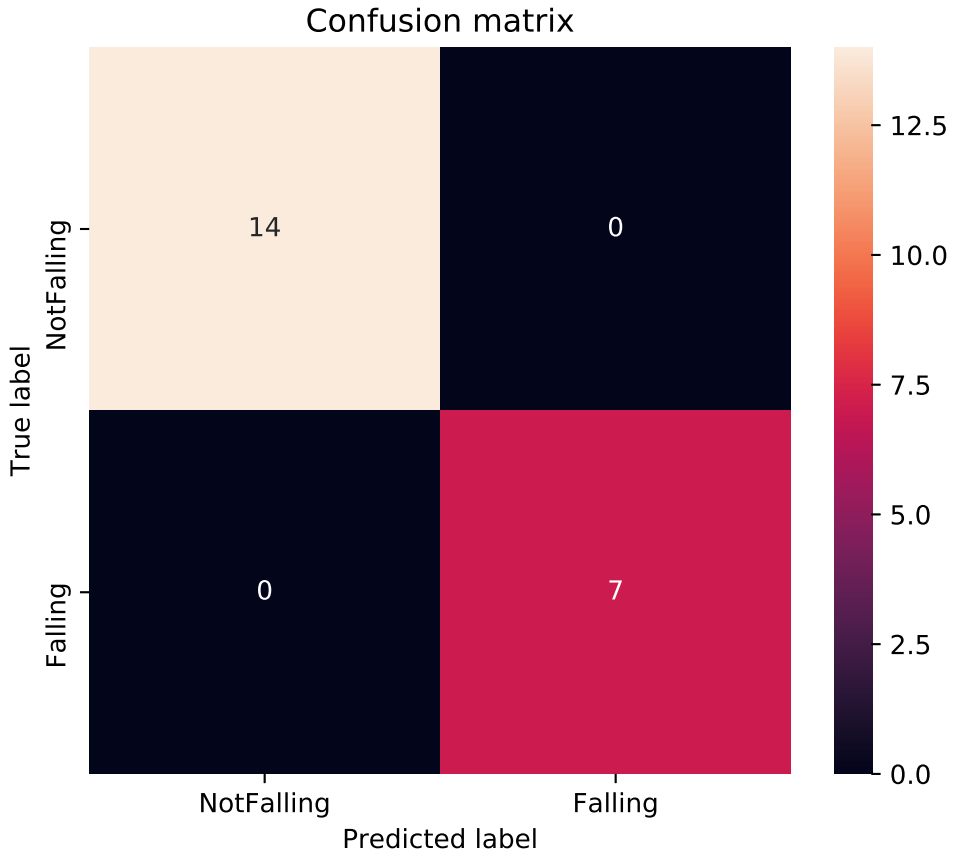


Figure 3.6: The confusion matrix, after training the model.

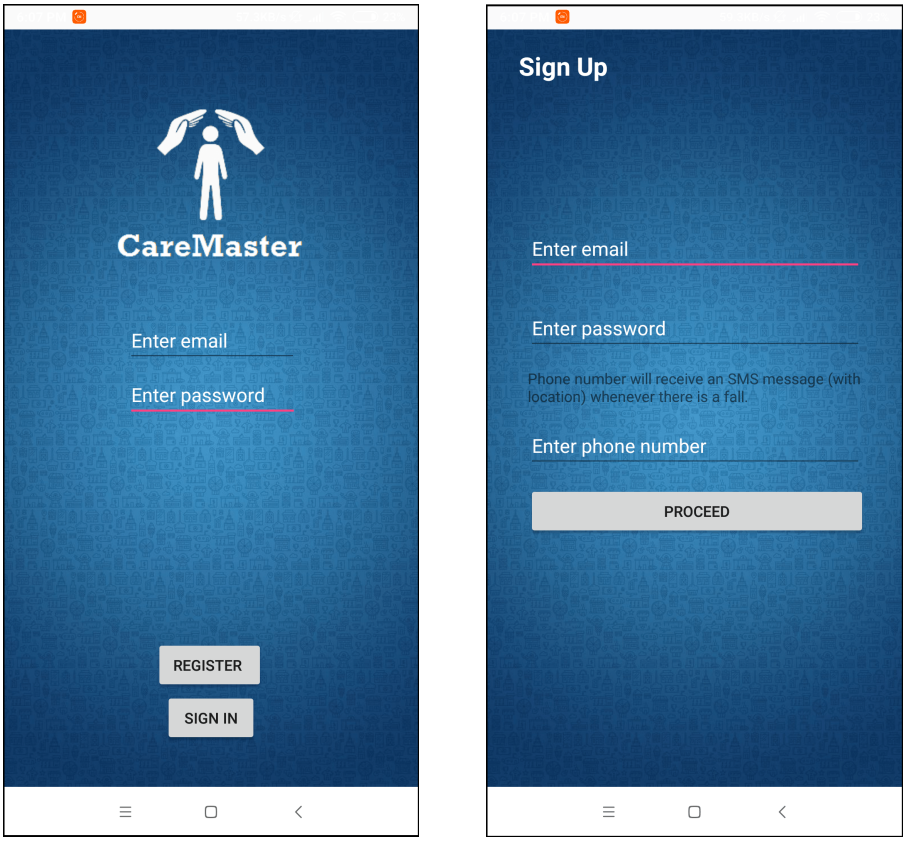
3.4 Android applications

All the scientific work developed until now must be implemented in a way that all the explored algorithms are made use of. The demonstration of these algorithms will be achieved by their implementation on a pair of Android applications: one for the smartphone and another for the smartwatch.

3.4.1 Smartphone app

The smartphone is the main processor of the system. It mainly receives data and transmits it, via the cellular network (or WiFi, when it is available), to a person or institution of interest. Not only that, its also the way the user is registered and authenticated on the server.

The proposed App has a major request of being intuitive and simple to use. It starts with a login screen where a registry is performed in order to ensure privacy issues. The interface aims to be extremely simple, as to the nature of the target public, whom due to age and other possible conditions, may be less familiar with smartphone based technologies and services.



(a) Login screen (b) Signup screen

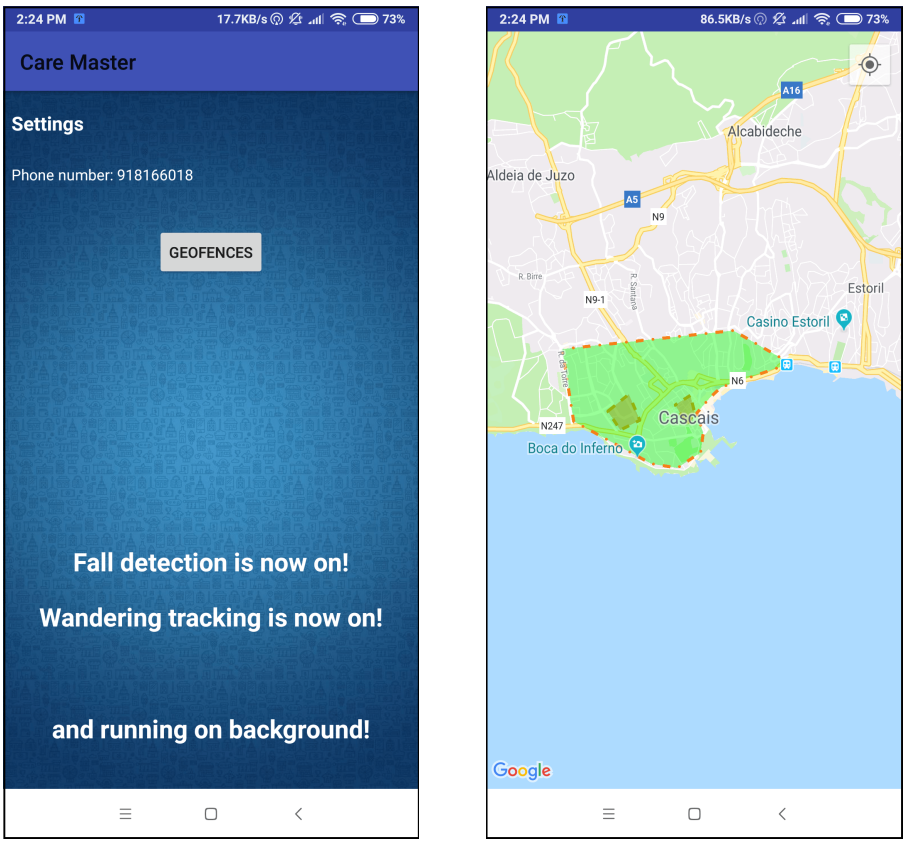
Figure 3.7: Login and signup screens.

The registration process consists of:

- An email address;
- The password;
- The mobile phone of the caregiver.

The rest of the APP is rather simple, but extremelly functional. Right after the login process, and if it is successful, it displays to the user the phone number of the person that is taking care of the pacient. It also warns the user that fall and wandering detection are activated. This is not only important from a practicality point of view, but also because of the privacy aspect, which was discussed in detail in the State of the Art section.

This screen also contains a button called "Geofences"that leads the user to a map that displays the currently allowed zones (green zones) and not allowed zones (everything else, red or not).



(a) After loging in screen

(b) Geofences

Figure 3.8: The screen after signing in and geofences display.

The smartphone aplication uses a service that processes some things:

- Wandering detection;

- Smartwatch fall and wandering confirmations;
- Sending the alerts to the caregiver.

Wandering detection is done via a method called `wanderingDetection`. This method runs on a seperate task, in accordance to a certain frequency. This frequency is determined by another system (it mainly depends on the type of patient and its' risks). However, the determination of that frequency is not part of this dissertation. For testing purposes, pre-determined frequencies, residing on the database, will be used.

The code bellow shows the retrieval of the non-permite zones from the database. These zones are defined by coordinates, that represent polygons. If its detected that the user's current location resides in one of these polygons, a variable, `containsone_RED[o]` is incremented. The code bellow shows the retrieval of those polygons, as well as the incrementation of that variable.

```
firebaseAuth = FirebaseAuth.getInstance();
final int[] p = {0};
final int[] r = {1};
String email_NON_permited = FirebaseAuth.getInstance().getCurrentUser().
    getEmail();
DocumentReference docRef_NON_permited = db.collection("non_permited_areas").
    document(email_NON_permited);
final int[] containsone_RED = {0};
docRef_NON_permited.get().addOnCompleteListener(new OnCompleteListener<
    DocumentSnapshot>() {
  @Override
  public void onComplete(@NonNull Task<DocumentSnapshot> task) {
    List<GeoPoint> pts;
    if (task.isSuccessful()) {
      DocumentSnapshot document = task.getResult();
      while (true) {
        if (document.exists()) {
          String i_poly = Integer.toString(r[0]);
          pts = (List<GeoPoint>) document.get(i_poly);
          List<LatLng> latlng_pts = new ArrayList<>();
          Log.e(TAG, "Coordinate SENSOR vector is : " + pts);
          // If there is a polygon to draw...
          if (pts != null) {
            // While there are points from the polygon to draw...
            while (true) {
              if (p[0] < pts.size()) {
                latlng_pts.add(new LatLng(pts.get(p[0]).getLatitude(), pts.
    get(p[0]).getLongitude()));
                p[0]++;
              }
              else
                break;
            }
```

```
30              boolean contains_RED = PolyUtil.containsLocation(latitude,
    longitude, latlng_pts, true);
31              if(contains_RED)
32                containsone_RED[o]++;
33            } else
34              break;
35            r[o]++;
36          } else {
37            Log.e(TAG, "No such document");
38          }
39        }
40      } else {
41        Log.e(TAG, "get failed with ", task.getException());
42      }
43    }
44 });
```

After the detection, if the value of the variable proves that the user is, in fact, in a non-permited zone, we must send the proper notification to the Smartwatch, as well as send all the information necessary (coordinates, battery info and date/time) to the database.

```
1  if(containsone_RED[o] > o) {
2    Log.e(TAG, "Contains = RED");
3    sendNotification_wandering();
4    Log.e(TAG, "Counter started!");
5    // Send information to the database that user is outside allowed zone to
       platform
6    Map<String, Object> allerts = new HashMap<>();
7    allerts.put("timestamp", FieldValue.serverTimestamp());
8    allerts.put("latitude", latitude);
9    allerts.put("longitude", longitude);
10   allerts.put("battery", batLevel);
11   db.collection("wandering_allerts").document(email)
12       .set(allerts)
13       .addOnSuccessListener(new OnSuccessListener<Void>() {
14         @Override
15         public void onSuccess(Void aVoid) {
16           Log.d(TAG, "DocumentSnapshot successfully written!");
17         }
18       })
19       .addOnFailureListener(new OnFailureListener() {
20         @Override
21         public void onFailure(@NonNull Exception e) {
22           Log.w(TAG, "Error writing document", e);
23         }
24       });
25 }
26 else {
27   Log.e(TAG, "Contains = GREEN");
28 }
```

3.4.2 Smartwatch app

The smartwatch app has three functions:

- Display notifications to the user, naimly the warnings for the detected falls and wandering events;
- Allow the changing of the time delay between the detection and the SMS send to the family person or healthcare institution;
- The implementation of the service that detects the falls, with the machine learning model discussed in the previous section.

As most smartwatches have a round shape, the menu has a design that allows main features to be displayed and easily accessed.

The interaction with the user is made at the screen of a smartwatch and must be kept as simpler as possible according to the target users specificities; people with some age and maybe lack of technological literacy or less dexterity.

Considering such constraints, the Smartwatch APP is simplified to maximize functionality and efficiency to the target users. It alerts in case of detected user's fall or the user wandering askingfor the user interaction to dismiss the emergency request. This procedure aims to dismiss false positives when the person is conscious and able to actively respond. In the case of no response, it acts like a confirmation of the detected risk and subsequent emergency procedures are activated for his/her location and rescue.

First, there's a need to indicate an exact ammount of time desired to send the allert to the person or institution of interest, after que allert is received on the smartwatch. That amount of time is defined on the main screen of the smartwatches' APP. It's possible to choose between 0 and 30 seconds, with a 15 second period being the default value. The screen displayed in Figure 3.9.

Figure 3.9: The main screen, to select the time.

The screen displayed to the user while a fall is detected can be found below in Figure 3.10.

Figure 3.10: The smartwatch notification for a fall.

Next in Figure 3.11, by using the geofence strategy, it is presented the screen for the detected possibility of wandering outside the proposed boundary.

Figure 3.11: The smartwatch notification for wandering.

The person that is confused, lost or with any cognitive limitation will most likely ignore the smartwatch screen but case she/he feels in danger, it is possible to tap the screen to request immediate help by contacting the person that was associated with the system as primary help contact. If the person does not return to the safe boundary within an established time (e.g. 10 minutes case the nearby boundaries do not present significant risks) the person registered in the system will receive a warning message, signaling the distress with the GPS coordinates of the patient in danger.

The service that resides on the smartwatch has two main functions:

- Detect the falls;

- Create notifications, either for falls or for wandering events;
- Send confirmations to the smartphone that a fall or wandering event might actually have happened.

Fall detection is processed by retrieving 100 samples of the accelerometer, with the "FASTEST"sampling rate on Android (which aims at a 0 milisecond delay).

Every time the accelerometer data is changed, the method `onSensorChanged` is called, which updates the accelerometer values and allocates them in an array.

```
@Override
public void onSensorChanged(SensorEvent event) {
  activityPrediction();
        updateAccelParameters(event.values[0], event.values[1], event.values
    [2]);
}
```

When that array reaches the 100 value mark, it is processed by calculating the 3 features we want (PEAK, DT and DP).

```
for (i = 0; i < 101; i++) {

  if (listSums[i] > PEAK_THRESHOLD)
          DP_num = DP_num + 1;

        if (listSums[i] > LAMFAM_THRESHOLD && listSums[i] <= PEAK_THRESHOLD
    && FAM_CHECK == 0) {
                FAM_CHECK = 1;
                FAM_t = i;
        }

        if (listSums[i] > LAMFAM_THRESHOLD && listSums[i] <= PEAK_THRESHOLD
    && FAM_CHECK == 1) {
                LAM_t = i;
                DT_num = LAM_t - FAM_t;
        }

}

PEAK.add(getMax(listSums));
DT.add(DT_num);
DP.add(DP_num);
```

Finally, with these 3 features, that make up the inputs of the classifier, it is classified by the machine learning model, which in Android consists of a class with methods to load the model file and use it to classify the example.

```
public class TensorFlowClassifier {
    static {
        System.loadLibrary("tensorflow_inference");
```

```
4       }
5
6       private TensorFlowInferenceInterface inferenceInterface;
7       private static final String MODEL_FILE = "file:///android_asset/
        frozen_har.pb";
8       private static final String INPUT_NODE = "x";
9       private static final String[] OUTPUT_NODES = {"Out/Softmax"};
10      private static final String OUTPUT_NODE = "Out/Softmax";
11      private static final long[] INPUT_SIZE = {1, 3};
12      private static final int OUTPUT_SIZE = 2;
13
14      public TensorFlowClassifier(final Context context) {
15          inferenceInterface = new TensorFlowInferenceInterface(context.
        getAssets(), MODEL_FILE);
16      }
17
18      public float[] predictProbabilities(float[] data) {
19          float[] result = new float[OUTPUT_SIZE];
20          inferenceInterface.feed(INPUT_NODE, data, INPUT_SIZE);
21          inferenceInterface.run(OUTPUT_NODES);
22          inferenceInterface.fetch(OUTPUT_NODE, result);
23
24          return result;
25      }
26  }
```

3.4.3 App interaction

One way we can implement the interaction between the two devices is through the *Wearable Data Layer API*. This API is part of the Google Play Services and consists of a set of data objects that the system can send and synchronize, along with listeners that notify applications of certain events. The ones we will use are:

- `DataItem` - A DataItem provides data storage with automatic syncing between the handheld and wearable. It generally consists of the following components:

  Payload — A byte array, which you can set with whatever data you wish, allowing you to do your own object serialization and deserialization. The size of the payload is limited to 100KB;

  Path — A unique string that must start with a forward slash.

- `WearableListenerService` - Lets you listen for important data layer events in a service. The system manages the lifecycle of the `WearableListenerService`, binding to the service when it needs to send data items or messages and unbinding the service when no work is needed.

Figure 3.12 shows a sample network that incorporates a connection between a mobile phone and two smartwatches.

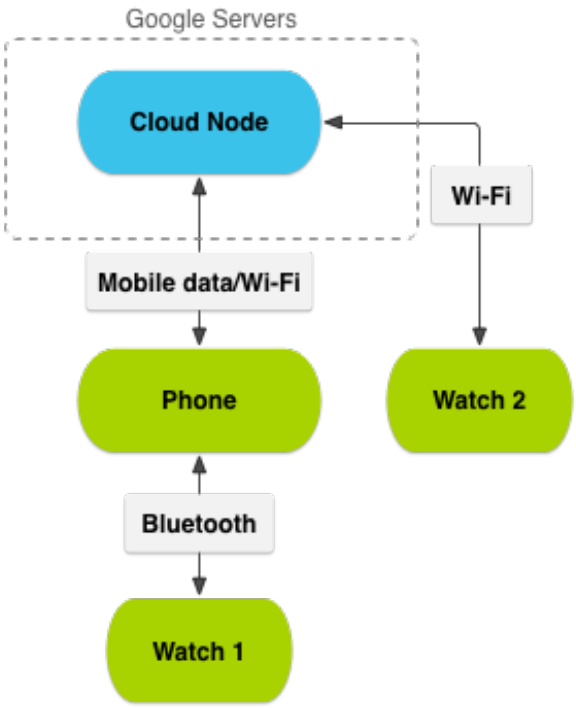


Figure 3.12: A sample network of nodes with handheld and wearable devices.

The first step is to establish a connection between the smartphone and the smartwatch, as shown bellow, using the *GoogleApiClient* in conjunction with the *WearableListenerService*:

```
apiClient = new GoogleApiClient.Builder(this)
                .addApi(Wearable.API)
                .addConnectionCallbacks(this)
                .addOnConnectionFailedListener(this)
                .build();

apiClient.connect();
```

From this point on, its now possible to communicate between the two devices (the smartphone and the smartwatch), by sending data between them. One portion of data we need to send via this connection is the wandering notification, from the smartphone to the smartwatch.

When the smartphones' GPS detects that the patient has gone out of the permited area, it sends a notification to the smartwatch. The data for this notification is sent via the `sendNotification_wandering` method, shown bellow:

```
private void sendNotification_wandering() {
  if (apiClient.isConnected()) {
```

```
3       Log.e(TAG, "Sending data.");
4       PutDataMapRequest dataMapRequest = PutDataMapRequest.create(
        NOTIFICATION_PATH);
5       dataMapRequest.getDataMap().putDouble(NOTIFICATION_TIMESTAMP, System.
        currentTimeMillis());
6       dataMapRequest.getDataMap().putString(NOTIFICATION_TITLE, "WANDERING
        DETECTED!");
7       dataMapRequest.getDataMap().putString(NOTIFICATION_CONTENT, "Tap to
        inform. You are OK.");
8       PutDataRequest putDataRequest = dataMapRequest.asPutDataRequest().
        setUrgent();
9       Wearable.DataApi.putDataItem(apiClient, putDataRequest);
10    } else {
11      Log.e(TAG, "No connection to wearable available!");
12    }
13  }
```

As the user also has the possibility to dismiss the notification and not send the allert to the caregiver, the smartwatch also needs to have the functionality to send data to the smartphone.

After the time to dismiss the allert, defined in the smartwatches' App, is finished, an allert is sent to the smartphone as to confirm that there really has been a wandering or a fall event.

```
1  public class MyCountDownTimer extends CountDownTimer {
2
3    public MyCountDownTimer(long millisInFuture, long countDownInterval) {
4      super(millisInFuture, countDownInterval);
5    }
6
7    @Override
8    public void onTick(long millisUntilFinished) {
9      Log.e(TAG, millisUntilFinished / 1000l + " seconds remaining : ");
10   }
11
12   @Override
13   public void onFinish() {
14     // SEND DATA TO SMARTPHONE, TO SEND SMS!!
15     if (apiClient.isConnected()) {
16       Log.e(TAG, "Sending data ServiceListener.");
17       PutDataMapRequest dataMapRequest = PutDataMapRequest.create(
       NOTIFICATION_PATH_SMS);
18       dataMapRequest.getDataMap().putDouble(NOTIFICATION_TIMESTAMP, System.
       currentTimeMillis());
19       if(title_noti.equals("FALL DETECTED!")) {
20         dataMapRequest.getDataMap().putString(NOTIFICATION_TITLE, "fall");
21         Log.e(TAG, "fall");
22       }
23       if(title_noti.equals("WANDERING DETECTED!")) {
24         dataMapRequest.getDataMap().putString(NOTIFICATION_TITLE, "wander");
```

```
25            Log.e(TAG, "wander");
26          }
27          dataMapRequest.getDataMap().putString(NOTIFICATION_CONTENT, "Tap to
        inform. You are OK.");
28          PutDataRequest putDataRequest = dataMapRequest.asPutDataRequest().
        setUrgent();
29          Wearable.DataApi.putDataItem(apiClient, putDataRequest);
30        } else {
31          Log.e(TAG, "No connection to phone available!");
32        }
33        myCountDownTimer.cancel();
34
35      }
36
37  }
```

3.5 Database

The database will be built on *Firebase*. This tool allow to easily create and implement databases on Android applications as well as web pages.

Firebase has two types of databases: the Real Time Databases and the Cloud Firestore Database. The most interisting one for this project is the last. Cloud Firestore is a NoSQL, document-oriented database. However, it differs from normal SQL databases. There are no tables and rows. Instead, all data is organizes in documents, which then are organizes in collections. Cloud Firestore is optimmized for large collections of small documents and all documents must be stored in collections.

The database for this project will contain various collections:

- User authentication data - properly encripted;
- Relative or care institution phone number;
- Fall allerts - contain the latest location and time of location of the latest fall allert;
- Wandering allerts - contain the latest location and time of location of the latest wandering allert;
- Permited areas - implemented by arrays of geogrphical coordinates;
- Non-permited areas - the same way as permited areas, implemented by arrays of geogrpahical coordinates.

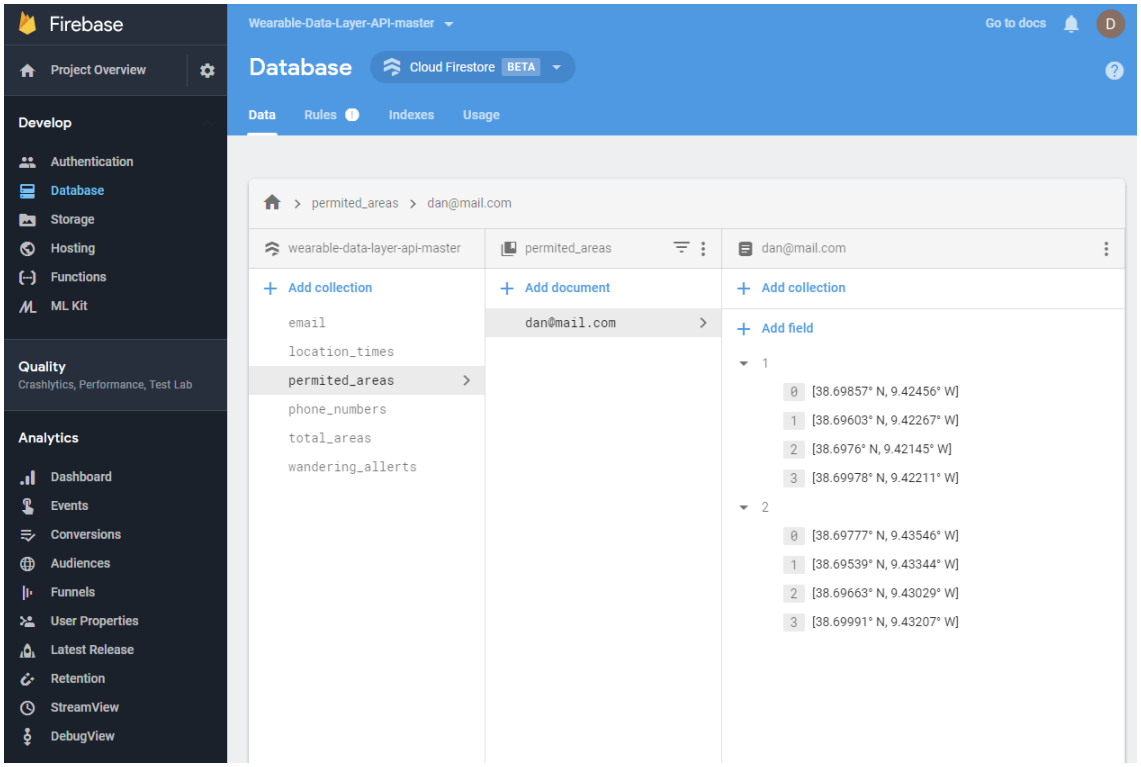


Figure 3.13: The firebase interface.

Figure 3.14 shows how wandering alerts are structured. As we can see, a collection called `wandering_allerts` contains documents, in which each one represents a certain user. There is one wandering alert per user at a maximum. The alerts contain the location, battery info (in percentage) and the time in which the alert was given. Each time there is a new alert, the location, battery info and time are updated, so that the the caregiver can see, immediately the latest info on the patient.

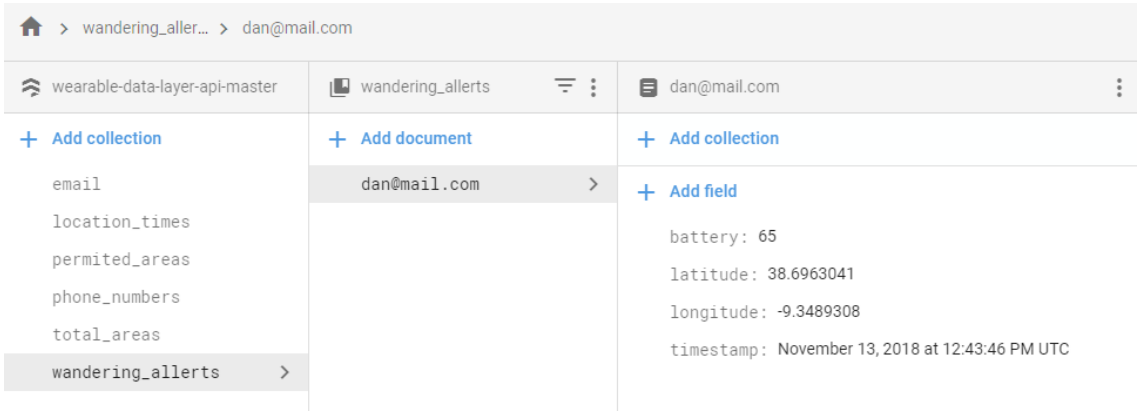


Figure 3.14: The wandering alerts.

One important aspect, as to take into account not only privacy concerns, but also to prevent excessive battery wear and cellular data usage, is to prevent the GPS from retrieving location every certain ammount of time. That time is defined in seconds on the

`location_times` collection, as we can see in Figure 3.15 . Some users may need more regular access than others. For example, a patient that is in a more advanced stage of the desiese is a riskier patient, therefore needs to be checked more regularly.

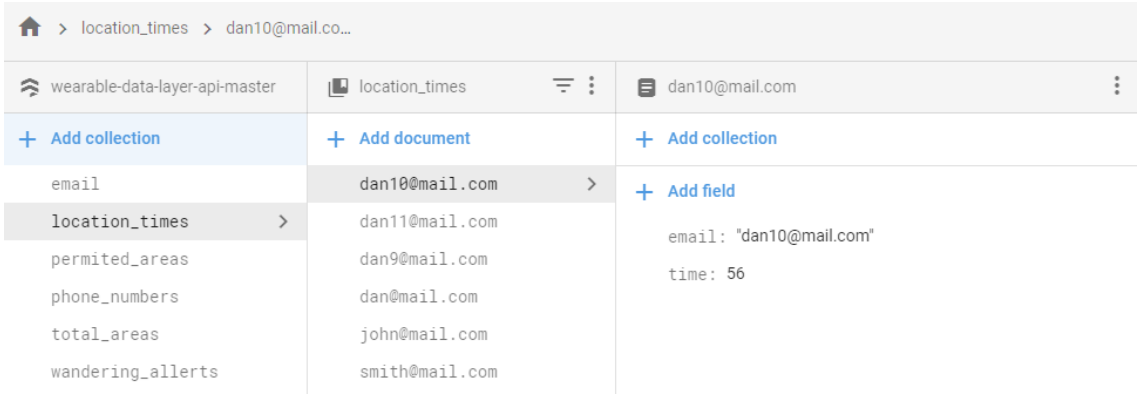


Figure 3.15: The location times.

The first step for defininf locations on the App is to define the general area that is permited for the patient to wander. The `total_area` collection represents just that. Each user has a collection of geographical coordinate points (which the App interprets as a polygon) that define that general area. Figure 3.16 represents it as `fields`.

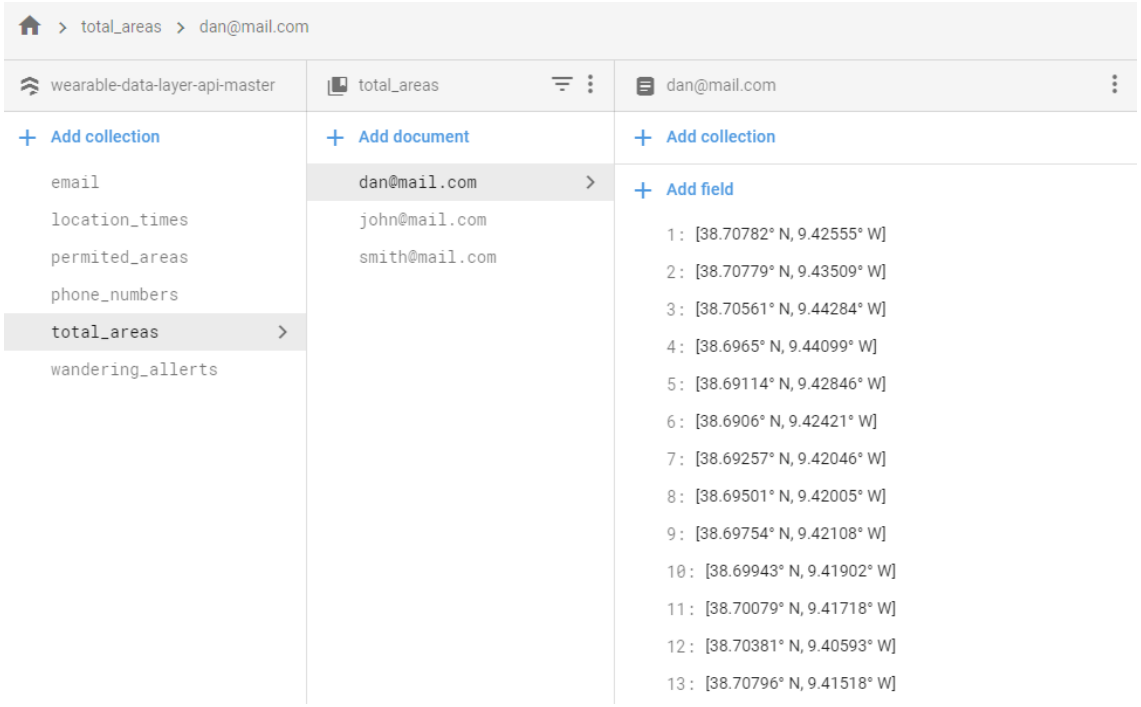


Figure 3.16: The total-area coordinates.

Figure 3.17 shows how non-permited areas are defined on the database. They are comprised of a series of geographical coordinates (with latitude and longitude values), that, overall, define a polygon on the application. As there can be many non permited areas,

each one is defined as an `array` of coordinates on a `field`. The application then defines as many non-permited zones as there are arrays.

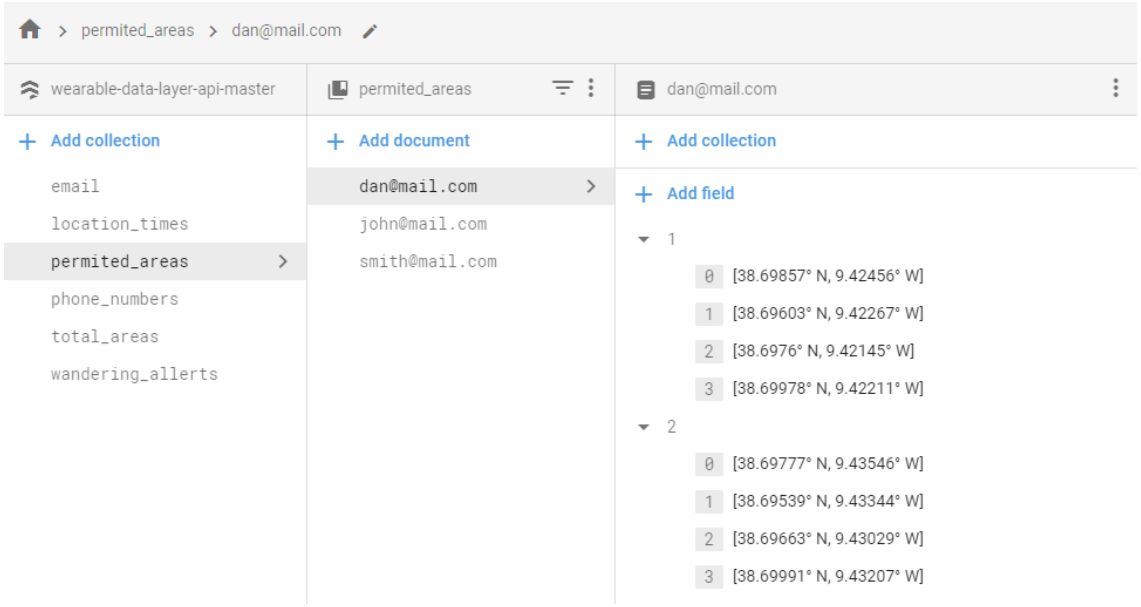


Figure 3.17: The non-permited areas coordinates.

CHAPTER 4

Experiments and validation

4.1 Introduction

In this chapter, the main aspects of the performed experiments on the APP and algorithms are clarified.

Tough the machine learning algorithm was already tested, the real world tests need to be performed, as the results may not be exactly the same.

To study the algorithms and the application performance, real world tests were made, recorded and registered as to compare them to the theoretical results. These tests were performed on various situations, as to try and inspect as many cases as possible.

4.2 Fall detection tests

A set of real world tests, as to test the performance of the fall detection system, were made. These tests analyze the performance of the following:

- The fall detection algorithm performance;
- The patient notification system for fall events;
- The cargeiver notification system for fall events.

The demonstration depicted on Figure 4.1 shows a person falling on an outdoor situation, which should be the more typical one. **About 1 or 2 seconds** after the fall, the watch automatically detects it and displays the notification.

Not less relevant are the other situations. So, other tests were made, whose results are described on Table...

(a) Right before the fall

(b) After the fall

Figure 4.1: The fall detection at work.

4.3 Wandering tests

Some wandering detection tests were made, also in various situations and locations. These tests analyze the performance of the following aspects and systems:

- The wandering detection algorithm performance;
- The patient notification system for wandering events;
- The caregiver notification system for wandering events.

Figure 4.2 shows a person walking arround a typical path, then starting to deviate from the path and then going outside the allowed area, which is interpreted by the algorithm as a wander.

(a) Walking arround

(b) Deviating from allowed area

(c) Outside allowed area

Figure 4.2: The wandering detection at work.

As with the fall detection test, right after the user went outside the allowed area, a notification was sent to the smartwatch and, consequently, to the phone number (in a real situation, the caregiver) defined in the application.

4.4 Validation

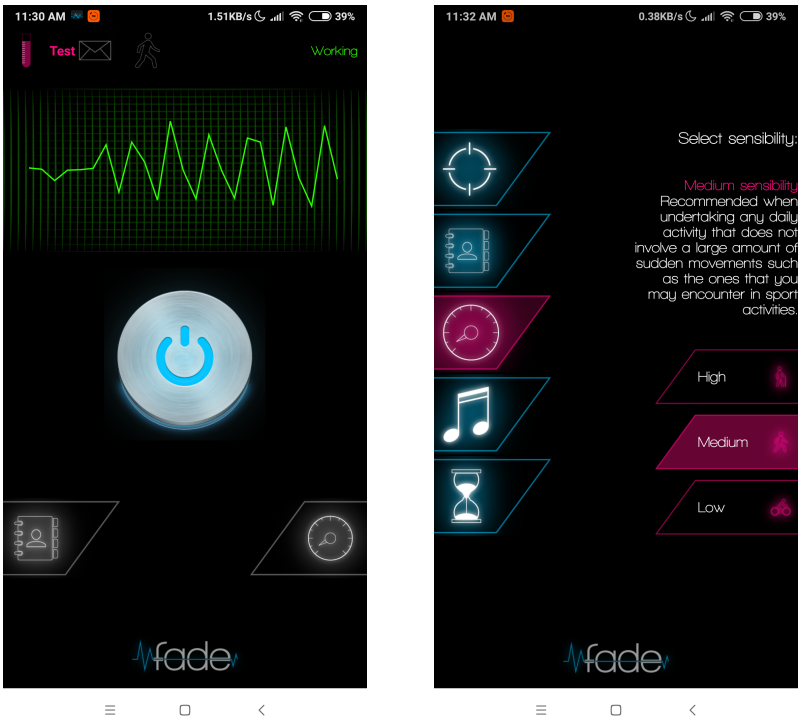
In this section some software competitors to the system developed on this thesis will be tested and compared with it.

4.4.1 *Fade*

Fade is a mobile application designed and developed by *ITER, S.A* able to detect falls using the information collected by some smartphone sensors, mainly the accelerometer. It has been designed for people that might be under risks of falling or in an isolated environment without supervision.

Fade can be configured to send an SMS message and/or email to a contact of choice in emergency situations. It reports the time and place where the incident occurred, using the device's geolocation tools. It's also possible to select the sensitivity of the fall detection.

All these things considered make *Fade* a direct competitor to the App being developed for this thesis.



(a) The screen, after turning *Fade* on.

(b) The settings menu.

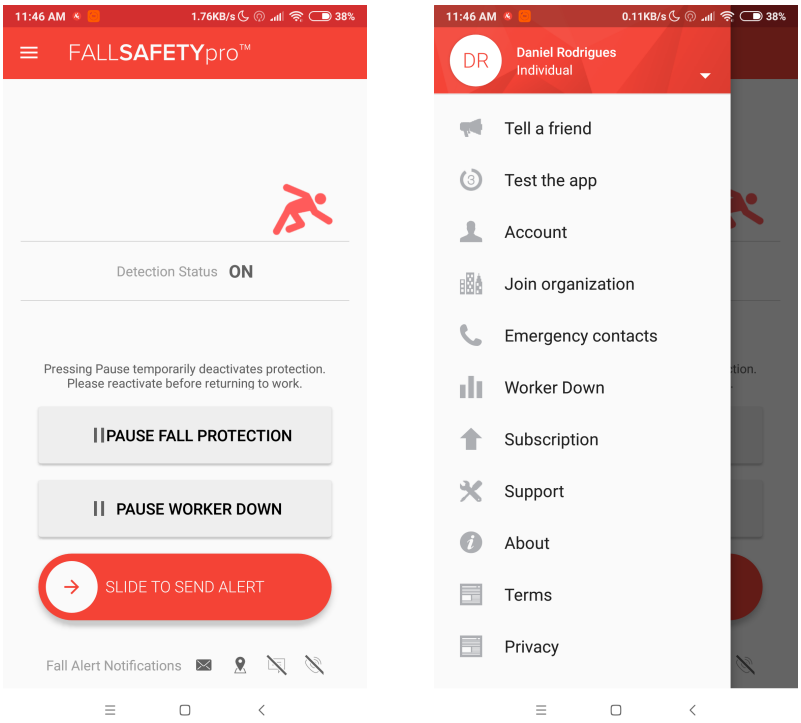
Figure 4.3: Some *Fade* screenshots.

4.4.2 *FallSafetyPro*

FallSafetyPro is an app mainly designed for roofers, lineman, oil field workers, framers, window cleaners and other skilled professionals whose jobs put them at risk of falls.

It leverages "powerful technology"in a smartphone to detect sudden movements that indicate a fall. It has a 45 second timer that gives you that time for the person to say that he or she is ok. A second alarm helps rescuers find the exact location of the person.

Once a fall is detected, FallSafetyPro alerts the emergency contact (or contacts) by text, voice and email, providing GPS location.



(a) The screen, after turning *FallSafetyPro* on.

(b) The settings menu.

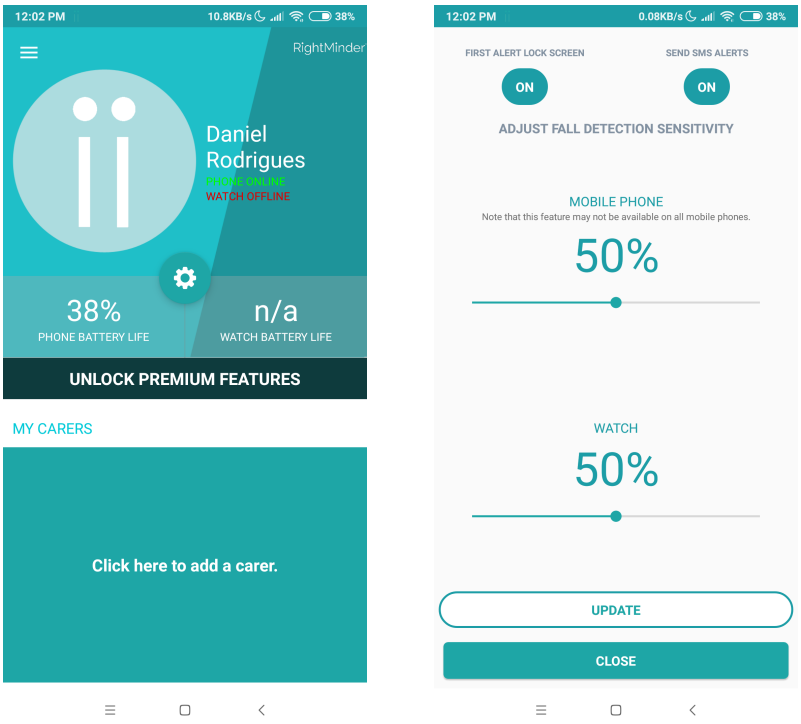
Figure 4.4: Some *FallSafetyPro* screenshots.

4.4.3 *RightMinder Plus*

RightMinder Plus is entitled as the "Worlds First Fall detection App & First Alerts for SmartWacthes plus phones". It can be used, not only using a smartphone, but also many smartwatches.

The "Premium Services"(at a cost) allow the unlock of certain features like GPS Locations, Battery Life Reminders and Fast Call.

RightMinder only works when securely connected to another person (one or many) registered as a user of *RightMinder*. One person must be a "Wearer"and another (or many others) a "Carer".



(a) The screen, after turning *Rightminder* on.

(b) The settings menu.

Figure 4.5: Some *Rightminder* screenshots.

4.4.4 Tests and results

Various tests were made for each mobile application. Each test consisted of a different type of movement or situation that could lead to a real fall detection (True Positive - TP) or a misinterpretation by the mobile app (False Postive - FP).

Starting with *Fade*, it is observable that jumping yielded bad results, as its a very dificult test. The falling of a person on the floor didn't give a very good True Positive rate either. One interesting aspect of this Apps algorithm is that it can detect constant, repetitive movements and ignore them, thus the impressive jogging results. Final result was decent, but the fall detection was poor.

FallSafetyPro gave, on the whole, the poorer of the results. Fall detection was very bad, as it couldn't detect either of the simple fall tests. However, it gave excellent results on all of the other False Positive detection tests. This probaby means that the algorithm on the app isn't sensitive enough and needs to be updated.

Very good and relatively consistent results were given by the *RightMinder* app. Jogging and jumping, beeing the two most dificult tests, did expose problems. However, everyday activities detection was very good and fall detection was on par with the best.

This leaves us with the app that yielded the best results of all: *CareMaster*, this thesis app. Even tough, on average, the results weren't much better, it improved slightly on daily activities, compared with *RightMinder*, while maintaining the good fall detection.

Table 4.1: The results of the four mobile applications fall detection systems.

Test type 1	**Test type 2**	***Fade***	***Fall Safety***	***Right Minder***	***Care Master***
Sitting	Chair	75%	100%	100%	100%
	Bed				
	Couch				
	Floor				
Standing	Chair	100%	100%	100%	100%
	Bed				
	Couch				
	Floor				
Jogging	Slow	100%	100%	50%	50%
	Fast				
Jumping	Low	0%	100%	50%	100%
	High				
Falling (x4)	Bed (x4)	63%	0%	75%	75%
	Floor (x4)				
AVERAGE RESULTS		**70%**	**60%**	**80%**	**85%**

CHAPTER 5

Conclusions and Future Work

5.1 Conclusions

In the first part of this dissertation the main research motivations that contributed to the work developed were presented. Some existing solutions to the problem at hands were mentioned, in conjunction with their benefits and issues. To solve these issues, a few technical solutions were presented, mainly algorithms that could help create a better solution to the problem.

Next, some of these algorithms were implemented. For the fall detection, an algorithm based on an Artificial Neural Network (ANN) was used, as it's the most commonly used machine learning algorithm for datasets with a relatively low number of examples. The algorithm was implemented using Tensorflow, an increasingly important platform for machine learning models development. The theoretical results, after the training of the model, were presented.

Two applications were developed: a smartphone Android APP and a smartwatch Android Wear APP. The former gave the ability to create an account for the patient and view the geofences (areas where the patient could and couldn't go). The latter was used to, through the smartwatch's accelerometer, detect the falls and, also, to give the ability to the patient to confirm false positives and consequently dismiss notifications to the caregiver, solving some privacy concerns.

Unifying the two applications was an online database, developed using Firebase. Fall and wandering events as well as all the accounts data was stored. The two devices (and respective App's) and database, together formed a system for fall and wandering detection.

The final solution mainly achieved what was looked after: a system that monitored

Alzheimer patients, detecting accidental falls and wandering, making sure that any caregivers were given every bit of information that was relevant about the patient.

The concept of this solution was, from the start, proven to be extremely useful nowadays, with an increasing number of patients with this desiese. There is, therefore somewhat of a high demand for these types of products.

Through all the testing done, it was concluded that, on the whole, the system worked in a way which proves that the achieved solution goes to meet the hypothesis of this dissertation. The limitations mainly had to do with optimization of the machine learning algorithm, as it's something that needs a lot of time and, mainly, a lot of data to train the models.

To conclude, this work was presented as a part of the European Union program *Ambient Assisted Living CARELINK*.

5.2 Future work

As almost all types of relevant and big projects, this one will, of course, be open for improvements. Some of these are described bellow:

- Improve the machine learning model, by creating a bigger and more robust dataset;
- Using the current Android applications as a basis, do some improvements on the UI level;
- Create a Web interface for the caregiver to inspect the patient's status and create/alter the geofences;
- Try and improve the battery performance, using the system, mainly by using more efficient location algorithms.

Bibliography

[1] *6 Easy Steps to Learn Naive Bayes Algorithm (with code in Python).* URL: https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/ (visited on 08/11/2018).

[2] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, and G. Brain. "TensorFlow: A System for Large-Scale Machine Learning TensorFlow: A system for large-scale machine learning." In: *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)* (2016), pp. 265–284. ISSN: 0270-6474. DOI: 10.1038/nn.3331. arXiv: 1605.08695. URL: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi.

[3] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems." In: (2016). ISSN: 0270-6474. DOI: 10.1038/nn.3331. arXiv: 1603.04467. URL: http://arxiv.org/abs/1603.04467.

[4] D. R.d.O. M. Abreu, R. C.d. S. Azevedo, A. M. C. da Silva, A. A. O. Reiners, and H. C. A. Abreu. "Fatores associados à recorrência de quedas em uma coorte de idosos." In: *Ciência & Saúde Coletiva* 21.11 (2016), pp. 3439–3446. ISSN: 1413-8123. DOI: 10.1590/1413-812320152111.21512015. URL: http://www.scielo.br/scielo.php?script=sci{_}arttext{\&}pid=S1413-81232016001103439{\&}lng=pt{\&}tlng=pt.

[5] Z. Alam, H. Samin, and O. B. Samin. "HealthBand for Dementia Patients: Fall and Scream Detector and Caretaker Helper." In: *Journal of Physics: Conference Series* 976.1 (2018), pp. 0–6. ISSN: 17426596. DOI: 10.1088/1742-6596/976/1/012015.

[6] A. R. C. de Almeida. "Environment-Aware System for Alzheimer ' s Patients." In: (2014).

[7] G. Brown. "An accelerometer based fall detector: development, experimentation, and analysis." In: *University of California, Berkeley* July (2005), pp. 1–9. DOI: 10.1.1.128.6988. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.128.6988{\&}rep=rep1{\&}type=pdf.

[8] *Checkpoints | TensorFlow*. URL: https://www.tensorflow.org/guide/checkpoints (visited on 08/11/2018).

[9] *Childcare, Babysitters, Childminders, Nannies & Senior Care*. URL: https://www.mindme.ie/ (visited on 08/01/2018).

[10] P. A. Defina, R. S. Moser, M. Glenn, J. D. Lichtenstein, and J. Fellus. "Alzheimer's disease clinical and research update for health care practitioners." In: *Journal of Aging Research* 2013.Mci (2013). ISSN: 20902204. DOI: 10.1155/2013/207178.

[11] F. Ferretti, D. Lunardi, and L. Bruschi. "Causas e consequências de quedas de idosos em domicílio." In: *Fisioterapia em Movimento* 26.4 (2013), pp. 753–762. ISSN: 0103-5150. DOI: 10.1590/S0103-51502013000400005. URL: http://www.scielo.br/scielo.php?script=sci{_}arttext{\&}pid=S0103-51502013000400005{\&}lng=pt{\&}tlng=pt.

[12] A. Geissbühler, J. Demongeot, M. Mokhtari, B. Abdulrazak, and H. Aloulou. "Inclusive smart cities and e-health: 13th international conference on smart homes and health telematics, ICOST 2015 Geneva, Switzerland, june 10-12, 2015 proceedings." In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9102.June (2015). ISSN: 16113349. DOI: 10.1007/978-3-319-19312-0.

[13] *GPS SmartSole® GPS SmartSole wearable tracking solution for wandering*. URL: http://gpssmartsole.com/gpssmartsole/ (visited on 08/01/2018).

[14] *GPS tracker device for kids with special needs. Order Now!* URL: https://www.angelsense.com/ (visited on 08/11/2018).

[15] W. Hedgecock, M. Maroti, J. Sallai, P. Volgyesi, and A. Ledeczi. "High-accuracy differential tracking of low-cost GPS receivers." In: *Proceeding of the 11th annual international conference on Mobile systems, applications, and services - MobiSys '13* (2013), p. 221. DOI: 10.1145/2462456.2464456. URL: http://dl.acm.org/citation.cfm?doid=2462456.2464456.

[16] *How to Prevent Wandering in Alzheimer's Patients | BrightFocus Foundation*. URL: https://www.brightfocus.org/alzheimers/article/how-prevent-wandering-alzheimers-patients (visited on 08/11/2018).

[17] *iTraq Global Location Tracking Device to find your loved ones – iTraq, Inc.* URL: https://www.itraq.com/ (visited on 08/01/2018).

[18] R. Jurdak, P. Corke, D. Dharman, and G. Salagnac. "Adaptive GPS duty cycling and radio ranging for energy-efficient localization." In: *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems - SenSys '10* SenSys (2010), p. 57. ISSN: 15504859. DOI: 10.1145/1869983.1869990. URL: http://portal.acm.org/citation.cfm?doid=1869983.1869990.

[19] E. M. Kato-Narita and M. Radanovic. "Characteristics of falls in mild and moderate Alzheimer's disease." In: *Dement Neuropsychol* 3.4 (2009), pp. 337–343. ISSN: 1980-5764. DOI: 10.1590/S1980-57642009DN30400013. URL: http://www.scielo.br/pdf/dn/v3n4/1980-5764-dn-3-04-00337.pdf.

[20] A. Küpper, U. Bareth, and B. Freese. "Geofencing and background tracking–The next features in LBSs." In: *Proceedings of the 41th Annual Conference of the Gesellschaft für Informatik eV* (2011).

[21] A. Kupper. "Location-based Services Location-based Services Fundamentals and Operation Axel K upper." In: *Location-based Services:Fundamentals and Operation* 5 (2005). URL: http://onlinelibrary.wiley.com/doi/10.1002/0470092335.fmatter/pdf{\%}5Ct{\%}5Cn.

[22] Y. T. Kwak, Y. Yang, and M.-S. Koo. "Wandering in Dementia." In: *Dementia and Neurocognitive Disorders* 14.3 (2015), p. 99. ISSN: 1738-1495. DOI: 10.12779/dnd.2015.14.3.99. URL: http://synapse.koreamed.org/DOIx.php?id=10.12779/dnd.2015.14.3.99.

[23] B. Kwolek and M. Kepski. "Human fall detection on embedded platform using depth maps and wireless accelerometer." In: (). URL: http://home.agh.edu.pl/{~}bkw/research/pdf/2014/KwolekKepski{_}CMBP2014.pdf.

[24] F. Luis-Ferreira, D. Rodrigues, D. Pereira, J. Sarraipa, and R. Jardim-Gonçalves. "Assessment of Risk for Alzheimer Patients by Monitoring Heartbeat and Identifying Falls." In: *Proceedings of 10th International Conference on e-Health 2018* (2018).

[25] D. Martino-Saltzman, B. B. Blasch, R. D. Morris, and L. W. McNeal. "Travel behavior of nursing home residents perceived as wanderers and nonwanderers." In: *Gerontologist* 31.5 (1991), pp. 666–672. ISSN: 00169013. DOI: 10.1093/geront/31.5.666.

[26] R. Mcshane, K. Gedling, J. Keene, C. Fairburn, R. Jacoby, and T. Hope. "Getting lost in dementia: A longitudinal study of a behavioral symptom." In: *International Psychogeriatrics* 10.3 (1998), pp. 253–260. ISSN: 10416102. DOI: 10.1017/S1041610298005365. URL: http://www.ncbi.nlm.nih.gov/pubmed/9785146.

[27] M. Monitoring, U. Sensor, and T. H.E. S. Vector. "A machine-learning based approach to pre-impact fall detection with wearable devices." In: (2015).

[28] S. Paiva and C. Abreu. "Low Cost GPS Tracking for the Elderly and Alzheimer Patients." In: *Procedia Technology* 5 (2012), pp. 793–802. ISSN: 22120173. DOI: 10.1016/j.protcy.2012.09.088. URL: http://linkinghub.elsevier.com/retrieve/pii/S2212017312005191.

[29] *PocketFinder 3G GPS Trackers for Children, Pets, Seniors, & Vehicles*. URL: https://pocketfinder.com/ (visited on 08/01/2018).

[30] *Premade Estimators | TensorFlow*. URL: https://www.tensorflow.org/guide/premade{_}estimators (visited on 08/01/2018).

[31] A. Z. Rakhman, L. E. Nugroho, Widyawan, and Kurnianingsih. "Fall detection system using accelerometer and gyroscope based on smartphone." In: *2014 1st International Conference on Information Technology, Computer, and Electrical Engineering: Green Technology and Its Applications for a Better Future, ICITACEE 2014 - Proceedings* July 2015 (2015), pp. 99–104. DOI: 10.1109/ICITACEE.2014.7065722.

[32] M. J. Rycroft. *Understanding GPS. Principles and applications*. Vol. 59. 5. 1997, pp. 598–599. ISBN: 1580538940. DOI: 10.1016/S1364-6826(97)83337-8. arXiv: arXiv:1011.1669v3. URL: http://linkinghub.elsevier.com/retrieve/pii/S1364682697833378.

[33] V. Sharma, S. Rai, and A. Dev. "A Comprehensive Study of Artificial Neural Networks." In: *International Journal of Advanced Research in Computer Science and Software Engineering* 2.10 (2012), pp. 278–284. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.468.9353{\&}rep=rep1{\&}type=pdf.

[34] *TensorFlow-DeveloperCoding*. URL: http://developercoding.com/tensorflow/intro-to-neural-networks.php{\#}what-are-neural-networks (visited on 07/20/2018).

[35] *TensorFlow-DeveloperCoding*. URL: http://developercoding.com/tensorflow/basics.php{\#}what-are-tensors? (visited on 07/20/2018).

[36] *The magic of LSTM neural networks – DataThings – Medium*. URL: https://medium.com/datathings/the-magic-of-lstm-neural-networks-6775e8b540cd (visited on 07/30/2018).

[37] H. A. Tran, Q. T. Ngo, and V. Tong. "A new fall detection system on Android smartphone: Application to a SDN-based IoT system." In: *Proceedings - 2017 9th International Conference on Knowledge and Systems Engineering, KSE 2017* 2017-Janua (2017), pp. 1–6. ISSN: 2164-2508. DOI: 10.1109/KSE.2017.8119425.

[38] *Understanding LSTM Networks – colah's blog*. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (visited on 07/30/2018).

[39] *Understanding Support Vector Machine algorithm from examples (along with code)*. URL: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/ (visited on 08/11/2018).

[40] *Wandering | Alzheimer's Association.* URL: https://www.alz.org/help-support/caregiving/stages-behaviors/wandering (visited on 08/11/2018).

[41] J. Whitney, J. C. T. Close, S. H. D. Jackson, and S. R. Lord. "Understanding risk of falls in people with cognitive impairment living in residential care." In: *Journal of the American Medical Directors Association* 13.6 (2012), pp. 535–40. ISSN: 1538-9375. DOI: 10.1016/j.jamda.2012.03.009. URL: http://www.ncbi.nlm.nih.gov/pubmed/22561138.