Master's Degree in Informatics Engineering

Dissertation/Internship

Final Report

The Happy System

Ricardo Miguel Pires Barbosa
rmpb@student.dei.uc.pt

Supervisors:
Jorge Sá Silva
David Nunes
Date: July 1st 2016

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Abstract

In the past few years, the idea of the "Internet of Things" has been developing rapidly, with sensors and machines communicating with each other through the Internet. These new technologies can be used to support new types of Cyber-Physical systems(CPS). Even though CPS consider humans as a part of itself, they still treat them as external element, with unpredictable behavior. In order to serve human needs better, they have to take into account all sorts of psychological and emotional states.

Smartphones present an excellent opportunity to do so, seen as they contain several sensors that allow us to collect information about user movement, location, their environment, and interaction with other people. This type of mobile device usually accompanies the user anywhere he goes throughout the day.

We have taken this opportunity and tried to create apps that can infer human emotions and help people by giving them suggestions that may improve their mood. These apps are HappySPEAK and WeDoCare. We also tested and analyzed the system on which these were built upon: HappyWalk.

All of these apps are based on the Happy System architecture. HappySPEAK aims to further improve this platform by detecting if the user is isolated and give suggestions that may help him overcome it. Migrants are the perfect candidates to use this, seeing that it is common for them to come alone to a new country in order to provide for their families.

WeDoCare aims to detect violent attacks against users, and aims to help them by alerting other people that an attack is happening, show danger zones and alternate paths to avoid them.

The rest of the report will focus on the Software Engineering part, implementation of these apps and what we can do to further improve them.

We will also present all of the performance and emotion detection test results and analyze them thoroughly to expose our findings.

Everything described in this report reflects the work done in the 2015/2016 school year.

Keywords

"Human-in-the-Loop", "Ubiquitous Computing", "Behavior Inference", "Smartphone", "Cyber Physical Systems", "Behavior Change Intervention", "Attack Classification", "Emotional Sensing"

Index

Chapter 1 1

Introduction 1

1.1-Context 1

1.2-Objectives 2

1.3- Report Structure 2

1.4- Work planning and methodology 2

1.4.1-Work Planning 2

1.4.2- Work Methodology 3

1.5- Technologies used 3

Android 3

Google Play Services 4

UBHave 4

Encog 4

Server 4

Hibernate 5

PostgreSQL 5

Trello 5

Chapter 2 7

State of the Art 7

2.1-Concepts and Paradigms 7

Human-in-The-Loop 7

Behavior Change Intervention 8

Cyber-Physical Systems 8

Recommender Systems 8

Emotional Maps 9

Artificial Neural Networks 9

2.2-Similar Apps 10

StudentLife 10

EmotionSense 11

Mapiness 11

Nike Running 13

Yelp 13

Chapter 3 15

The Happy System 15

3.1-The Happy System 15

3.1.1- The Happy System Architecture 15

3.1.2- HappyHour 16

3.1.3-Happywalk 17
3.2-Software Requirements Specification 20
3.2.1-HappySPEAK 20
Non-Functional Requirements 30
3.2.2- WeDoCare 31
Non-Functional Requirements 34
Chapter 4 35
Development 35
4.1- HappySPEAK 35
4.1.1-Task Assignment 35
4.1.2-Android app 35
4.1.3- WebService 39
4.1.4-HappySPEAK Server 40
4.1.5-HappySPEAK Test Setup 42
4.2- WeDoCare 42
4.2.1-Task Assignment 42
4.2.2-Android app 42
Chapter 5 45
Tests and Results 45
5.1-Performance Tests 45
5.2-Neural Network Accuracy Tests 46
5.3-HappyWalk Test Results 46
5.3.1- HappyWalk Performance Test Results 46
5.3.2-HappyWalk ANN accuracy Tests 50
5.4-HappySPEAK Test Results 52
5.4.1- HappySPEAK Performance Test Results 52
5.4.2- HappySPEAK Accuracy Test Results 53
Chapter 7 59
Conclusions 59

Figure Index

Figure 1-Human-in-the-loop Control Loop[7] 7
Figure 2-Component Interaction in Student Life[5] 11
Figure 3-Happy System high-level architecture 15
Figure 5-HappyHour Heat Maps[6] 16
Figure 4-HappyHour Employee backend[6] 16
Figure 6-HappyWalk Logo[7] 17
Figure 7-HappyWalk HiTL control[7] 17
Figure 8-HappyWalk Neural Network Diagram[7] 17
Figure 9-HappyWalk Feedback Request[7] 18
Figure 10-HappyWalk component interaction[7] 18
Figure 11-HappyWalk behavior[7] 19
Figure 12-HappyWalk emotion classification[7] 19
Figure 13-HappyWalk Database Diagram 19
Figure 14-SPEAK Website 22
Figure 15-HappySPEAK User-Environment Design Model 23
Figure 16-Login Screen Mockup 24
Figure 17-Login Dialog Mockup 24
Figure 18-Registration Dialog Mockup 25
Figure 19- Event List Mockup 25
Figure 20-Information Screen Mockup 26
Figure 21- Home Configuration Screen 26
Figure 22-WeDoCare User Environment Design 33
Figure 23-WeDoCare User perspective 33
Figure 24-HappySPEAK Information Screen 36
Figure 25-HappySPEAK Login Screen 36
Figure 26-Happy SPEAK Login/Register Dialogs 37
Figure 27-HappySPEAK Event List Screen 37
Figure 28-HappySPEAK Event Details Screen 38
Figure 29-HappySPEAK User Feedback Menu 39
Figure 30-- HappySPEAK Database Diagram 41
Figure 31-WeDoCare Login Screen 43
Figure 32-WeDoCare Map Screen 43
Figure 33-WeDoCare attack classification 44
Figure 34-WeDoCare app behavior 44
Figure 35-HappyWalk energy consumption with GPS+Wi-fi 47
Figure 36-HappyWalk Energy Consumption with Wi-Fi 47
Figure 37-HappyWalk battery usage with Alarm Manager 48
Figure 38-CPU usage for ANN #1 49
Figure 39-CPU usage for ANN #2 49
Figure 40-CPU usage for ANN #3 50
Figure 41-HappyWalk accuracy over time 51
Figure 42-HappyWalk accuracy Trend Lines 51
Figure 43-HappyWalk user feedback scatter chart 52
Figure 44-HappySPEAK battery life with Alarm Manager 53
Figure 45-HappySPEAK accuracy test results 54
Figure 46- HappySPEAK user feedback histogram 54
Figure 47-Work progress over the year 57

Table Index

Table 1-HappySPEAK Stakeholders 21
Table 2-HappySPEAK Login Functional Requirements 27
Table 3-HappySPEAK Registration Functional Requirements 28
Table 4-HappySPEAK See Events Functional Requirements 28
Table 5-HappySPEAK See Event Details Functional Requirements 29
Table 6-HappySPEAK Event Subscription Functional Requirements 29
Table 7-HappySPEAK Event Unsubscription Functional Requirements 29
Table 8-HappySPEAK Collect Data Functional Requirements 30
Table 9-HappySPEAK User Feedback Functional Requirements 30
Table 10-WeDoCare Stakeholders 32
Table 11- Information Screen Resources 36
Table 12-Login Screen Resources 36
Table 13-Event Screen Resources 37
Table 14-Custom adapter resources 38
Table 15-Event Details Screen Resources 38
Table 16-HappySPEAK Android Request 39
Table 17-HappySPEAK Webservice methods 40
Table 18-Database Tables Description 42
Table 19-Performance request description 42
Table 20- Tested network configurations 48
Table 21-HappySPEAK average user accuracy 54

Abbreviations

AFA	Associação Fazer Avançar
ANN	Artificial Neural Network
BCI	Behavior Change Intervention
CPS	Cyber-Physical Systems
DNN	Deep Neural Networks
GPA	Grade Point Average
GPS	Global Positioning System
HiTL	Human-in-The-Loop
MVC	Model View Controller
MVCC	Multiversion Concurrency control
OS	Operating System
POI	Point of Interest
UUID	Universally unique identifier

Chapter 1

Introduction

1.1-Context

This report describes the work done for the Internship/Dissertation course of the school year of 2015/2016. This course is part of the Masters in Computer Engineering in the University of Coimbra.

The world faces many social problems such as child labor, sub-par health access and the current migrant crisis. Over the years, technology has been developed to mitigate these problems, providing education for children in underdeveloped countries[26], health management in underprivileged areas[27] and outreach programs to help refugees start new lives[28]. We believe that technology is the path forward to solve many more of these problems, and as such, we explored and developed a set of solutions to help migrants and refugees improve their lives: WeDoCare, and HappySpeak.

These projects are built upon the Happy System, an architecture that aims to use sensors to infer human emotions, and use the knowledge it gains to improve the user's life. Smartphones are the perfect tool for this, as they are becoming more widespread each year and possess an array of sensors that can be used to monitor the user and his environment.

The projects at hand were developed in partnership with a group of investigators of the Communications and Telematics Lab of the University of Coimbra (LCT UC). Even though the apps all had different goals and were directed at different classes of end-users, they were all part of the Human-in-the-Loop (HiTL) paradigm.

We mainly focused on two groups of people: migrants and refugees. In order to understand the needs of migrants, we exchanged knowledge and formed a partnership with AFA and its program SPEAK.

SPEAK is a cultural program designed to bring people together by promoting language learning. Its premise is simple: the main barrier that stops foreign people from bonding is language. SPEAK tries to bring down this barrier by allowing anyone from anywhere to sign up and learn or teach other languages. It goes even further by organizing social events with the purpose of creating a bond between participants. [1]

Migrants mostly move to a new country to find better standards of living (due to financial or social needs), but sometimes can't speak the language and they travel alone. Not knowing the language makes it hard for them to socialize and meet new people. This is where HappySPEAK tries to help.

HappySPEAK is an Android system that replicates SPEAK's platform, allowing users to register, and view future SPEAK events in their area. Also, it collects information about the user, such as messaging and call logs, in an attempt to detect if the user feels lonely. This information is then used to suggest the user places he can visit or events he can attend.

As we have stated before, one the main social issues is the huge influx of refugees due to the war in Syria. Millions of refugees are trying to get to Europe in order to escape war in their country. To help with these efforts many refugee camps have been created, but, due to a lack of resources and manpower, security in these places is becoming scarce. To help with this problem, we also created WeDoCare.

WeDoCare is an Android system aimed at detecting violent attacks. The app collects data from motion and sound sensors from the smartphone from time to time, and if an attack is detected, it sends a distress signal to nearby people and policemen.

We will also test and analyze the system on which the former two were built upon: HappyWalk.

With these systems we expect to improve people's lives and show the potential of the Happy System to solve social issues.

1.2-Objectives

For the first semester, the main goals were to make the people involved in the project comfortable with the various technologies used and paradigms that are being researched.

David and other previous graduate students had already worked on Android apps (discussed in Chapter 3) that made use of the sensors available in today's smartphones to achieve people-centric sensing. Thus, it was planned that, during the first semester, we would build upon this previous work.

The main objective of these apps is to collect relevant data that can help users with their daily lives, by giving them helpful tips at useful times.

In the first semester the author also had the responsibility of building two apps, HappySPEAK and WeDoCare. These apps were built upon the Happy System, a Behavior Change Intervention system, which we will talk about in Chapter 3 of this report.

For the second semester we aimed to test the HappySPEAK and HappyWalk systems in terms of battery life and Artificial Neural Network (ANN) accuracy. By doing these tests we wanted to see if the developed apps were energy efficient, an important aspect of mobile development, and if the implemented classifier was effective in correctly categorizing the different emotions in our spectrum.

These tests would show if the developed apps were power efficient and if their ANN is accurate.

1.3- Report Structure

In this report we will start by presenting our objectives, the followed work plan/methodology and the technologies we used to achieve our goals.

In the second chapter we present important concepts and paradigms, and showcase some apps similar to the ones we are developing, showing what they do, how they work and how they compare to the ones we developed.

Chapter three contains the Software Engineering section of this report, where we will describe how the developed apps are structured, their architecture, features, functional and non-functional requirements.

Chapter four will expose how we implemented the features on the HappySPEAK and WeDoCare apps, by showing the resulting user interfaces, how the servers and web services were structured/built and how we connected all these components together.

In the fifth chapter of this report we will present our test results for battery life and ANN accuracy for the HappyWalk and HappySPEAK apps. We will also discuss the results and present our conclusions.

The sixth chapter revises the work done in this past year and explores what could be done in the future to improve the apps that were developed.

Finally, in the conclusions, we will discuss if the goals of the internship were accomplished and how we can improve the work we've done over this year based on the results we got from the tests.

1.4- Work planning and methodology

1.4.1-Work Planning

In terms of work planning, we had two weekly meetings throughout the year. One of these meetings was with the two supervisors, Professor Jorge Sá Silva and Master David Nunes and the other was with the whole research group, which included Ashley, a Biomedical Engineering student that worked closely with us.

In the individual meetings we discussed all of the work done in the previous week, talked about possible improvements we could do to refine the current project and were given new tasks for the next week.

In the group meetings all of the people in the investigation group would get together and make presentations to expose the current state of their projects, brainstorm ideas to improve what is already done, explore new possibilities and discuss if they were feasible in the available timetable.

1.4.2- Work Methodology

The work methodology used was based on Kanban, where the work flow was organized by states. Kanban helps developers visualize how their work is progressing by having several categories: "To Do", "Doing" and "Done".

Tasks are in the "To Do" phase when nothing is developed and are moved to the "Doing" phase when a team member starts implementing it. When the task is done it is moved to the "Done" section. This type of work methodology allows us to create more work phases such as "Testing", when a feature is being tested, or "Future features" when a feature is considered for future implementation. Each feature was developed to be as modular as possible, to allow easy integration in a later time. Tools like Trello, an online chore distribution service and Slack, a chat service aimed at development teams were also used to organize work distribution and discussions about the projects being worked on at the time.

Trello allowed the team to divide tasks by the following states, which have been inspired by SCRUM's "sprint task boards":

- TODO- The task still has not been developed.
- Doing- The task is currently being developed.
- Testing- The task is being tested.
- Done- The task has been tested and is finished. Changes may still occur, sending the task back to the "testing" state.
- Future Work- Features that could possibly be implemented at a future time.

It also allowed us attribute a task to a specific member of the team. The team was constituted by the following:

- Jorge Sá Silva
- David Nunes
- Ashley Figueira
- Ricardo Barbosa

Jorge Sá Silva and David Nunes were our supervisors, providing guidance throughout the implementation and testing of the projects. Ashley researched algorithms that were capable of detecting human emotions and did most of the work on developing the servers, because he had previous experience with the technologies used. Ricardo planned, evaluated, developed and tested the app's features.

We also used Git as a control version software to merge the various parts written by each team member and to be able to recover a previously working version, just in case something went wrong.

1.5- Technologies used

The projects developed throughout the first semester used several technologies. One of them was the use of smartphones with the Android operating system.

Android

Android is a mobile operating system (OS) currently being developed by Google. This OS is based on the Linux kernel and is designed to be used with touchscreen smartphones and tablets. Even though the base system is Linux, the OS runs on Java Virtual Machine designed specifically for this kind of devices.

Smartphones are perfect for the project we worked on because they are manufactured with a variety of sensors which can be used to extract important information about the user. This information can then be used to infer how the user is feeling and provide helpful tips to improve his/her day-to-day activities.

The primary programming language used to develop apps for this system is Java although we can use C++ to do some low-level programming.

Even though Android is fully compatible with Java there are many quirks associated with developing for this platform. Some of them are activity and fragment management, native tools only available in Android such as parsers, file access and memory management.

One of the reasons Android was chosen as the development platform was the previous work developed by David and other students, which provided us with some resources we could use to accelerate the project development. Part of that work was the HappyWalk app, which served as a basis for the apps we would build (we will talk about it further down in the report).

Even though the Android system has many helpful resources and a very large community that can help developers with their problems, it has some limitations that could harm the planned projects. Fortunately, there are many libraries with extended documentation and support that are compatible with it. The libraries we used were UBHave, Google Play Services and Encog. In the future we also hope to work with iOS, which will allow us to expand the number of devices the apps can run on.

Google Play Services

Google Play Services allows users to access an array of services, provided by Google, such as viewing world maps, locations of places in the real world, face recognition, among others.

We chose to use this library because it is continuously supported with new versions and functionalities, it has been extensively tested by Google and has a lot of documentation and useful tutorials.

UBHave

UBHave is a project developed for ubiquitous and social computing for positive behavior. It was developed at Southampton university with its main goal being to investigate the power and challenges of using mobile phones and social networking for Digital Behavior Change Interventions (DBCIs). [2]

The reason we used this library is that it was designed especially for DBCI systems on smartphones, allowing the developer to query information from sensors and manipulate it with less effort.

Encog

Encog is an advanced machine learning framework that supports many algorithms as well as mechanisms to normalize and process data. [3]

Encog has a Java implementation, which makes it compatible with Android development and was used in the HappyWalk app to infer how the user is feeling, making it a very good choice to use in this project.

Server

Another important part of the project is being able to transmit and save data in a server so that we could do some statistical analysis in the future. To do this we used a server previously created for HappyWalk. This server provided us with a basis that already had communication and database management. For database management it uses Hibernate and PostgreSQL.

Hibernate

Hibernate is a high-performance Object/Relational persistence and query framework licensed under a LGPL (Lesser General Public License). Its primary function is mapping Java classes to database tables while also providing data query and retrieval facilities. [4]

We used it to map the objects sent by the mobile device to the server from a Java class to a SQL record.

PostgreSQL

PostgreSQL is an open source object-relational database system. Some of its features include MVCC, point in time recovery, tablespaces, replication and transactions.

In the context of our work it was used to create the database and to query the various tables, allowing us to test if the data transfer between server and mobile device was successful and correct.

Trello

Trello is an online tool that has a visual way to manage and organize projects. A team can divide a project in tasks, and attribute tasks to the different team members. We used this website to have a visual representation of the project's progress, by dividing the tasks into several categories (as explained in the work and methodology section of this report).

Chapter 2

State of the Art

In this section of the report we will discuss concepts and paradigms that are used in the context of the developed systems, and some apps that are similar to the ones developed throughout the first semester. For each of those we will discuss their purpose, how they were created and how they compare to the ones we created.

2.1-Concepts and Paradigms

In this section we will discuss some of the paradigms associated with the projects we developed throughout this year.

Human-in-The-Loop

HiTL considers humans as vital part of the control loop. It can be divided in three main components: the human being, its environment and the system.

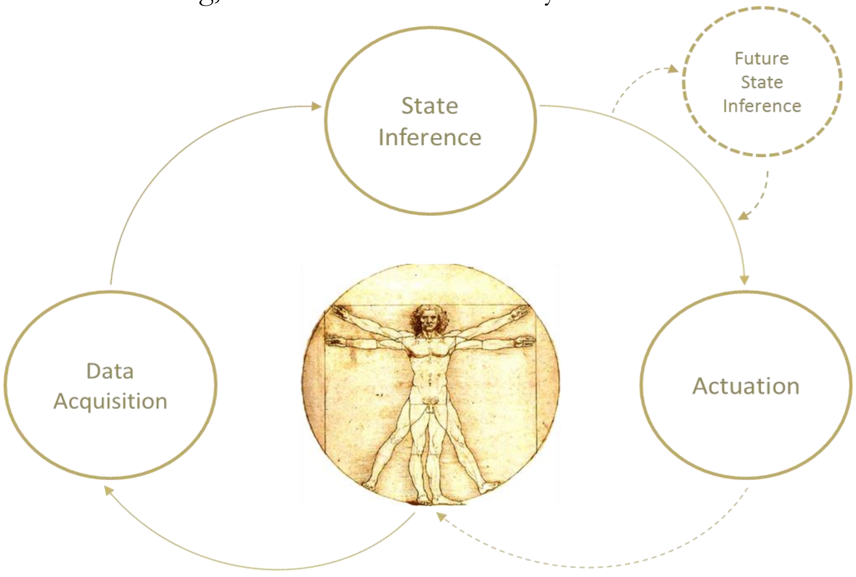


Figure 1-Human-in-the-loop Control Loop[7]

In the figure above we can see how the HiTL control loop works. Everything is centered on the human being:

- Data acquisition- Sensors collect information about the user and his environment.
- State inference- Infers how the user is feeling.
- Future State Inference- Uses previous information to improve its accuracy in future inferences.
- Actuation- The system tries to expose useful data to the user, in the form of information or suggestions.

There are many real-world applications that use this paradigm. Some of those are:

- Electronic Health systems that monitor patients to reduce human error. This system monitors, for instance, heartbeats and oxygen to calculate the appropriate amount of medicine that should be injected. [13]
- A wheel chair with assisted driving that avoid obstacles such as stairs or drops. [12]

- A memory jog system for supportive information in meetings and collaborative workspaces.[11]
- A system that suggests a better path to get to a destination, in order to avoid high traffic areas.[18]
- An electrical grid management system that uses information from its users to create and improve inter/intra-enterprise processes.[23]

Behavior Change Intervention

Behavior Change Intervention (BCI) systems aim to alter human behavior in a positive way. These interventions are mainly focused on giving advices and expose information that will help the user correct negative behavior (such as a sedentary lifestyle, smoking, over eating, etc.).[16]

This type of system requires a lot of information about the user and the places he goes to. Smartphones are extremely useful for this because they accompany the user for most of the day and have sensors that can be used to infer how he feels and the state of environment around him.

These interventions traditionally occurred in therapeutic sessions between two or more people, but are also starting to occur through the Internet and in smartphones.

Cyber-Physical Systems

Today's technology has evolved to a point where machines and sensors communicate with each other seamlessly, giving way to the so called "Internet of Things". This allows new types of Cyber-Physical Systems (CPSs) to be developed. While CPSs today are built around human interaction, many of them just consider humans as external, unpredictable element to the control loop.

Other than e-Health, there is not much scientific work done about the human context in the control loop of CPSs [11]. Smartphones present an excellent opportunity to do so. They possess the processing power and the array of sensors required to develop CPSs to the next level. In short, when we start using this type of mobile devices to evaluate and monitor human nature, humans become a part of CPSs. This brings us to Human-in-The-Loop Cyber-Physical Systems (HiTLCPS).

By inferring emotional and psychological state of the user, as well as actions and behavior we can increase the accuracy of the control-loop. For instance, let's take into account cruise control systems. If the user is tired, it could suggest to him that maybe he should consider turning cruise control on [11]. Since human beings are considered to be extremely unpredictable this presents an enormous challenge.

Recommender Systems

Recommender Systems (also known as Recommendation Systems) have become widespread in recent years. Many well-known companies are using them to improve their services and expose new products to their customers. These include Amazon, that uses content-based recommendation to suggest items the user may want to buy, based on previous purchases; Netflix combines 107 recommendation algorithms to form a single prediction and suggest new video content to their users. Many other companies like Facebook, Twitter, Google and LinkedIn use recommendation engines to expose interesting information to their users[14].

These systems usually use one of two approaches: Collaborative Filtering or Content-based filtering. Hybrid approaches also exist.

Collaborative Filtering is based on user behavior, and can construct its model from it or from the collective behavior of users with similar traits. In essence, it aggregates users based on their preferences and recommends content based on it.

Content-based filtering constructs its recommendation on the basis of a user's behavior. For example, if a user reads a lot of articles about Linux, and comments on them, the system can then recommend other similar articles.

Recommendation Systems use a variety of algorithms, from clustering algorithms (K-means, Fuzzy logic, Expectation-Maximization) to Bayesian Nets and Markov chains.[14]

In the work we have developed throughout the year, we recommend places the user can visit to improve its mood, but the approach is based on the location of the person instead of his interests and preferences. We can further improve the systems we built by using information users post on social networks, which will have to be done at a future time.

Emotional Maps

Emotional Mapping is a methodology that helps to visualize and analyze human reactions to the environment. Typically, it uses Bio-Sensors to extract information about the subject's body. Today, biosensors have a variety of applications from glucose monitoring, sensing airborne bacteria and pathogen detection, among many others. [15]

Emotional maps are also being used with GPS trackers in order to build maps that represent the emotional state of a region/country. Using this information, it is possible to build maps that visually represent the needs of people in an area, allowing for better resource allocation analysis which leads to cost-effective solutions.

Artificial Neural Networks

An Artificial Neural Network(ANN) is a machine learning model, inspired by biological neural networks. In the field of machine learning, it is used to predict an output based on a set of inputs, using a mathematical function.

This model uses three components: an input layer, one or more hidden layers and an output layer. The input layer is used as a data transmitter to the hidden layers. This data is composed by one or more parameters, known as "features". Each feature is attributed to an input neuron which in turn transmits it to each neuron of the first hidden layer.

Each hidden layer contains multiple neurons, all connected to the neurons of the next hidden layer. These neurons store parameters called "weights", which are adjusted using training data. The last hidden layer transmits an output to the output layer, which in turn, outputs a result.

Training data can be described as a set of features that represent a specific output, and is used to "train" the network in order to help it to correctly classify an output with higher accuracy.

One of the models that has been surging in these past years is Deep Learning, which has revolutionized many domains of signal and information processing (such as computer vision, speech and object recognition).[17] This type of machine learning model has already been used in smartphones for speech recognition models (such as Google Talk), but most of the computing is done in the cloud. Usually, deep learning is not used in smartphones locally due to latency and energy constraints.[17].

Research done in the field of Emotion Sensing using DNN's mostly focus on emotion inference using Speech Recognition[24][17][25].

The main difference between DNN and ANN are the number of hidden layers. DNN possess many more layers and can be trained in a supervised or unsupervised manner while an ANN can only be trained with supervised training.[21].

In the projects we developed, the ANN was the model used to classify user emotions using features collected in the user's smartphone. Network training was achieved by using a set of features and asking the user how he felt at that time.

2.2-Similar Apps

In this section of the report we will present some apps that are similar to the ones we built throughout this year, and explain what they are, how they work and how they compare to the ones we developed.

StudentLife

What is StudentLife?

StudentLife was the first study that used passive and automatic sensing data from student phones to assess their mental health, academic performance and behavioral trends, among others. This study was done by researchers at Dartmouth University with the goal of discovering the factors which affect academic performance the most. To do this, the team of researchers developed an Android app that collected sensor data and distributed it by 48 students over a 10-week term. [5]

How does StudentLife work?

By using machine learning algorithms to interpret sensor data and making inferences about user behavior (without any user interaction), they measured the following factors daily:

- Number and duration of conversations.
- Number of people around the student.
- Bed time, wake up time and sleep duration.
- Physical activity
- Where they were and for how long (in campus).
- Outdoor and Indoor mobility (in campus).
- Stress level.
- Positive affect.
- Eating habits (where and when they ate).
- App usage
- Comments about campus and national events.

With this data, they used mental health surveys and student GPA as ground truth to evaluate mental health and academic performance, respectively. They were also able to predict student's GPA by using the same data.

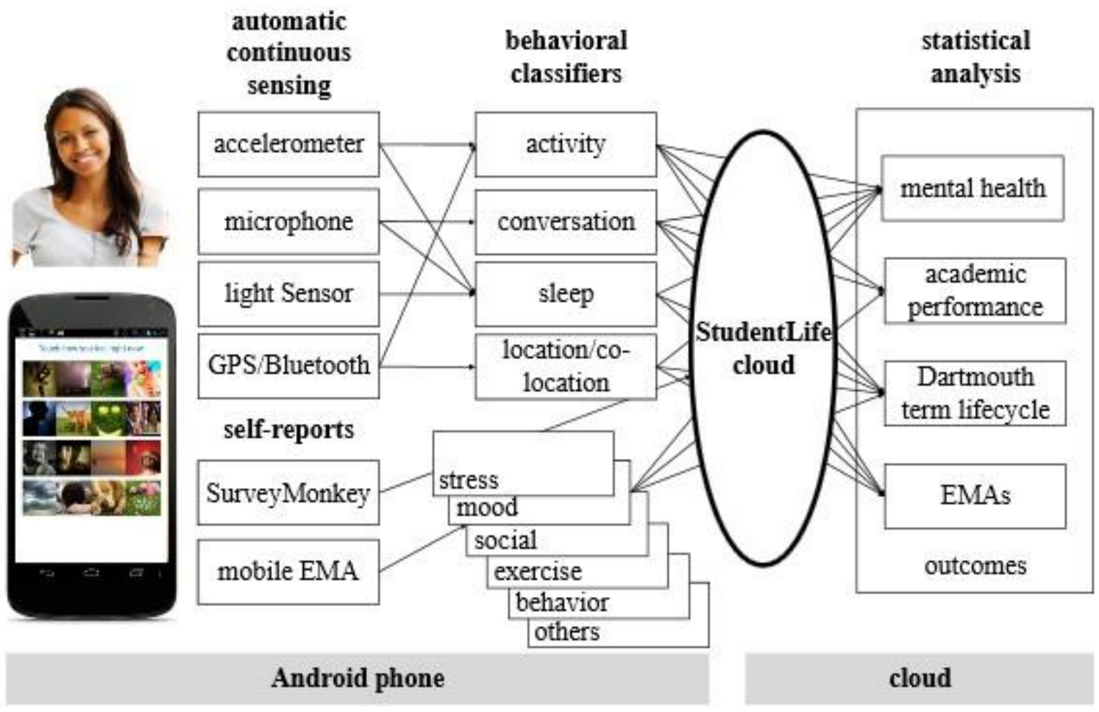


Figure 2-Component Interaction in Student Life[5]

We can see in the image above how all the sensor and self-reports interact with the behavioral classifiers. All of the data collected is sent to the StudentLife cloud and saved to do statistical analysis.

Even though this app was a big success because it allowed the researchers to see the activity trends of the students, it still lacks the mechanisms to impact the student in a positive way. A second version is being worked on that aims to give students guidance throughout the year, using the data it collects.

EmotionSense

What is EmotionSense?

The EmotionSense app was developed by a research team at Cambridge, for the Android OS, with goal of monitoring the users' emotional state. It achieves this by using the sensors present in today's smartphones, which accompany the user throughout the day.

How does EmotionSense work?

This app collects data from GPS, microphone, accelerometer and patterns from calls and messages. It also asks questions via questionnaires and an emotion grid. This allows correlating the data collected by the smartphone with the users' mood changes.

HappyWalk's emotion grid was heavily influenced by this app. Unfortunately, this app simply monitors human mood and only offers the opportunity to check a history of the collected data and the inferred result. It does not provide any sort of proactive attitude towards the user, which is what HappySPEAK and WeDoCare try to do.

Mapiness

What is Mapiness?

Mapiness is an iPhone app that is a part of a research project at the London School of Economics. It aims to tell the user useful information about his own happiness by studying how his happiness is affected by his environment, which includes factors such as air pollution, noise, green spaces among others.[22]

How does Mapiness Work?

The user can configure the app to only ask how he is feeling a specific number of times per day and at what time. The feedback requests ask how happy/relaxed/awake the user is in a scale of 1 to 10. It also asks where the user is, who he is with and what he is doing.

The feedback given by the user is then aggregated and compiled into a series of charts that shows the progression of the user's mood over time and a contextualized view of with who and where he feels the happiest.[22]

This app only acts as an informational tool and does not provide any actual suggestions for the user to improve his mood.

What is Motorola Alert

Motorola Alert is an emergency alert app that allows the user to designate contacts that will receive an alert when he is in danger. It is not meant to be used as a tracker app, as it just notifies other people that the user is in danger, or may be in a risky situation. Currently it has three modes:

- Meet Me: Sends a message to the designated contacts to meet the user in its current location.
- Emergency: Broadcasts a warning to all designated contacts informing them that the user is currently in danger. It allows the user to call the Emergency Services by pressing a button.
- Follow Me: Sends the periodical location information to a designated contact. This interval is also defined by the user.

It also allows the user to define places he visits regularly and notify other people when he is there.[20]

How does Motorola Alert Work?

This app uses GPS and Wi-Fi to track the user's location and standard messaging mechanisms to notify other people.

The location obtained by the smartphone is then converted to an address using a strategy called geocoding. Geocoding takes a location (usually latitude and longitude) and uses an online service to figure out the address it belongs to.

When activated, the emergency button starts a 5 second countdown that allows the user to cancel the alarm. If this countdown gets all the way down to zero, the alert is broadcasted to all designated contacts.[20]

Something to note about this app is that it is completely manual, meaning that the user must explicitly send an emergency signal. This may not the optimal solution in a danger situation; e.g: if the user is caught off guard, or if his cell phone is stolen. The fact that only designated contacts are notified may also be a problem. If notification receiver is far away, it may be unfeasible for him to help the user. Also, relying on just messaging to notify other people may pose problem in places with no network or when the user has no money to send the message.

The WeDoCare app relies on several mechanisms that fix these problems, like using Bluetooth and Wi-Fi Broadcasting to notify all nearby people with the app. It also automatically classifies attacks, sends messages and calls the emergency services.

Nike Running

What is Nike Running?

Nike Running is an app that logs the user's progress when he goes out to run. The app allows users to add friends and compare running results such as number of times the user went out running, who has the greatest traveled distance, etc. This incentivizes to user to run more frequently, thus maintaining a healthier lifestyle. The app also registers the speed, distance and travel time of each run.

How does Nike Running Work?

This app uses GPS and accelerometer to record distance and speed. The values gathered are then converted to a path that represents the places the user went through. This path changes color in the places the user ran faster or slower (green for lower speeds and red for faster speeds).

WeDoCare has a similar feature using heat maps; green zones are considered to be safer and red zones are considered to be dangerous. Nike Running also presents static information as opposed to WeDoCare, who generates content automatically via crowdsourcing.

Yelp

What is Yelp?

Yelp is a service directory that shows reviews for places like restaurants, gyms, parks, etc. These reviews are submitted by the platform users and displayed on the place's page. Users can classify places and leave short reviews that describe their experience while visiting the establishment. There is also a "competition" component to this platform; users receive badges if they complete certain objectives, like reviewing a certain amount of places.[19]

The owners of these places can see the reviews posted by the users and interact with them to improve their service.

How does Yelp work?

Yelp takes the information given by users and aggregates it in the page of a specific place. This platform uses the concepts explained in the "Recommender Systems" section, by using the places that the users have previously visited, recommending new ones that he may enjoy. The crowdsourcing approach is comparable to the one used by the WeDoCare app, but instead of using user created content to review places, it uses it to display dangerous places and notify the user if he approaches one.

Chapter 3

The Happy System

3.1-The Happy System

In this section we will expose who is working on the Happy System, how the architecture of the system is organized and how the HappyWalk and HappyHour apps work. HappyWalk and HappyHour are both based on the Happy System. HappyWalk served as case studies/basis for the apps we built.

3.1.1- The Happy System Architecture

For a system of this nature to work properly we primarily need a sensor array that allows us to measure human behavior and his environment, and optionally, a server to save and process information. Fortunately for us, today's smartphones come equipped with many sensors, such as:

- Accelerometers- Provides information about user movement.
- Microphone- Can measure the amount of activity around the user.
- GPS- Allows us to see where the user is, at what time and consequently at what speed he is moving.
- Temperature-Measure ambient temperature.
- Ambient Light – Measures the amount of light around the smartphone.

In the figure below we can see a generic interpretation of how a Happy System is structured:

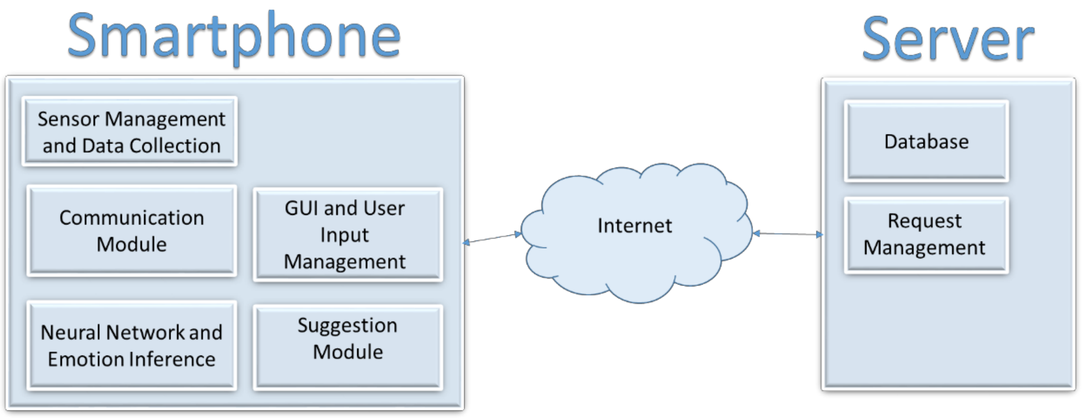

Smartphone
Sensor Management and Data Collection
Communication Module
GUI and User Input Management
Neural Network and Emotion Inference
Suggestion Module
Internet
Server
Database
Request Management

Figure 3-Happy System high-level architecture

In the smartphone system, we have four different components:

- Sensor Management and Data Collection- This module manages when the sensors must be activated to collect data. It also processes the data and saves it into the appropriate data structures.
- Communication Module- Establishes connection with the server and handles all incoming/outgoing requests.
- Neural Network and Emotion Inference- A neural network is a model inspired by biological neural networks. It is trained by giving it a training set which contains a group of inputs and the desired output. These training sets adjust the network allowing it to classify inputs with the correct output more often. In the Happy System the neural network is "fed" with sensor data and infers how the user is

feeling. If a specific output is detected, the app will give suggestions on how to improve the users' life.

- GUI and User Input management- Present the several user interfaces and handles all of the user input.
- Suggestion Module – Generates a suggestion for the user to improve his behavior.

In the server we have two different components:

- Database- Saves incoming data from the user and saves it for future analysis.
- Request Management- Handles all incoming/outgoing requests from the user.

3.1.2- HappyHour

HappyHour was developed to help users find the right place for night entertainment. It worked on the premise that the app could collect smartphone sensor information from many different users and use it to show how a specific area was felling.

HappyHour also integrates information via social networks, to find out what each user likes the most in terms of:

- Music
- Band
- Drinks

By collecting this type of information, it was expected for the app to give better suggestions to the user.

This system allowed businesses to implement marketing techniques by using innovative technologies, like notification scheduling to share promotions or events to potential clients in the vicinity. [6]

This system was composed by three main components:

- The HappyHour app- Client side application in which the user could access a map to obtain points of interest and information about them.
- Web Backend- Allowed employees to insert information about the place and add notifications about promotions or events.
- Central Server –Used for user management, information aggregation and parsing and event/promotion notification.

The app was installed on Android mobile devices, and it informed the users about the location and type of night time entertainment.

It collected GPS, accelerometer and noise data and aggregated it on a central server in the form of heat maps, presented to the user on top of a map

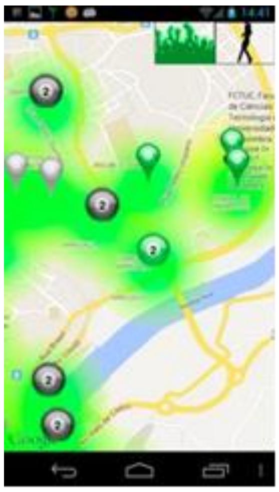
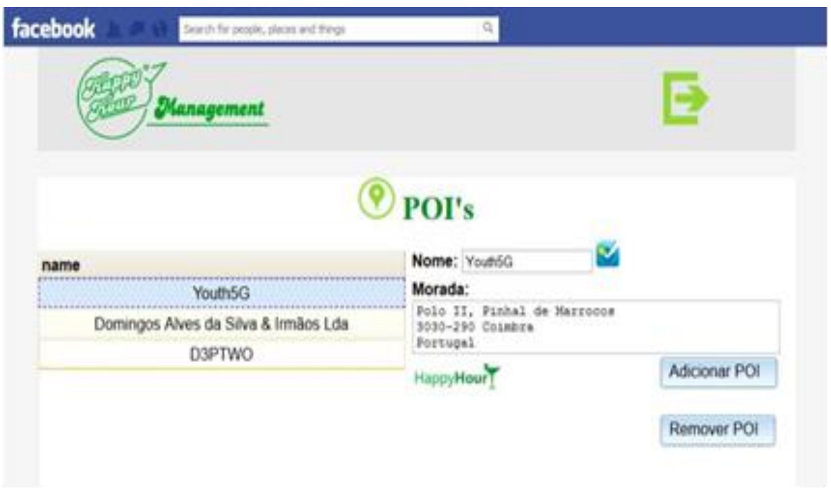


Figure 5-HappyHour Heat Maps[6]



name
Youth5G
Domingos Alves da Silva & Irmãos Lda
D3PTWO

Nome: Youth5G
Morada:
Polo II, Pinhal de Marrocos
3030-290 Coimbra
Portugal

Figure 4-HappyHour Employee backend[6]

What really made HappyHour stand out from other mobile social apps was the real time information sharing and the lack of active and manual user participation.

3.1.3-Happywalk

Figure 6-HappyWalk Logo[7]

HappyWalk is a Behavior Change Intervention system that estimates the users' current mood and tries to improve their physical and mental well-being. It does this by monitoring user activity, using smartphone sensors, and inferring the users' mood. If the system detects that the user is in a bad mood, it will suggest that he takes a walk by displaying several points of interest in a map.

This app, like the ones we developed, employs a full closed HiTL control- loop as shown in figure 7.

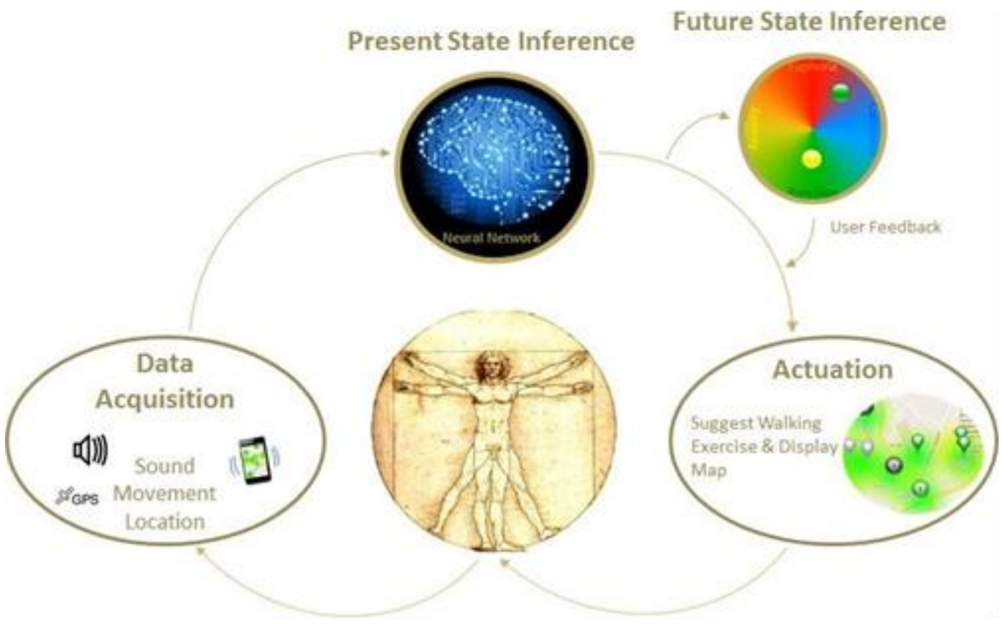


Figure 7-HappyWalk HiTL control[7]

How does HappyWalk work?

HappyWalk uses data from the smartphone's microphone, accelerometer and GPS sensors as inputs in a neural network. The output will be one of four emotions: Euphoria, Boredom, Calmness and Anxiety. This system considers euphoria and calmness as positive emotions so it only gives the user suggestions if it detects Boredom and Anxiety. We can see the network's several layers in the next figure:

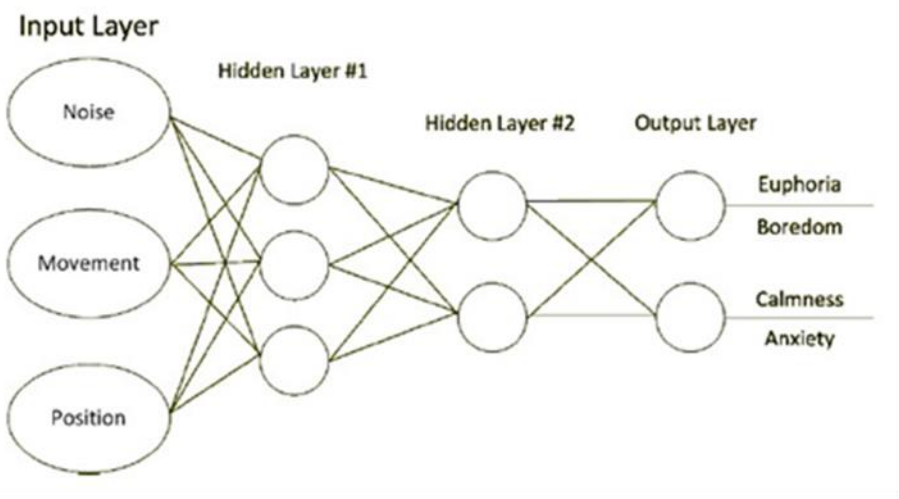


Figure 8-HappyWalk Neural Network Diagram[7]

In order to train the neural network, the system occasionally queries the user on how he is feeling. The app tells the user that it has detected an emotion and promptly asks the user how he is feeling by showing him a notification. If the user clicks it another menu is shown:

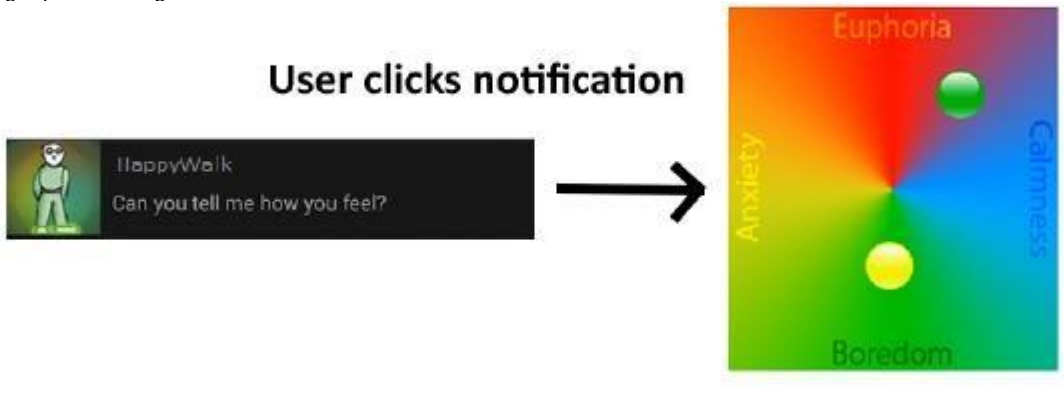


Figure 9-HappyWalk Feedback Request[7]

In the previous figure we can see a Feedback Request from the app. The user can drag the green circle to a place he feels represents his current mood. This input is saved and added to the neural network's training set. The yellow circle represents the mood that the system thinks the user is at that moment.

By asking the user for feedback, the network will require less and less training, and as accuracy grows, the number of times feedback is requested also drops.

HappyWalk Architecture

The HappyWalk app is based on the Happy System architecture. It uses the foursquare API to get data for its suggestion module. Foursquare is a company that uses location intelligence to improve business solutions and consumer experiences. The API provided by them allows developers to access data about places and businesses and interact with its users and merchants. In the context of HappyWalk it was used to show the user possible locations he may like to walk to.

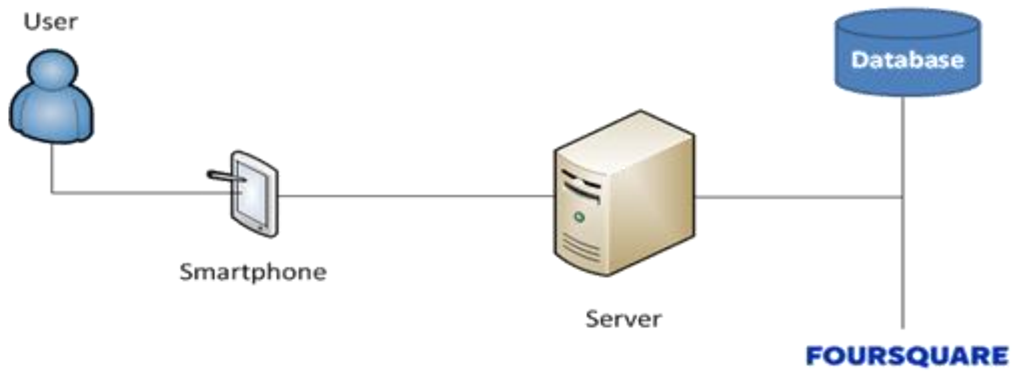


Figure 10-HappyWalk component interaction[7]

To improve performance, this system uses a clustering algorithm that groups Points of Interest, instead of showing them all on the map.

The app's behavior is represented in figure 11:

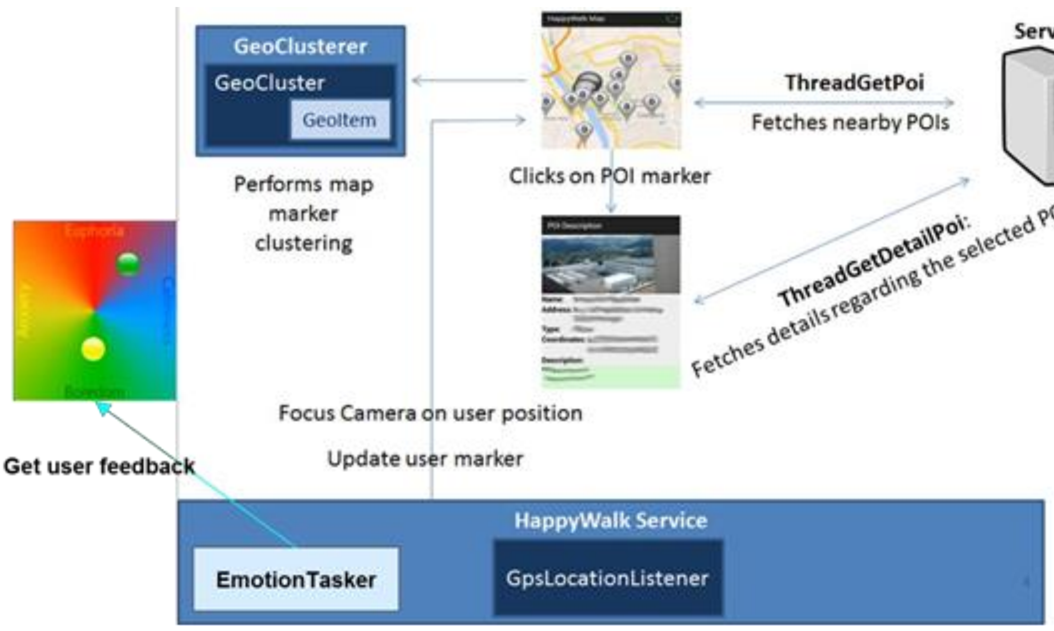


Figure 11-HappyWalk behavior[7]

The sensor module collects data from the accelerometer and microphone to infer the user's emotional state and sends that information to the neural network. The sound waves amplitude is averaged and the values from the accelerometer undergo a Fast Fourier Transformation. These resulting values are used as inputs to the neural network and classified:

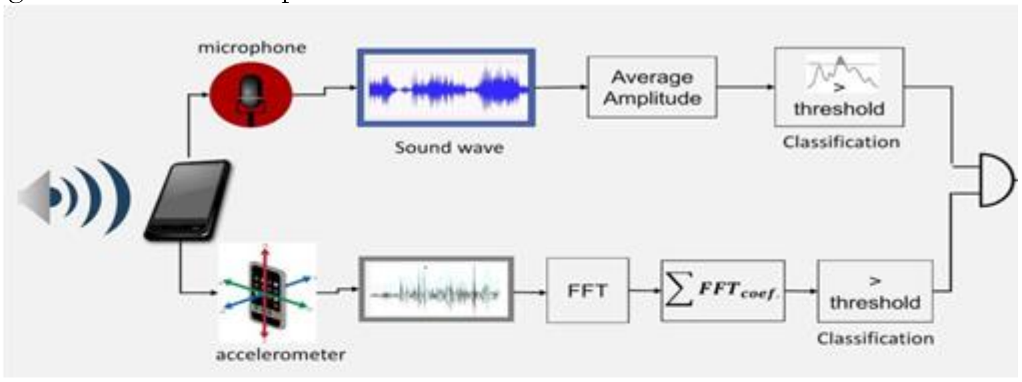
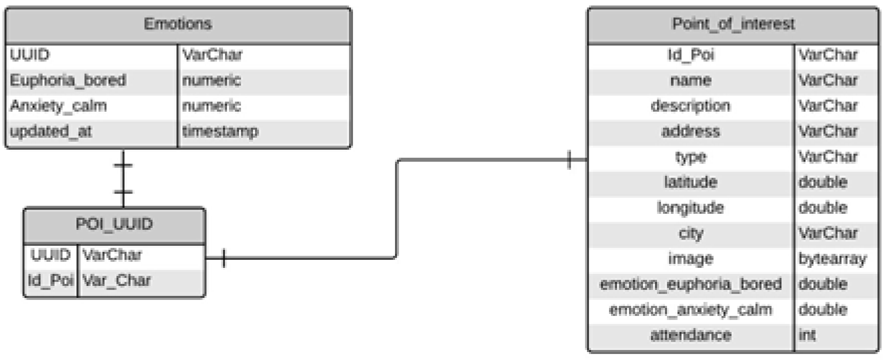


Figure 12-HappyWalk emotion classification[7]

The HappyWalk Database can be represented by the following Entity-Relationship Diagram:



Emotions	
UUID	VarChar
Euphoria_bored	numeric
Anxiety_calm	numeric
updated_at	timestamp

POI_UUID	
UUID	VarChar
Id_Poi	Var_Char

Point_of_interest	
Id_Poi	VarChar
name	VarChar
description	VarChar
address	VarChar
type	VarChar
latitude	double
longitude	double
city	VarChar
image	bytearray
emotion_euphoria_bored	double
emotion_anxiety_calm	double
attendance	int

Figure 13-HappyWalk Database Diagram

As mentioned before, HappyWalk was used as groundwork for the projects developed throughout the first semester, because it already provided a structure on which we could build upon. The fact that it already possesses the mechanisms to collect and process data from the sensor implies that we can make significant changes to the system with little difficulty.

In the second semester, we created a set of tests to analyze this app's performance and ANN accuracy. We decided to test this app due to the fact that it was used as the groundwork for HappySPEAK and WeDoCare. The results can be found in Chapter 5 of this report.

3.2-Software Requirements Specification

In this section we will discuss all of the requirements we took into account for the development of the HappySPEAK and the WeDoCare apps.

3.2.1-HappySPEAK

Product Scope

In the past decades the world has been witnessing a huge increase in the number of migrants, searching for better standards of life. The reasons that make people migrate are varied, and can be classified as:

- Economical
- Social
- Political
- Environmental

Moving to a different country isn't easy. Cultural differences and linguistic barriers make it harder for migrants to connect with citizens of the hosting country. This can lead to migrants feeling lonely, especially if they came to a new country alone.

This is where SPEAK tries to help. First it tries to connect them with people in the same situation, by organizing dinner parties and events that brings them together and makes them learn more about one another.

Context

SPEAK[1] is a cultural program designed to bring people together by promoting multiculturalism and the democratization of language learning. It provides courses and encourages people from different cultures to share their interests and break any prejudices they may have about foreigners.

By partnering with SPEAK, we can have access to a control group that can test our system, receive feedback about usability, missing features and study the accuracy of isolation detection.

As we discussed before, HappySPEAK aims to measure human isolation and help people who feel this way by showing them SPEAK events that are coming soon while at the same time allow people to have access to the SPEAK platform directly in their smartphones.

To measure this type of human emotion it collects smartphone data (call/message logs and SPEAK class attendance). By collecting these types of data we will be able to infer human isolation and help people by giving them helpful tips.

Stakeholders

Stakeholders are the people that interact with the app in a direct and indirect way. For this app we defined the following stakeholders:

Stakeholders	Who are they	What do they do
HappySPEAK development team	Jorge Sá Silva David Nunes Ricardo Barbosa Ashley Figueira	Jorge and David are the team supervisors. Ashley researches algorithms and papers about inferring human emotions and helps implementing the software. Ricardo plans, evaluates and implements the software.
SPEAK	Hugo Menino Aguiar	SPEAK CEO and Co-Founder

SPEAK Users	Migrants	Migrants who want to connect with new people and learn/teach about new/their culture and language.

Table 1-HappySPEAK Stakeholders

Product Features

SPEAK users are the main actors of this project. They are the ones that are going to use the app the most to find out about SPEAK events. For these users we defined the following features:

1. Login- Users must be registered in the SPEAK database to be able to see current and past events.
2. Register- If the user is not currently registered in SPEAK, they must be able to do so from their smartphone.
3. See SPEAK Social platforms- Takes the user to Facebook, Youtube and Twitter pages created for SPEAK.
4. See SPEAK Events – The user can check a list of SPEAK events that have and will occur.
5. See Event detail- This feature allows the user to learn more about the event, such as where it is located, how to get there, time and date, who is going and who organized it. It also allows users to subscribe/unsubscribe to this event.
6. Subscribe to SPEAK Events- Allows the user to say that he is going to that particular event.
7. Unsubscribe to SPEAK Events- If the user no longer wishes to attend the events he can unsubscribe from them.
8. Collect Data- The system will collect data to check if the user feels isolated.
9. User Feedback –From time to time the system will ask the user how he is feeling.
10. Give a Suggestion- When isolation is detected, the app will show the user SPEAK events that will occur soon, as well as places he can go to.

User Profiling

SPEAK users consist mostly of foreigners who wish to meet new people and linguistic enthusiasts who want to learn how to speak a new language. This implies that the app must be "readable" by people from many different backgrounds, and more importantly, from many different languages.

Contrary to popular belief, migrants who come from poorer countries are used to work with this type of mobile devices and have been for quite some time.

Operation Environment

The app is being developed for the Android OS, from version 2.2 to version 5.1.1(there are still some compatibility issues with the new 6.0 permissions structure). This implies that it must be able to run on a large number of devices, with different software and hardware configurations. In 2015 there were more than 24000 different devices on the market [8]. This, in conjunction with the fact that there are still 15 Android versions (v2.2. to 6.0) still supported by Google, presents a huge challenge in device fragmentation.

These restrictions were taken into account during the development of the app.

Design and Implementation Restrictions

Our limitations when developing this app were mostly due to Android system limitations. These include the programming language (Java), and adapting the software to run on a large number of devices, all with different hardware and software specifications

Another restriction was the matter of privacy. We must implement privacy mechanisms that prevent the linkage between the user's identity and the data being collected.

There is also the matter of how the graphical interface must be built. Hugo has instructed us to make it as close to the website as possible.

Figure 14-SPEAK Website

Dependencies

The features currently assigned to this project can change at any moment. During the weekly group meetings, it can be decided to add, remove or refine features to this app. We also have two test phases planed. In the first test phase, our research group will test the app give feedback about was can be improved in terms of usability. The second test phase will consist of battery life/ ANN accuracy tests. Battery tests will be done using a test phone, and the accuracy tests will be performed on real-world users.

Another dependency is the use of the Google Play Services library. At this time this library is in version 8.6 and the minimum version number required for the app to work correctly is 6.5. Users can choose to not upgrade their Google Play Services so mechanisms must be developed to allow the app to function normally if their current version is not in this spectrum.

User Interfaces

The design process started by exploring the existing site, specifically the Events and Subscription modules. We began by taking into account Hugo's directives (the mobile app UI must be on par with the website's design) and the modules we were to develop. With these restrictions into account we developed the following User-Environment Design Model:

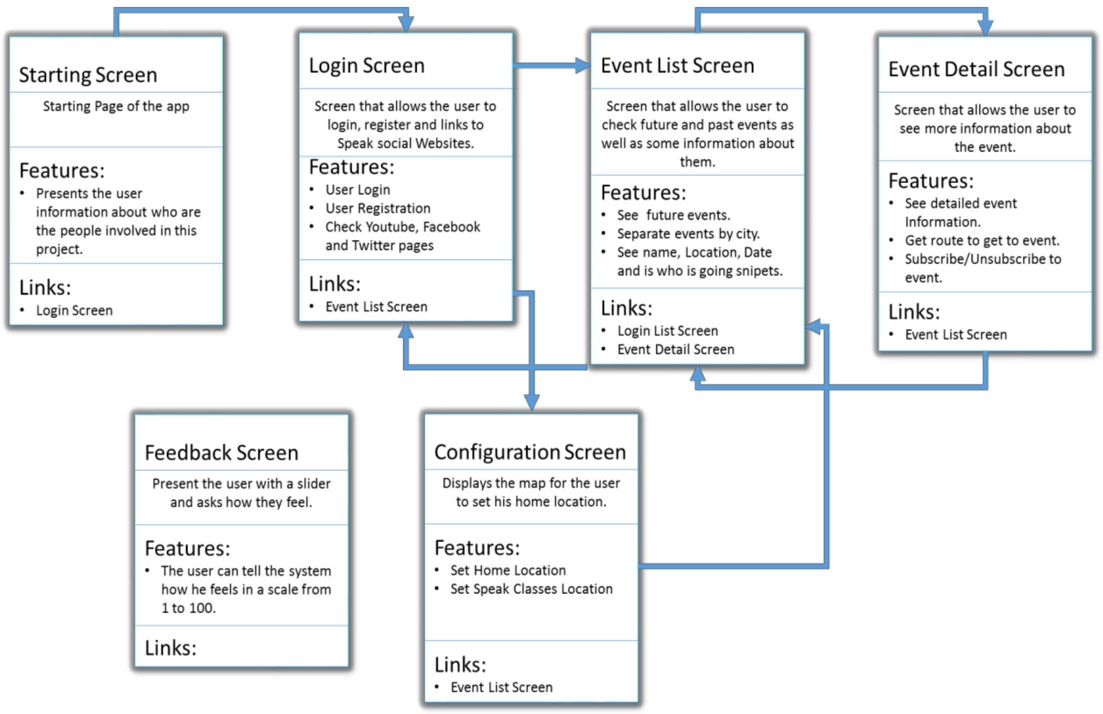

Starting Screen
Starting Page of the app
Features:
• Presents the user information about who are the people involved in this project.
Links:
• Login Screen
Login Screen
Screen that allows the user to login, register and links to Speak social Websites.
Features:
• User Login
• User Registration
• Check Youtube, Facebook and Twitter pages
Links:
• Event List Screen
Event List Screen
Screen that allows the user to check future and past events as well as some information about them.
Features:
• See future events.
• Separate events by city.
• See name, Location, Date and is who is going snipets.
Links:
• Login List Screen
• Event Detail Screen
Event Detail Screen
Screen that allows the user to see more information about the event.
Features:
• See detailed event Information.
• Get route to get to event.
• Subscribe/Unsubscribe to event.
Links:
• Event List Screen
Feedback Screen
Present the user with a slider and asks how they feel.
Features:
• The user can tell the system how he feels in a scale from 1 to 100.
Links:
Configuration Screen
Displays the map for the user to set his home location.
Features:
• Set Home Location
• Set Speak Classes Location
Links:
• Event List Screen

Figure 15-HappySPEAK User-Environment Design Model

Fortunately for us, the website already had mechanisms that transformed the web- page's graphical interface to a mobile-friendly interface. This allowed us to model most the screens we needed without building all the mockups ourselves. Therefore, using the Android Studio Development Environment and the screenshot tool, we were able to get the models for the following pages:

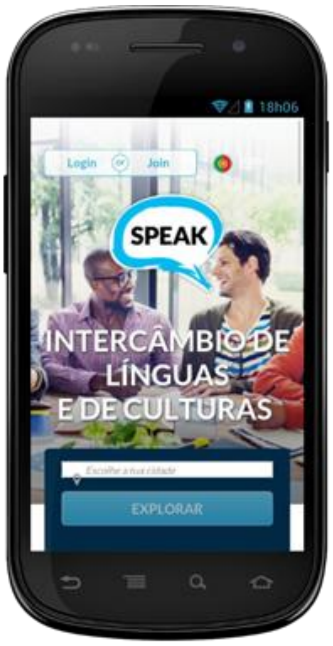


Figure 16-Login Screen Mockup

This is the mockup for the login screen. The most important elements of this are the Login and Join buttons that allow the user to access his account or create a new one. Each of these buttons opens a pop-up that allows the user to insert the data required to perform these operations. These pop-up boxes can be examined in the figures below.

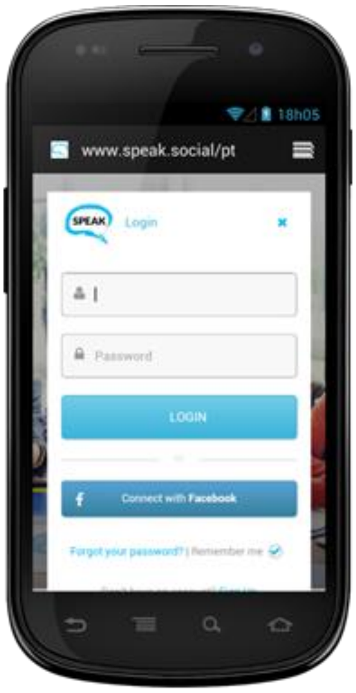


Figure 17-Login Dialog Mockup

This is the Login pop-up. It has two text fields, one for the username and another for the password. It also has two buttons, one for submitting the data and another to login using Facebook credentials. For the time being, we only need to use the normal login button and the two text fields.

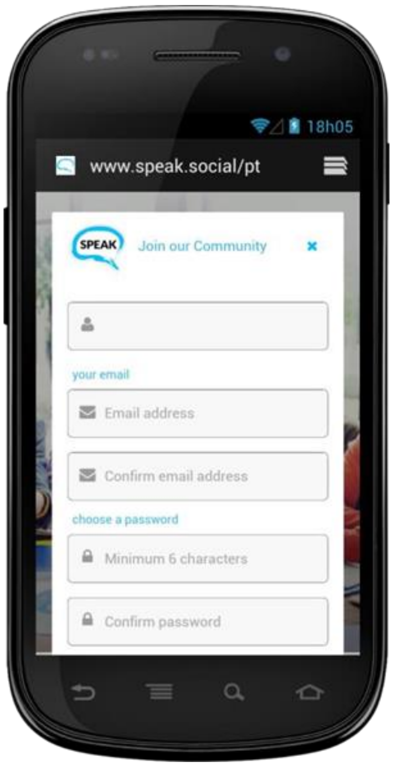
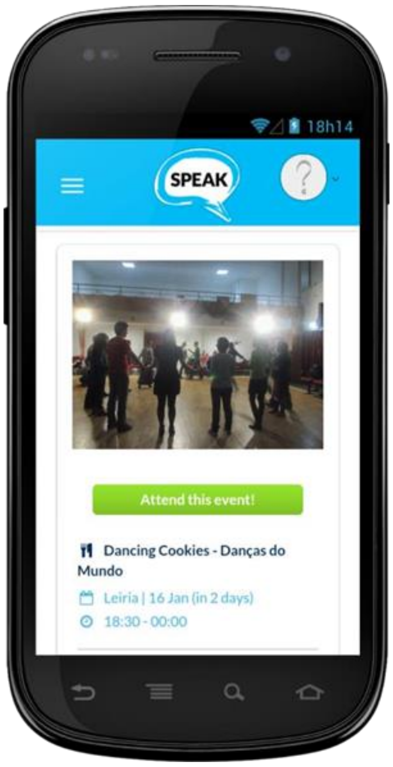


Figure 18-Registration Dialog Mockup

This is the Register pop-up. It has all the necessary fields for user registration like Username, Email/Email confirmation, Password/ Password confirmation and a submit button.


18h14
SPEAK
Attend this event!
Dancing Cookies - Danças do Mundo
Leiria | 16 Jan (in 2 days)
18:30 - 00:00
Figure 19- Event List Mockup

This is the Event List Screen. It presents information about future and past events such as: the name of the event, where and when it will happen and a subscribe button. The subscribe button will allow the user to register for that event.

This screen also shows who is going, if such people exist and a short description (limited to 50 characters).

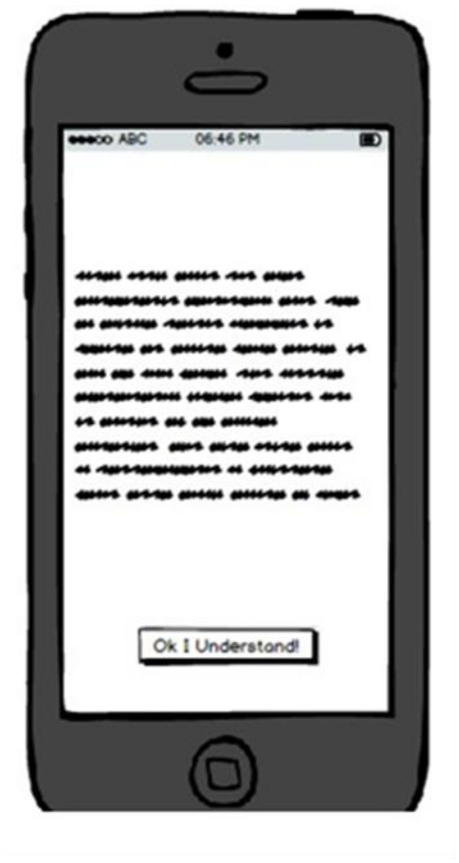


Figure 20-Information Screen Mockup

This mockup represents the information screen. This screen must show the user information about the goals of the app and the entities involved in its development. It must also inform the user what is the data that will be collected, how it will be used, and reinforce the anonymity policies.

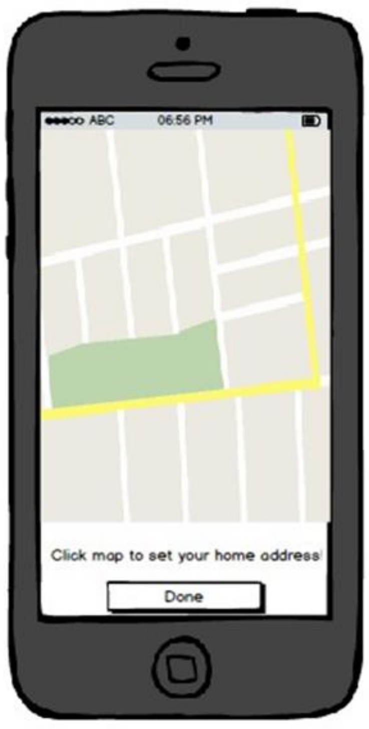


Figure 21- Home Configuration Screen

This mockup represents the home configuration screen. It presents a Google map that allows the user to indicate where his home is. When the user clicks the map, a marker will appear representing the location of his home. This marker must be draggable to correct mistakes. When the user is done he may click the "Done" button to save.

Software Interfaces

The software interface the system must have are the following:

- The app shall communicate with the SPEAK webservice to get any and all information about the user and the events.
- The app shall communicate with the HappySPEAK server to store all information about the neural network values and the raw data collected by the sensors, as well as feedback provided by the user.

Communication Interfaces

The communication interfaces the app must have are the following:

- The app shall use SOAP to communicate with the Webservice because the SPEAK platform is only compatible with this protocol.
- The app shall use REST to communicate with the HappySPEAK server because the HappyWalk server is already built with the mechanisms to support it.

Feature Description

In this section of the report we will present the features for the HappySPEAK System and give a more in-depth look on how they should work.

Feature 1-Login

Description

Users must be registered in the SPEAK database to be able to see current and past events. The app must allow the user to insert a valid username and password to proceed to the next menu.

Action and reaction

1. The user clicks on the "login" button.
2. The user inserts his username and password
3. The user clicks "Submit" button
    a. If the username/password is correct, it goes to the Event List menu.
    b. If the username/password is incorrect, an error message should be shown.

Functional Requirement

Requirement ID	Description
1	If the username or password do not exist on the server, the user cannot proceed to the next menu.
2	If the username or password are not valid, a message must appear to inform the user.
3	If there is no connection to the Internet a message must appear to inform the user.

Table 2-HappySPEAK Login Functional Requirements

Feature 2-Register

Description

If the user is not currently registered in SPEAK, he must be able to do so from his smartphone.

Action and reaction

1. User clicks the "Registration" button.
2. User inserts a username, password, confirmation password and e-mail.
3. User clicks "Submit" button.

a. If the email/username are already taken, an error message should be presented.

Functional Requirements

Requirement ID	Description
4	If the username inserted is already in use, the user cannot be registered.
5	If the email inserted is already in use, the user cannot be registered.

Table 3-HappySPEAK Registration Functional Requirements

Feature 3-See SPEAK Social platforms

Description

Takes the user to Facebook, Youtube and Twitter pages created for SPEAK.

Action and reaction

1. User clicks Facebook, Youtube or Twitter icons.
2. The app opens the browser with the right link.

Functional Requirement

There are no function requirements for this feature.

Feature 4-See SPEAK Events

Description

The user can check a list of SPEAK events that will or have occured.

Action and reaction

1. User Logs in the system.
2. The user is presented with a list of events.

Functional Requirement

Requirement ID	Description
6	Events must be separated by Cities.
7	If a city does not have any events, the text "No events for this city yet" must be displayed.
8	If no one is going to that event the text "Who is Going?" must disappear.
9	The description section must only present 50 letters at most.

Table 4-HappySPEAK See Events Functional Requirements

Feature 5-See Event detail

Description

This feature allows the user to learn more about the event, such as where it is located, how to get there, time and date, who is going and who organized it. It also allows users to subscribe/unsubscribe to this event.

Action and reaction

1. The user clicks a specific event of the list.
2. The user is presented with the event information.

Functional Requirement

Requirement ID	Description
10	The subscription status of the user must be known to present the status of the "Subscribe" button. If the user is not subscribed the button is green; if he is, it must be gray.

Table 5-HappySPEAK See Event Details Functional Requirements

Feature 6-Subscribe to SPEAK Events

Description

Allows the user to say that he is going to that particular event.

Action and reaction

1. The user clicks the subscribe button.
2. The user is presented with a success/failure message.

Functional Requirement

Requirement ID	Description
11	The button only changes color if the operation is successful in the SPEAK webservice.
12	If the subscription request is successful, the button must change its color to gray.

Table 6-HappySPEAK Event Subscription Functional Requirements

Feature 8-Unsubscribe to SPEAK Events

Description

If the user no longer wishes to attend the event he can unsubscribe from it.

Action and reaction

1. The user clicks the unsubscribe button.
2. The user is presented with a success/failure message.

Functional Requirement

Requirement ID	Description
13	The button only changes style if the operation is successful in the SPEAK webservice.
14	If the subscription request is successful, the button must change its color to green.

Table 7-HappySPEAK Event Unsubscription Functional Requirements

Feature 9-Collect user data

Description

The system will collect data from the call/message logs to infer isolation.

Action and reaction

1. The system triggers a data collection event.
2. The data is collected.
3. Neural network infers if the user is isolated.
4. If he is, SPEAK events are suggested to him.

Functional Requirement

Requirement ID	Description
15	The sensors must only be triggered every 30 minutes.
16	The data must be normalized before being sent to the neural network: • Divide number of received calls by number of calls made. • Divide number of received SMS by number of messages sent.

Table 8-HappySPEAK Collect Data Functional Requirements

Feature 10-User Feedback

Description

When the system asks the user how lonely he feels, the feedback values are used to train the network and sent to the server if possible.

Action and reaction

1. The system asks the user for feedback.
2. The user responds.
3. The network is trained and the data is sent to the server.

Functional Requirement

Requirement ID	Description
17	The amount of times the app asks for feedback must be smaller the longer the user uses the app.

Table 9-HappySPEAK User Feedback Functional Requirements

Feature 11-Give suggestion

Description

If the system detects isolation, it shall present the user with a message showing him SPEAK events and places he can go to.

Action and reaction

1. The system detects the user feels lonely.
2. The system shows him SPEAK events and places to visit.

Non-Functional Requirements

Performance

As we discussed earlier, performance is a big problem in some devices, so the following guidelines must be taken into consideration:

1. Blocking operations such as requests sent to the server must be done on threads and not on the UI Thread of the app.
2. The Event list must use the ViewHolder pattern to save memory.
3. Bitmaps must be released as soon as they are not needed.

Usability

As requested by Hugo, the User Interface must:

1. Provide a uniform look and feel between all activities.

2. Activities must use a toolbar to display global operations available.

Accessibility

Since the user base for this app is made of many cultural backgrounds and languages:

1. The system must provide multi language support.
2. The system must support right-to-left text (for Middle Eastern people).

Privacy

In the HappySPEAK server all users must be identified by a randomly generated UUID, so that no personal data is usable if the system is compromised. This UUID will be associated to a user until he wipes the app data or changes to a new smartphone. The generated UUID must be stored in the smartphone's private space so that the information is not available to other apps.

3.2.2- WeDoCare

In this section we will expose the requirements we had to take into account while developing the WeDoCare app.

Context

WeDoCare is a system built upon the Happy System that aims to prevent violent attacks on refugees. It takes advantage of the mechanisms built in HappyWalk for detecting human emotion and applies it to this kind of situations, trying to help people by warning nearby policemen and common citizens that someone is in danger.

The app periodically collects data and uses it to check if the user is in a danger situation. The data comes from three sensors: the accelerometer, GPS and microphone, and is able to work with Wi-Fi, mobile networks and Wi-Fi beacons.

Product Scope

According to the United Nations High Commissioner for Refugees(UNHCR) web site there were 19.5 million refugees at the end of 2014, 14.4 of which were under the mandate of UNHCR. Most of the time these were from developing countries and tried to find shelter within the borders of their countries or neighboring countries. [9]

At least half of these refugees are adult women and children. Having to live without the protection of their local government, friends and traditional family structures, they often encounter situations when they are very vulnerable. [10]

To extend UNHCR's work, we have developed this app to help automating attack identification and notify nearby people/law enforcers to stop them as soon as possible.

Stakeholders

Stakeholders are the people that interact with the app in a direct and indirect way. For this app we defined the following stakeholders:

Stakeholders	**Who are they**	**What do they do?**
WeDoCare development team	Jorge Sá Silva David Nunes Ricardo Barbosa Ashley Figueira André Reis	Jorge and David are the project mentors. David also implemented the attack classification module. Ashley and André Worked on the server-side of the app. Ricardo implemented the notification system.
UNHCR	United Nations Division that deals with refugee situations.	Organization that will give us the opportunity to test the app on the field and will help us define the requirements needed for this system.

WeDoCare Users	Mostly people who were forced out of the place they lived in and are currently living in refugee camps.	People in refugee camps that want to have extra security.

Table 10-WeDoCare Stakeholders

Product Features

For this app we have identified the following features:

- Attack identification.
- User Registration
- Show heat maps that represent danger zones.
- Alert authorities
- Alert nearby people

User Profiling

The target users of this app are people in refugee camps or on their way to Europe who are vulnerable to violent attacks.

Operation Environment

The app will be developed for the Android Operating system and must be compatible with versions 2.2 to 5.1.1. Compatibility with version 6.0 of Android is limited, due to new way it handles permissions like using location, etc. It must also take into account the instability of Internet connectivity, due to refugees being constantly moving from one place to another. The smartphone must also be equipped with the necessary sensors to be able to work properly.

These restrictions were taken into account during the development of the app.

Design and Implementation Restrictions

The programming language is restricted to Java and the app must implement the necessary mechanisms to switch between different data connections when necessary (Mobile networks if Wi-FI is not available)).

Like the HappySPEAK app, it must also take into account that users may not have the required Google Play Services version and implement the mechanisms to ensure that the app runs without errors.

Dependencies

Like the HappySpeak app, the features currently assigned to this project can change at any moment. During the weekly group meetings, it can be decided to add, remove or refine features to this app.

Another dependency is the use of the Google Play Services library. At this time this library is in version 8.6 and the minimum version number required for the app to work correctly is 6.5. Users can choose to not upgrade their Google Play Services so mechanisms must be developed to allow the app to function normally if their current version is not in this spectrum.

User Interfaces

Because user interaction is very limited, only two screens were deemed necessary:

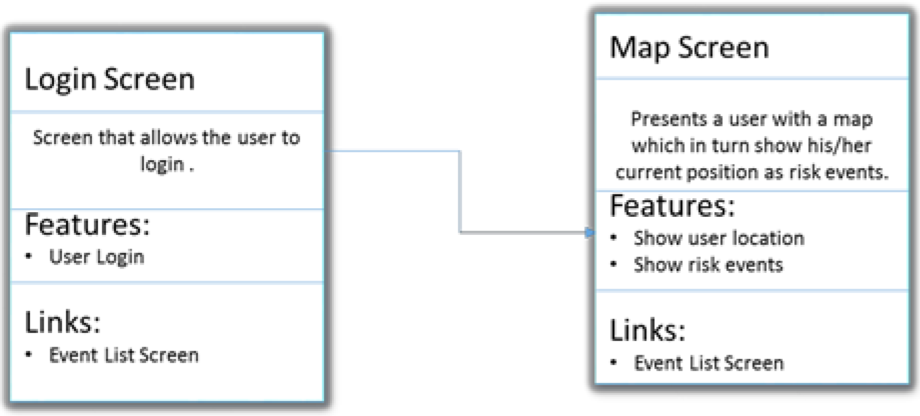


Figure 22-WeDoCare User Environment Design

User Interfaces

For this we did not need to define user interfaces, only adapt existing ones:

- The login screen was modified from the HappySPEAK login screen
- The map screen was modified from the HappyWalk map screen.

Software Interfaces

Here are the software interfaces the system must have:

- The app shall communicate with the WeDoCare server to store the user's current position and to receive danger zone locations.

The following image illustrates the app's working flow:

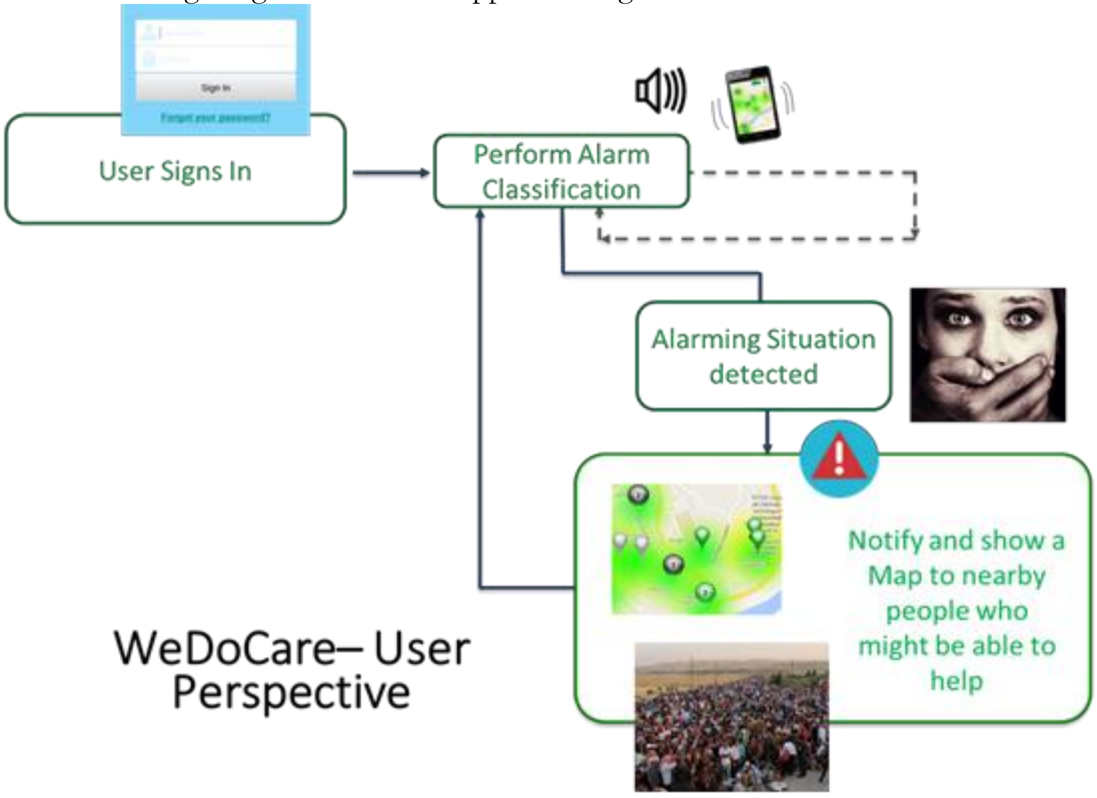

Sign in
Forgot your password?
User Signs In
Perform Alarm Classification
Alarming Situation detected
Notify and show a Map to nearby people who might be able to help
WeDoCare– User Perspective

Figure 23-WeDoCare User perspective

Communication Interfaces

The app will use REST to communicate with the WeDoCare server.

Feature Description

In this section of the report we will present the features for the WeDoCare system and give a more in-depth look on how they should work.

Feature 1-User Registration

Description

Users are required to insert a username and their phone number to register in the system.

Action and reaction

1. The user inserts the username and phone number.
2. The user clicks the "submit" button.
3. The user is presented with a map with heat maps.

Feature 2-Show danger zones.

Description

When users register, the app must redirect them to a map that displays danger zones around them.

Action and reaction

1. The user submits its data.
2. The app is redirected to the map activity.
3. The user is presented with a map showing the map with heat maps.

Feature 2-Attack detection

Description

When an attack is detected, a message must be sent to the server and redirected to the people around the user.

Action and reaction

1. An attack is detected.
2. The app sends a message to the server with the position of the user.
3. The app sends a distress signal to the people around the user via Wi-FI beacon, Bluetooth, etc.

Non-Functional Requirements

Performance

The app shall only use the sensors every 2 minutes to conserve battery.

Reliability

The app must have: efficient attack detection and reliable communication (use Wi-FI and cellular network)

Chapter 4

Development

In this chapter of the report we will discuss how we implemented the HappySPEAK and WeDoCare systems, justifying the choices made. The apps developed are prototypes and still require testing before being deemed good enough to be in production.

4.1- HappySPEAK

4.1.1-Task Assignment

For the first semester, work was divided between Ashley and Ricardo to maximize development speed.

Ashley was responsible for researching papers about emotion inference, so that we could know what sensors to use for the neural network input, and also did most of the work on modifying the HappyWalk server, seeing as he had previous experience with the Hibernate framework.

Ricardo was responsible for most of the Android development part, helping Ashley integrating his work in the Android app and testing the app, server and webservice.

In the second semester Ricardo was in charge of setting up battery and ANN accuracy tests. To do this, changes were required in both the Android app and HappySPEAK server, such as logging battery consumption.

4.1.2-Android app

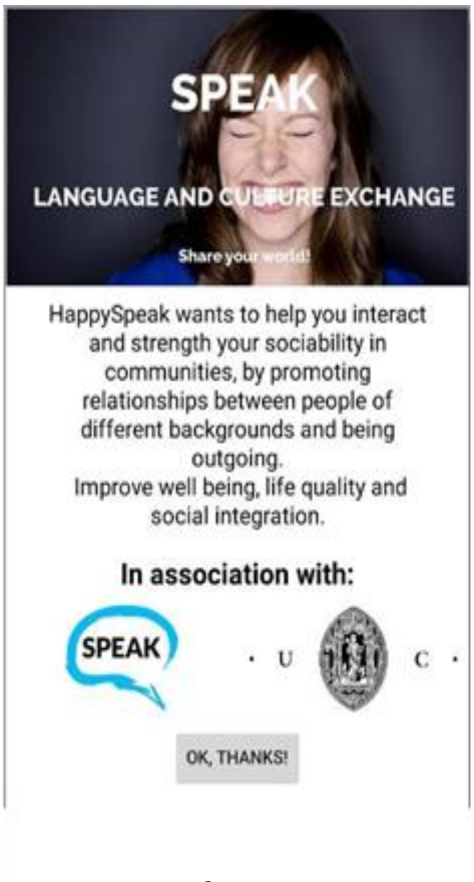
As discussed before, we started this project by expanding the HappyWalk code. In order to separate the SPEAK part of the project from the rest of the code, a new package was created to contain all the HappySPEAK user interfaces, webservice requests, and information collection.

User Interfaces

The Android architecture allows developers to have a sort of MVC by default. The User interface is built using a drag-and-drop tool which translates the User interface elements as XML.

The User Interface logic is declared in a class that extends the Activity class.

Information Screen


SPEAK

LANGUAGE AND CULTURE EXCHANGE

Share your world!

HappySpeak wants to help you interact and strength your sociability in communities, by promoting relationships between people of different backgrounds and being outgoing.
Improve well being, life quality and social integration.

In association with:

Screen

This is the first screen of the app. It informs the user about what its goals are and who are the entities involved in the project.

To proceed, all the user has to do is click the bottom button to get to the login screen.

The corresponding files are:

XML	Class
Activity_main	Main_Activity

Table 11- Information Screen Resources

This activity is used to start up the app's Service class which we will talk about further in the report.

Login Screen

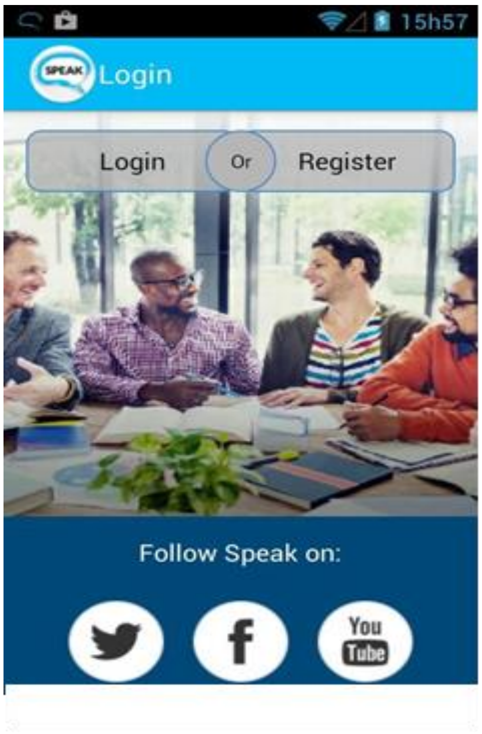


Figure 25-HappySPEAK Login Screen

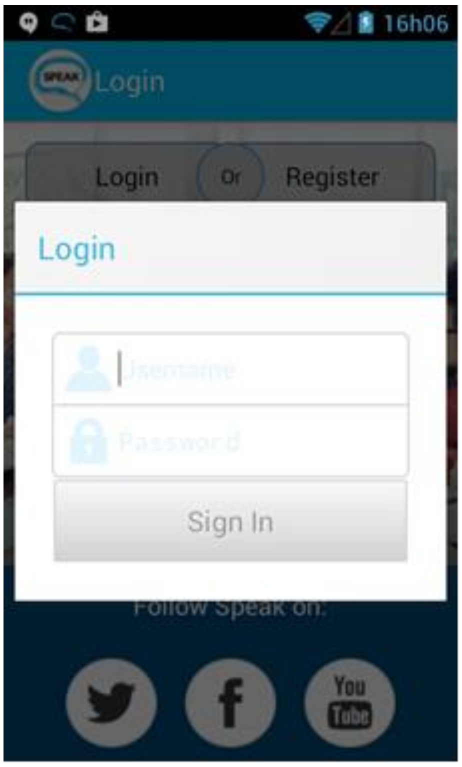
This activity allows the user to login, register or see SPEAK's social network websites.

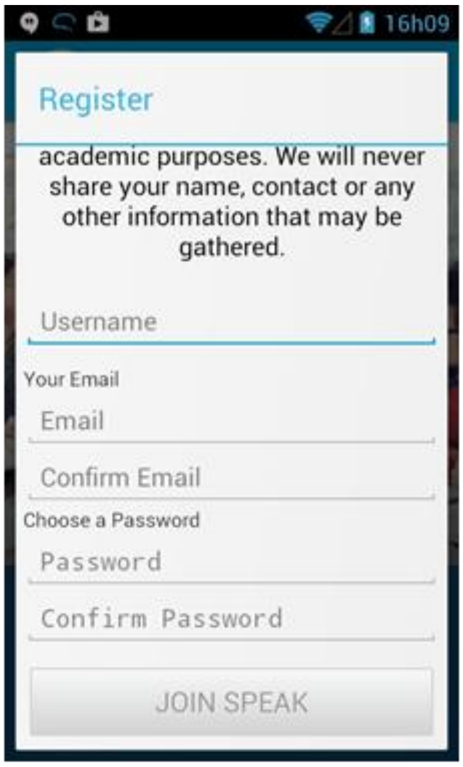
The Files used by this activity are:

XML	Class
Activity_login	LoginActivity
Dialog_ login	DialogLogin
Dialog_register	DialogRegister

Table 12-Login Screen Resources

Due to the limited functionality of the login and register features, it was not deemed necessary to build standalone activities for them. Instead we used dialog boxes (pop-up boxes) as we can see in the images below:





academic purposes. We will never share your name, contact or any other information that may be gathered.

Figure 26-Happy SPEAK Login/Register Dialogs

The Login Dialog allows the user access to his account and is necessary to see SPEAK events. The Register Dialog allows the user to create a new SPEAK account.

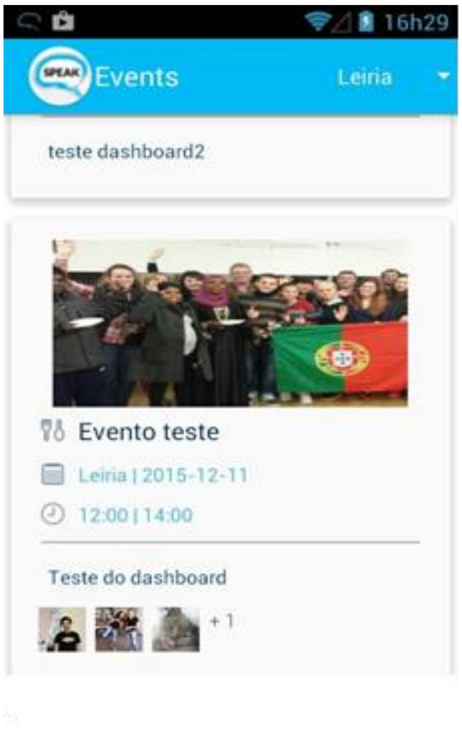


Figure 27-HappySPEAK Event List Screen

Event List Screen

This activity allows the user to see all future and past events for a specific city. If the user changes the city using the widget in the toolbar, the list will refresh and display only the events of that city.

XML	Activity
Activity_show_events	EventsActivity

Table 13-Event Screen Resources

To be able to display a custom list like this we had to create a custom ListAdapter. ListAdapters are used to draw and organize List Entries, but, to achieve a layout similar to the website, a custom one had to be created using the following resources:

Resource	Description
Listitem_event	Xml file that defines the list entry layout
CustomListAdapter	Class that defines how the list item is drawn and fills the fields with the event information

Table 14-Custom adapter resources

Event Detail Screen

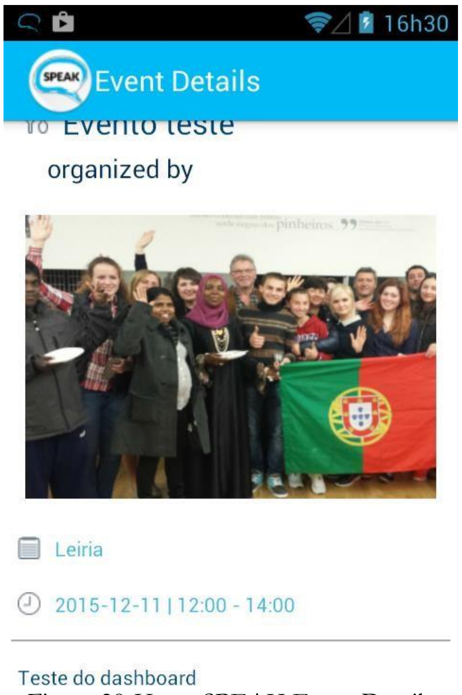


Figure 28-HappySPEAK Event Details Screen

XML	Activity
Activity_event_detail	EventDetailActivity

Table 15-Event Details Screen Resources

This activity allows the user to see more information about the event, such as its location on a map, who organized it, a more extensive description and the ability to subscribe/unsubscribe to the event.

Traditionally, to display a place on a Google Map we require its latitude and longitude. Unfortunately, no such information existed in the SPEAK Database. By using geocoding, we were able to use the events address to get its geographical coordinates. Android already possesses the necessary mechanisms to do this, by communicating with Google servers.

Webservice Client

After the SPEAK webservice was created we checked the wsdl file to get the information need to access the SOAP methods:

- SoapAction
- MethodName
- Webservice Address
- Namespace

And created the following SOAP requests:

Class Name	Description
GetDelegations	Get all the existing cities
GetEventsbyDelegation	Get all events corresponding to a city
GetOrganizers	Gets the organizers of a specific event
GetWhoGoing	Gets the ids of the people that are going to a specific event.

LoginTasker	Sends the user's credentials and waits for confirmation.
RegisterTasker	Tries to register a user.
SubAndUnsebTask	Tries to subscribe/Unsubscribe to an event.
GetClassAttenadance	Gets information about how many SPEAK classes the user has attended/missed.

Table 16-HappySPEAK Android Request

HappySPEAK Service

An Android Service is a component that can perform long-running operations in the background. They are usually used to perform operations that do not require any user interaction. Services also run in a higher priority than inactive activities so it is less likely that the OS terminates them.

We used this component to collect the sensor data from time to time and send it to the HappySPEAK server. Basically what it did was schedule a data collection task that, when finished, send the data to the server (if possible).

This Service was also used to launch an Activity that asks the user for feedback on how they are feeling:

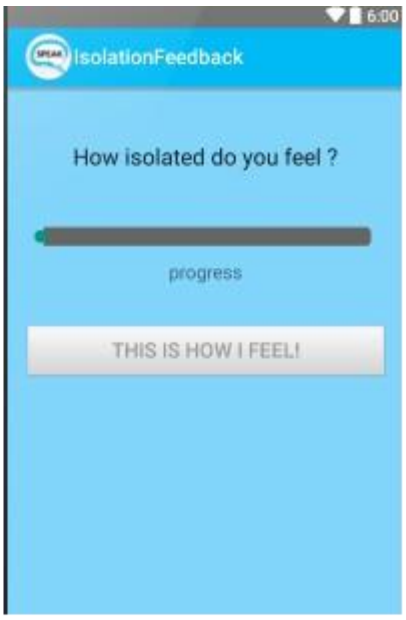


Figure 29-HappySPEAK User Feedback Menu

In order to avoid influencing the user in giving results that don't reflect reality, we only show what the neural network inferred after the user has submitted his feedback.

Neural Network

The neural net that is currently implemented is a Feedforward neural network with a sigmoid function.

After the data is collected it undergoes the following transformations:

The number of calls/messages that were made/received are used to calculate a ratio to see if the user sends more messages than he receives.

If the network detects that the user feels lonely, it triggers the map activity which displays suggestions of SPEAK events occurring soon and places that can help him meet new people. These suggestions will help the users to socialize; thus, reducing his isolation.

4.1.3- WebService

This webservice was created using the OutSystems platform. Originally we planned on using a REST webservice but unfortunately at the time, the platform's version did not support such kind of webservice, only SOAP.

Since Android does not support SOAP natively, we had to use a library to send/receive requests from the webservice. We ended up using **Ksoap2**, a mature and much recommended library used by many Android developers.

Before building the webservice we knew the types of information needed:

- User.
- Event City.
- Event Information.

- Event Participants.

And so the following webservice methods were created:

Method Name	Description
GetDelegations	A Delegation represents a city. This method returns all the delegation names and Id's.
GetEventsByDelegationID	Gets events for only that city. Returns an array with all of the event information.
GetOrganizer	Gets who the organizer(s) is for a particular event and returns the username.
GetWhosGoing	Gets the ids of who is going and returns them. These ids' will be used to get their SPEAK/Facebook photos.
Login	Checks if the user exists in the SPEAK database. It receives the username and password and returns a Boolean confirming the user identity.
RegisterUser	Receives the Name, Email and password of the new user and checks if he exists or not. It returns a Boolean to confirm if the user was created or not.
SusbcribeAndUnsubscribe	Receives the user Id and Event Id and checks his subscription status. If no subscription exists he gets subscribed. If he is already subscribed, then he gets unsubscribed.
GetClassAttendance	Receives the user Id and retrieves how many SPEAK classes he attended or missed.

Table 17-HappySPEAK Webservice methods

4.1.4-HappySPEAK Server

The HappySPEAK server is built upon the HappyWalk Server with the following changes:

- Added new database tables to store new information. The resulting database is represented by figure 30:

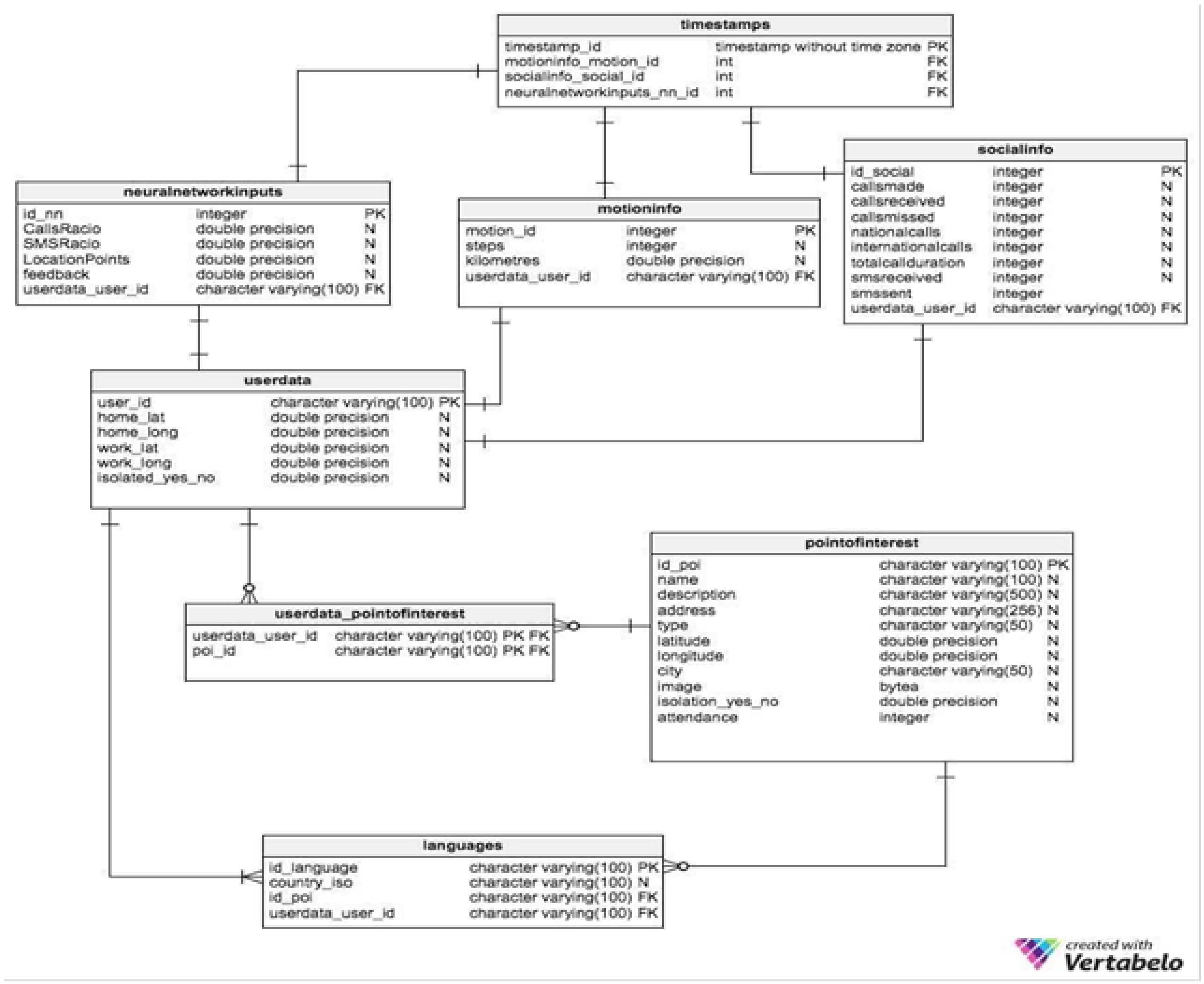

timestamps
timestamp_id timestamp without time zone PK
motioninfo_motion_id int FK
socialinfo_social_id int FK
neuralnetworkinputs_nn_id int FK
neuralnetworkinputs
id_nn integer PK
CallsRacio double precision N
SMSRacio double precision N
LocationPoints double precision N
feedback double precision N
userdata_user_id character varying(100) FK
motioninfo
motion_id integer PK
steps integer N
kilometres double precision N
userdata_user_id character varying(100) FK
socialinfo
id_social integer PK
callsmade integer N
callsreceived integer N
callsmissed integer N
nationalcalls integer N
internationalcalls integer N
totalcallduration integer N
smsreceived integer N
smssent integer
userdata_user_id character varying(100) FK
userdata
user_id character varying(100) PK
home_lat double precision N
home_long double precision N
work_lat double precision N
work_long double precision N
isolated_yes_no double precision N
pointofinterest
id_poi character varying(100) PK
name character varying(100) N
description character varying(500) N
address character varying(256) N
type character varying(50) N
latitude double precision N
longitude double precision N
city character varying(50) N
image bytea N
isolation_yes_no double precision N
attendance integer N
userdata_pointofinterest
userdata_user_id character varying(100) PK FK
poi_id character varying(100) PK FK
languages
id_language character varying(100) PK
country_iso character varying(100) N
id_poi character varying(100) FK
userdata_user_id character varying(100) FK
created with Vertabelo

Figure 30- HappySPEAK Database Diagram

- We created container classes so Hibernate could transform Java objects into database records. These classes correspond to the neural network inputs, motion information and social data.
- New REST methods were created to accommodate incoming requests populated with this new type of information.

Table	**Description**
Neuralnetworkinputs	Contains the inputs used on the neural network.
motioninfo	Contains the information collected by the accelerometer.
Socialinfo	Contains the information collected from the call and message logs.
Timestamps	Links all of the data tables by grouping their timestamps.
userdata	Contains the user ID and location of his home.

languages	Contains the user language and links the user to the point of interest table

Table 18-Database Tables Description

4.1.5-HappySPEAK Test Setup

This sub-chapter describes part of the work done on the second semester. This includes how the tests were setup and how we collected the results.

The tests were developed to answer two questions:

- Is the app battery intensive?
- Is the ANN accurate?

To do this, two tests suites were created: performance tests and ANN accuracy tests. In order to analyze battery consumption, we were required to add some extra classes to the android and server. To get performance statistics we created a new request:

Fields	**Description**
Timestamp	Time the feedback request started.
Battery_level_before	Battery level before the feedback request began.
Battery_level_after	Battery level after the feedback request finished.
CPU_level	Average CPU usage for the feedback request.
beta	Feedback request frequency.

Table 19-Performance request description

The CPU_level field was meant to represent CPU usage by the feedback requests. However, Android does not possess native methods that allow us to get this result. In order to get these metrics, we required a set of classes that would access Linux-specific files and monitor the process in which the app was running on. This turned out to be very unreliable, because the results obtained with this method differed greatly from the one being showed by Android Studio's CPU Monitor. Due to this factor, we decide to analyze CPU usage by analyzing the CPU Monitor.

For the ANN accuracy tests we already had built the necessary methods to send and save information in the server.

The results we obtained from these tests are described and analyzed in chapter 5.

4.2- WeDoCare

4.2.1-Task Assignment

For this project we had to divide the tasks among more people because of a tight deadline. We were given two weeks to build a working prototype in order to showcase it to an UNHCR representative.

David was tasked with building the attack classification module, Ashley and André developed the server and Ricardo changed some Android Code, namely the notification system, user interfaces and activity flow. In the next section, we will only specify the work done on the Android part of the project.

4.2.2-Android app

User Interfaces

Login Screen

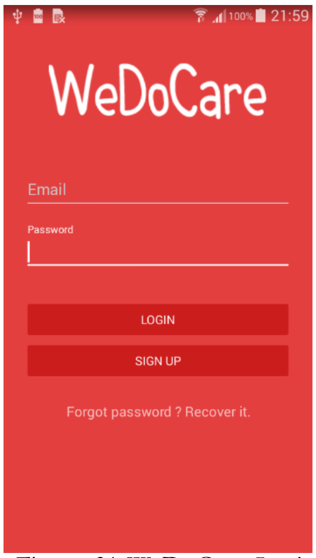


Figure 31-WeDoCare Login Screen

As stated before, the login menu was an adapted version of the HappySPEAK login menu. All that was changed was the logo and some of the text

Map Screen

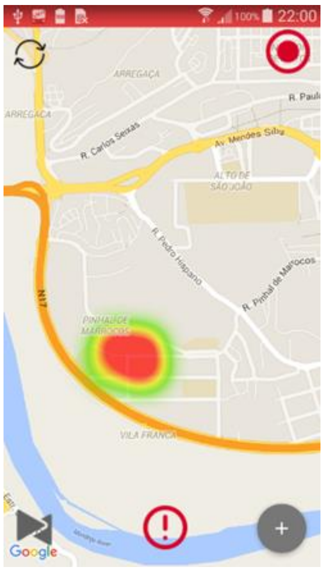


Figure 32-WeDoCare Map Screen

The map screen is the same as the one in HappySPEAK. A red marker represents the current location of the user and the other one represent ongoing attacks in the vicinity. The map contains four buttons. (described from the top left to bottom right):

- Refresh button: refreshes the Points of Interest in the map.
- Toggle Danger Zones: Shows or hides danger zone heat maps.
- Activate Communications: Activates the hardware required to communicate with other devices.
- Emergency button: Notifies all people around the user about the attack.
- Add POI- Adds a point of interest to the map.

WeDoCare Service

The WeDoCare service runs a thread every two minutes and collects data from the accelerometer, GPS and microphone. After, it collects the data and parses it, sends the results to the attack classification module, where it decides if the user is in a dangerous situation or not.

Attack classification

The attack classifier for the WeDoCare app takes as input data from the microphone and accelerometer. Before being used as input, the data has to undergo some processing.

The sound waves captured by the microphone are averaged and the data from the accelerometer undergoes a Fast Fourier Transformation.

If these values are above a certain threshold, an alarm is sent to the server.

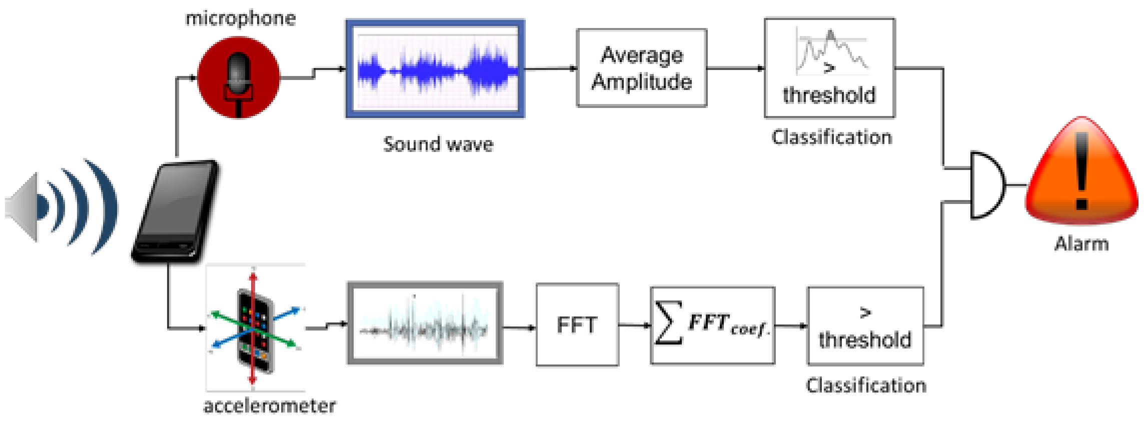


Figure 33-WeDoCare attack classification

We present below a simplified diagram of how the WeDoCare system works:

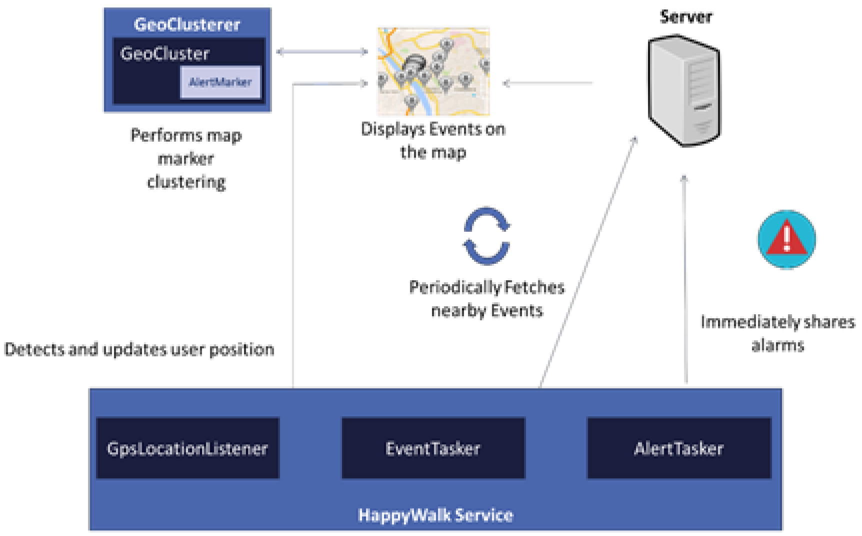

GeoClusterer
GeoCluster
AlertMarker
Performs map marker clustering
Server
Displays Events on the map
Periodically Fetches nearby Events
Immediately shares alarms
Detects and updates user position
GpsLocationListener
EventTasker
AlertTasker
HappyWalk Service

Figure 34-WeDoCare app behavior

During the development of this app, a short video was made by Ashley and André to expose what WeDoCare is capable of. This video is available at:

https://www.youtube.com/watch?v=x7bucCdWoGQ

As stated before, all the work described in this sub-chapter was developed in the first semester. In the second semester, all responsibilities on this project were shifted to Ashley, who built upon this working prototype and improved UI, attack classification, device connections and more

.

Chapter 5

Tests and Results

This chapter reflects all the work done in the second semester.

Every new generation of smartphones comes with more powerful processing capabilities, added functionalities and new Operating Systems. Usually, a trade-off of this evolution is the bigger power consumption. As a result, it is important to manage the apps background processing correctly in order to improve battery lifetime.

There is a delicate balance between battery lifetime and the amount of feedback requests presented to the user; if we ask the user for feedback in a short time interval, the ANN will increase its accuracy faster, but it will also drain the device's battery at a higher rate. On another hand, if we ask the user for feedback too often he can get annoyed and may start giving inaccurate feedback.

In order to analyze these restrictions several tests were designed for HappyWalk and HappySPEAK. We decided to start out testing with the HappyWalk app because it is a more "mature" app with a lower probability of having undiscovered bugs. These tests were divided in two categories: Battery Life tests and Neural Network Accuracy tests.

Battery Life tests monitor how the battery life progresses over time and the percentage of CPU used by the app when a new feedback request is launched. Neural Network Accuracy tests were meant to see how the neural network accuracy progressed over time.

5.1-Battery Life Tests

The testing device we used was a Huawei ascend Y 201 Pro, which comes with a Cortex A-5 single core Processor (clocking at 800 MHz) with Android 4.0.3. Its approximate battery life is approximately **130 hours** in full standby mode. We will be using this value as a reference throughout the analysis of our performance test results. It is important to note that, for this test suite, we stopped all background services and apps to ensure that the app being tested and Google Play Services were the only elements running in the smartphone.

The operational environment of the tests was also taken into account: all the performance tests were executed in the same place and using the same WI-FI access point, to avoid outliers in the results due to inconsistent Wi-Fi and GPS connections.

A feedback request consists of three steps: data collection, user feedback and network training. Data collection consists of getting the data from the sensors described in the previous section and inferring the user's mood. After that, a notification appears asking the user for feedback. HappyWalk's feedback menu is a chart with two axis: the horizontal axis represents calmness and anxiety and the vertical axes represent euphoria and boredom (*recall Figure 9-HappyWalk Feedback Request[7]*). HappySPEAK's feedback menu has a slider that goes from 0 to 100. We considered 0 to 49 as the user feeling good, and 50-100 as the user feeling isolated. (*recall Figure 29-HappySPEAK User Feedback Menu*)

To measure battery consumption, we created two test types: one used GPS and Wi-Fi to discover the users' location and the other only used Wi-FI. We also considered four different time intervals to request user feedback: every one, two, three and four hours. From now on, we will refer to this time interval as β.

Instead of doing the tests manually, we added some code to automate this particular test suite: when a feedback request is received, the screen turns on, random values picked as feedback and the submit button is clicked automatically. Methods were also added to retrieve the current level of the battery before and after the feedback request and send it to a server for statistical analysis.

Some complications arose in this part of the internship; the usage of GPS maintains a wake lock (a wake lock prevents the smartphone from entering a sleep state) on the smartphone's CPU in order to receive location updates. Wi-Fi location does not use this mechanism, and as such, the smartphone would enter a "sleep" state and never do the tasks required for power consumption analysis in the required time interval.

These complications were unknown to us at the time, and due to the nature of these tests, the debugging process was very slow. For each value of β we were required to wait long periods of time to check if the app would "wake up" and launch the feedback request. This debugging process continued for several weeks until the problem was identified and fixed.

This problem was solved by using a partial wake lock on the service, keeping the CPU awake so that the service could wake up when it should and collect data about the battery. A trade-off of this was a larger power consumption.

Unsatisfied with this trade-off, we decided to experiment with an Android mechanism called "Alarm Manager". This mechanism allows an app to run a block of code at a future time, even if the process that launched it is not active. This mechanism also uses a wake lock but only while it executes the registered block of code, releasing it as soon as the task is finished. This proved to be an excellent strategy, as it increased battery life by a large amount.

5.2-Neural Network Accuracy Tests

The neural network accuracy tests were designed to analyze the accuracy of the ANN over time. To do this, the apps were distributed to a group of people who used them to over a period of time. The HappyWalk app was distributed inside our research group, and the HappySPEAK app was distributed to a control group of 10 people.

These tests allowed us to evaluate how the ANN behaves over time and check how its accuracy progresses. The operational environment was different for every user, due to the enormous variety of Android devices available.

5.3-HappyWalk Test Results

In this section of the report we will present and describe the various results of the two test suites we executed for the HappyWalk app.

5.3.1- HappyWalk Performance Test Results

The tests in this section include two tests using GPS and/or Wi-Fi, and another with a mechanism called Alarm Manager, which we will discuss later.

For the first test, we used GPS and Wi-Fi for location discovery. We expected a high battery consumption, as GPS usage in mobile devices has a very large impact on battery life.

The results of the Performance test suite, shown in Figure 35, have a very approximate battery decay pattern for all β values, indicating that the user feedback process itself is not battery intensive, but using GPS and WI-FI at the same time for location discovery is, shortening battery life by 90%.

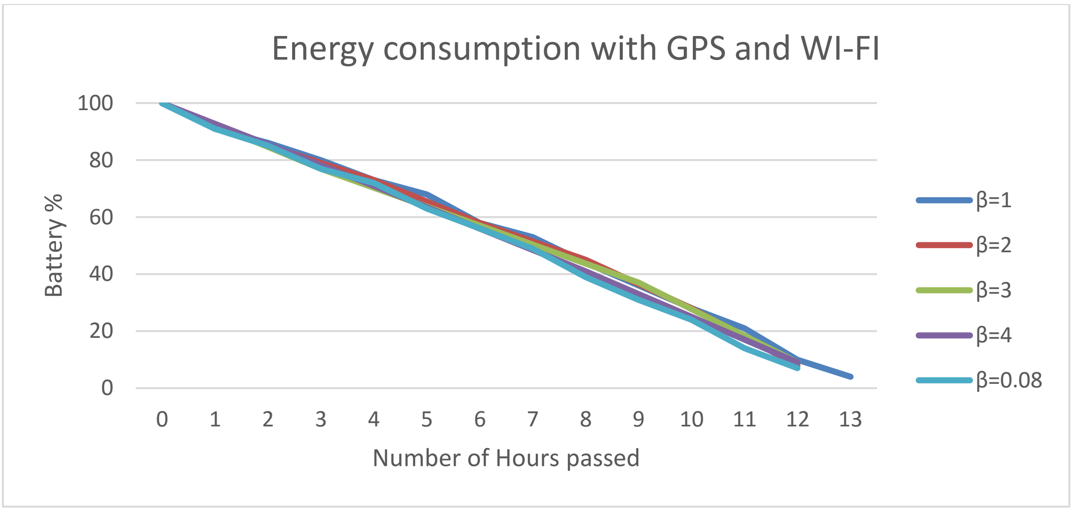


Figure 35-HappyWalk energy consumption with GPS+Wi-fi

To validate this hypothesis, an extra test was created with a β of 0.08 (equivalent to 5 minutes). The results validate the hypothesis, showing a very similar decay pattern. These results tell us that the frequency of feedback requests are not relevant to improve the smartphone's battery life, but the correct usage of GPS/WI-FI for location discovery is. This is further validated by the results of the second test suite:

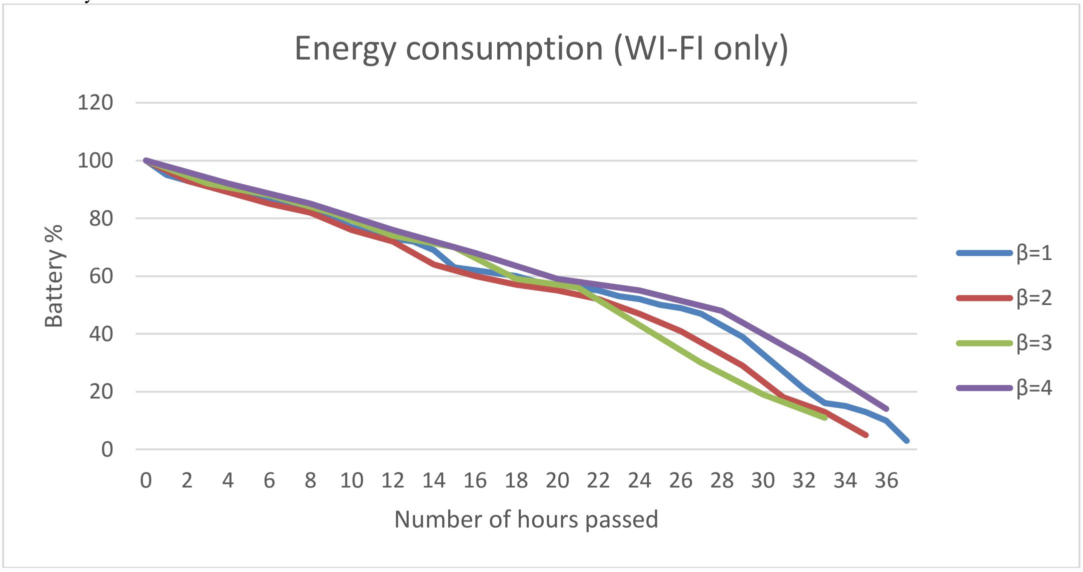


Figure 36-HappyWalk Energy Consumption with Wi-Fi

As shown in Figure 36, using only WI-FI to discover the user's location, battery live is decreased to 72.3% and for all the tests, battery decay follows approximately the same pattern.

After collecting and analyzing these results we can conclude that that β does not have a big impact on the battery lifetime of the smartphone. However, the power consumption used by the app was still unsatisfactory, due to the use of a wake lock.

To overcome this problem, we decided to use a mechanism called Alarm Manager. This class allowed the app to schedule a task and run it after a certain period of time, and helped improve battery life by launching at a designated interval. All of the task scheduling and launch are handled by the OS. Using this mechanism, we were able to have feedback request without the use of a persistent wake lock.

After we finished implementing this new strategy, we repeated the test (using only Wi-Fi) and got the following results:

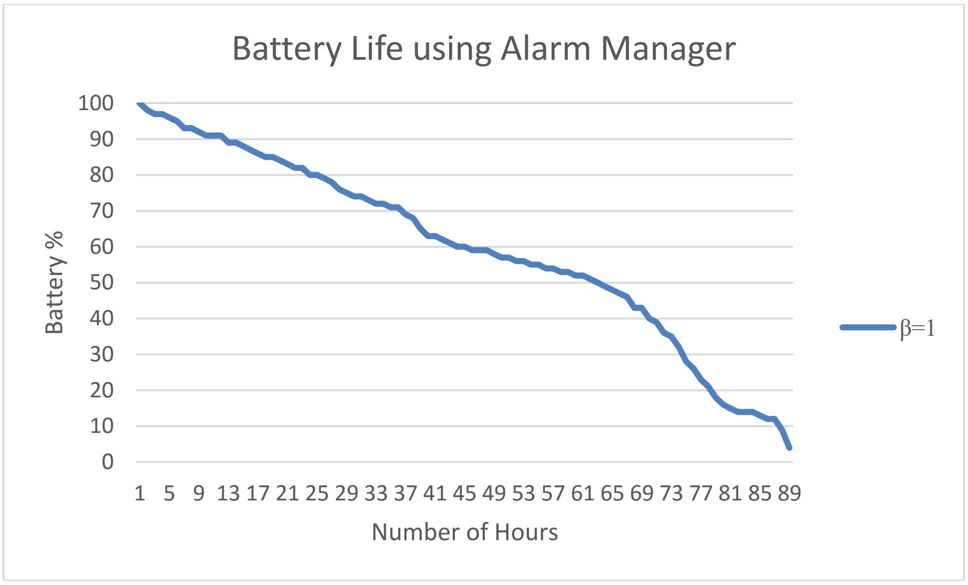


Figure 37-HappyWalk battery usage with Alarm Manager

As we can see there is a massive improvement on the devices' battery compared to the previous test, decreasing it only by 31.5%.

Seen as we have already established that that β does not have a significant impact on the battery life of the smartphone, we only executed one test for β =1.

In addition to the battery life tests we also ran several performance tests to check the CPU usage for different types of ANN configuration. These networks only differ from each other in the number of hidden layers and neurons. The different configuration tested were:

Network id	Number of inputs	Number of Neurons in Hidden Layer #1	Number of Neurons in Hidden Layer #2	Number of Neurons in Hidden Layer #3	Number of Outputs
#1	4	3	2	None	2
#2	4	5	5	None	2
#3	4	5	5	5	2

Table 20- Tested network configurations

To get the CPU percentage used for each stage of a feedback request, we used timestamps to get the time each task took to execute. These times where then compared against the CPU monitoring tool available in Android Studio. It is important to note that the results we are about to present were all collected using the same smartphone as the battery life tests. Due to the enormous amounts of Android devices that exist results will vary depending on the hardware.

The several stages of a feedback request we considered were:

- UI Building- An User Interface of the app is being built.
- Data collection- Sensors are obtaining data and normalizing it.
- Emotion Inference- ANN is inferring an emotion with the new data collected by the sensors.
- ANN training: ANN is trained with the feedback provided by the user.

After the tests we obtained the following results:

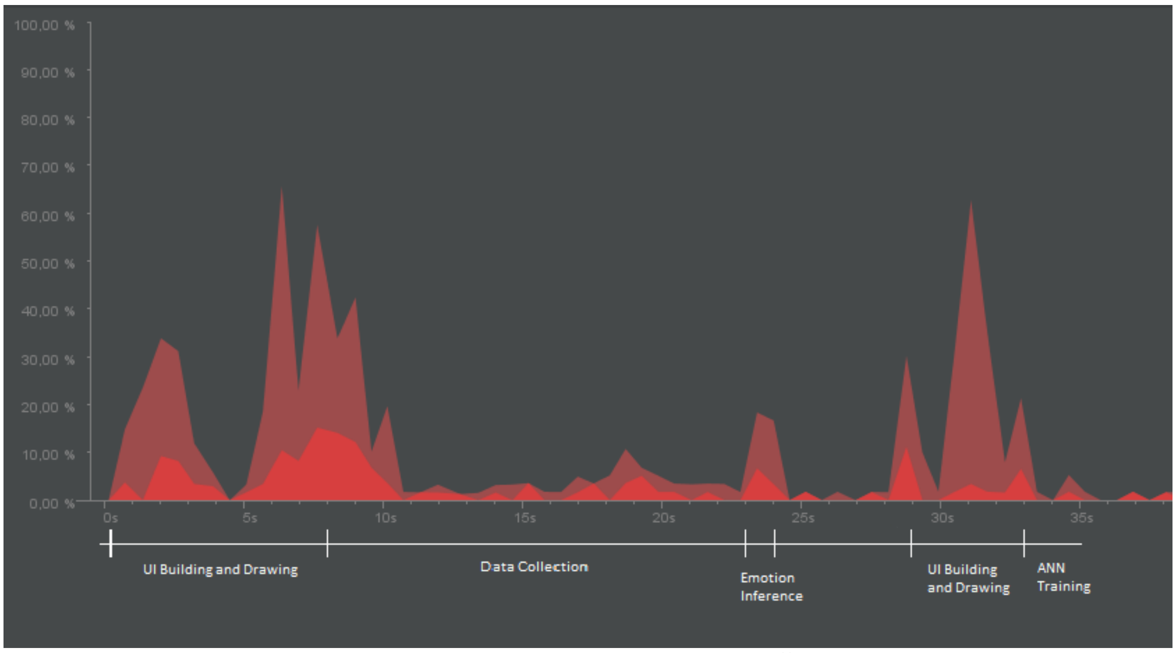


Figure 38-CPU usage for ANN #1

As we can see, most of the CPU allocation is used to build and draw the User Interface. The ANN uses about 20% to infer a new emotion from the collected data.

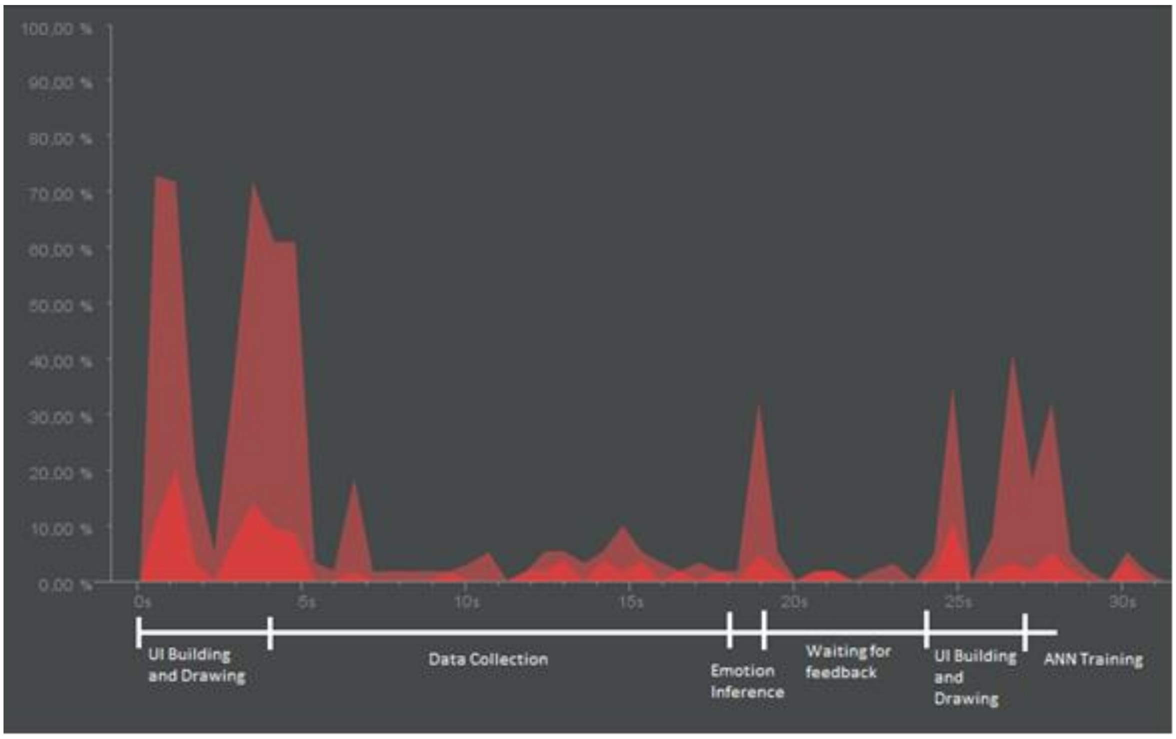


Figure 39-CPU usage for ANN #2

Results for this network configuration are very similar to the last one, with the exception of the Emotion Inference. Adding additional neurons to the second hidden layer appears to use approximately 30% of the CPU, 10% more than the last configuration.

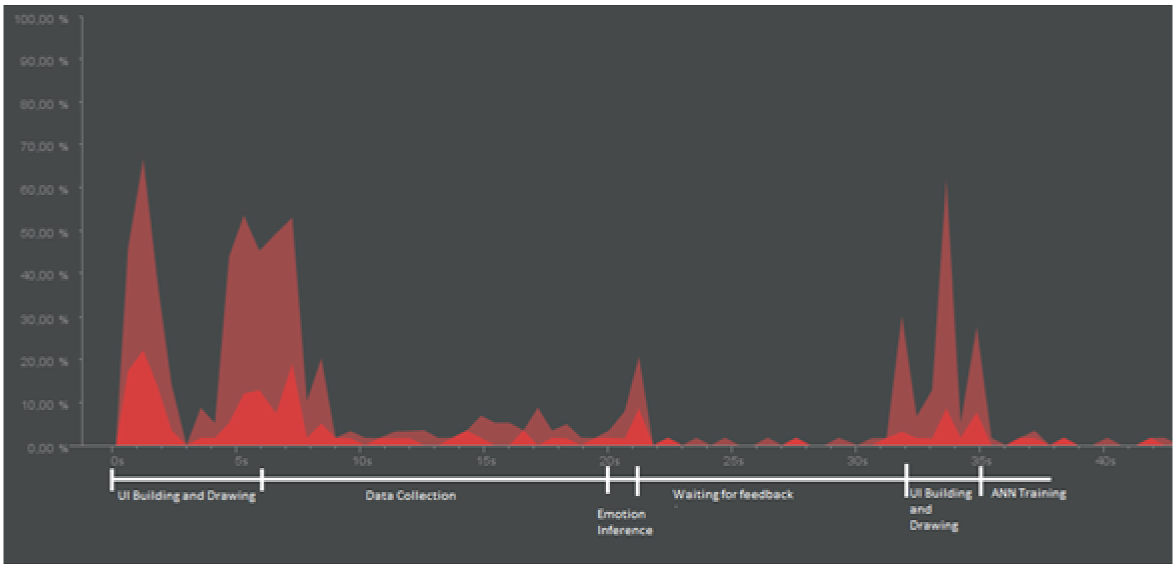


Figure 40-CPU usage for ANN #3

By adding another layer to the network, with 5 neurons, CPU usage drops down to 20% again. This tells us that the result obtained for configuration #2 is probably an outlier, because increasing the complexity of the network should also increase CPU usage. From this we can conclude that an increase of neurons for this system is not relevant in terms of CPU usage. This means we can focus on finding a network configuration that provides the best accuracy while simultaneously avoiding overfitting.

5.3.2-HappyWalk ANN accuracy Tests

The accuracy of our ANN is an important part of this project. To evaluate its evolution, we installed HappyWalk in the smartphones of several tests subjects, with different values of β, and asked them to use HappyWalk during a period of at least one day for each β value.

Each time the user provides feedback, the neural network is trained and the values of the feedback and the inferred result provided by the ANN are sent to and saved on a remote server.

This allowed us to calculate the accuracy of our artificial neural networks for different frequencies of feedback requests.

After the user gives his feedback, both the yellow and green circles(recall Figure 12) coordinates are converted into two values, one for each axis. We shall refer to these values as:

- oY and oX for the values inferred by the ANN.
- fY and fX for the feedback values provided by the user.

These values are then used to calculate the Euclidean distance between two points using the following formula:

$$\text{Accuracy}=\sqrt{(oY-fY)^2+(oX-fX)^2}$$

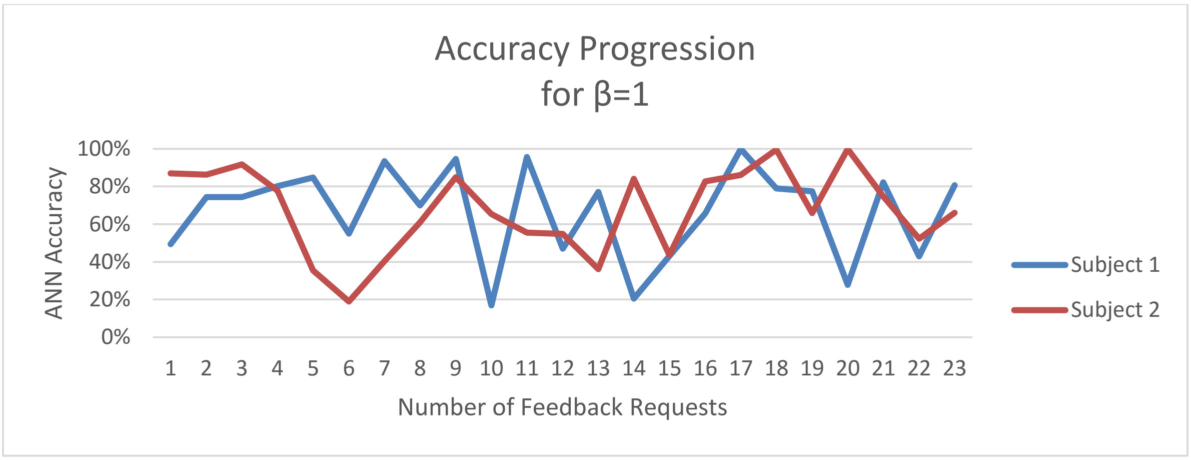


Figure 41-HappyWalk accuracy over time

Figure 41 represents the results for 24 hours of usage for two subjects. We can observe that the accuracy varies greatly along this time interval ranging from 20% to almost 100%. This is expected in the beginning, as the network is still in training. With these results we calculated the trend lines for the accuracy progression of both subjects. This can give us a better comprehension on how the accuracy evolved over time.

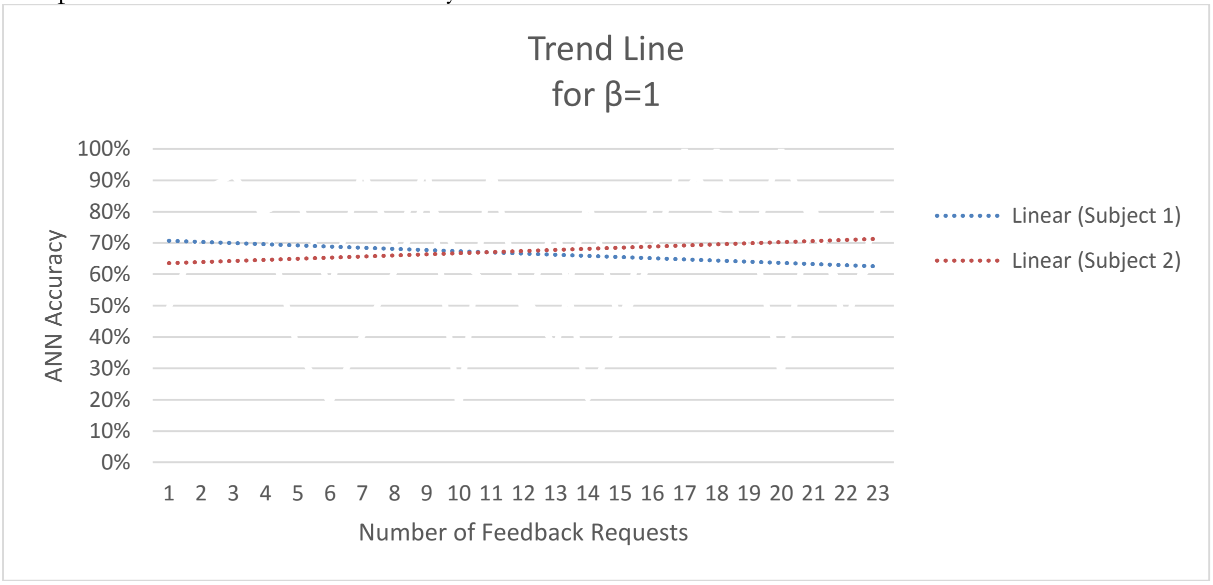


Figure 42-HappyWalk accuracy Trend Lines

The evolution of accuracy seems to be opposite between the two subjects. This difference can be explained by the fact that every person's personality and emotional states differ greatly, and that the network requires more time/data to learn how to correctly classify the four different emotions in our spectrum.

To confirm this hypothesis, we organized the feedback given by the two subjects in a scatter chart, so we could have a visual representation, and put this hypothesis to the test:

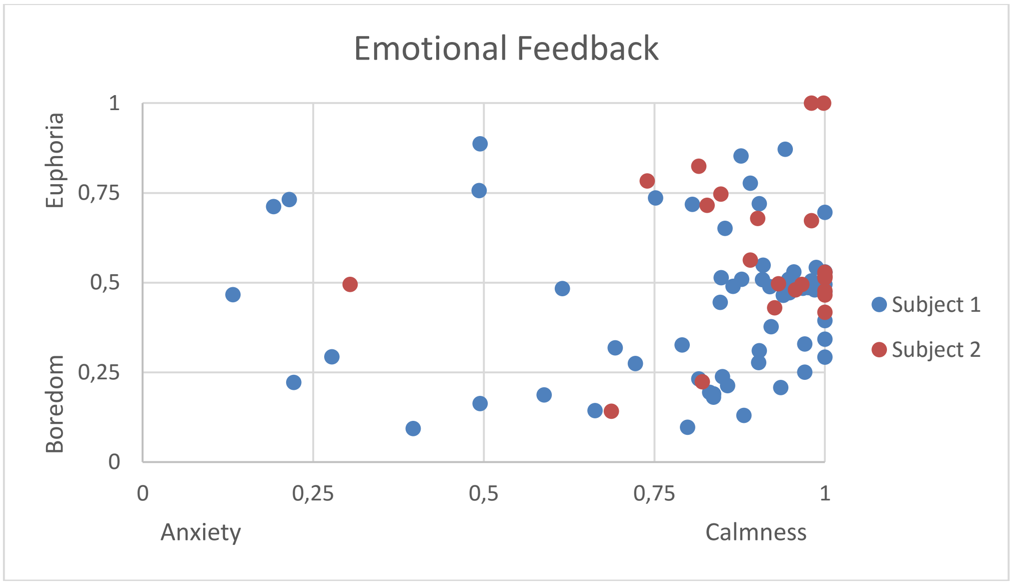


Figure 43-HappyWalk user feedback scatter chart

As we can see each subject tends to give feedback in a specific region, which makes it harder for the ANN to correctly infer a new emotion. The only way to overcome this problem is waiting for the user to experience all emotions in our spectrum and give the appropriate feedback. This may take weeks, if not more, depending on the lifestyle of the user. Due to limited time, this issue has to be analyzed at another time.

5.4-HappySPEAK Test Results

In this section we will present all tests related to the HappySPEAK system. This includes our battery and accuracy tests, which are very similar to the ones executed to the HappyWalk system. These tests were conducted under the same restrictions as the previous ones.

5.4.1- HappySPEAK Battery Test Results

After we finished the battery tests for the HappyWalk app, we executed similar ones for HappySPEAK. These came with a few changes, namely the use of Alarm Manager to increase battery life. These changes were applied due to the results of the HappyWalk performance tests; HappySPEAK's background processing architecture was heavily based on HappyWalk's, and after we discovered that Alarm Manager was more power efficient we applied it to this app as well.

The inputs of the ANN were also changed; the ANN no longer uses the location of the user as an input, extending the overall battery life as a result.

We also decreased the number of tests for this particular test suite. As we discussed earlier the value of β does not affect the overall battery life, but the correct management of sensor does. Seen as this app does not use any sensors to detect if the user feels lonely, we only executed one test for β =1. The results are presented in Figure 44:

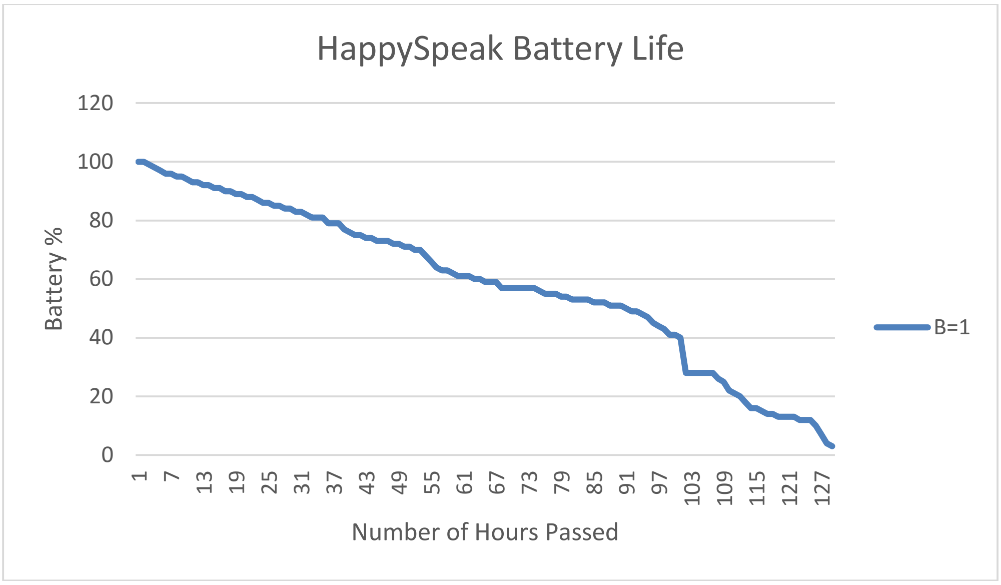


Figure 44-HappySPEAK battery life with Alarm Manager

The app was able to run for 127 hours, responding with automated feedback requests every hour. This is a very good result, seeing as the idle battery life of the phone is 130 hours, the battery percentage used is only 2,3%. This is a direct consequence of not using any sensors, and using only data retrieved from messaging and call logs.

5.4.2- HappySPEAK Accuracy Test Results

To analyze the accuracy of the developed ANN we distributed the app by 10 people over the period of one week. Feedback requests would be launched every hour asking the user how lonely he felt. The users answered this question by using the slider in the Feedback Menu. This slider contains values between 0 and 100. We considered values between 51 and 100 to represent loneliness and any other values to represent the opposite.

It is important to note that the users did not respond to every single feedback request; many of these were ignored. Also, many of the users did not have enough feedback responses to be considered for our results. As such we only considered users that had 10 or more feedback responses.

Accuracy was calculated using the following formula:

$$Accuracy = 1 - |O - F|$$

O represents the output of the ANN for a set of inputs and **F** represents the feedback provided by the user, for that same set. The results of these tests can be consulted in Figure 45:

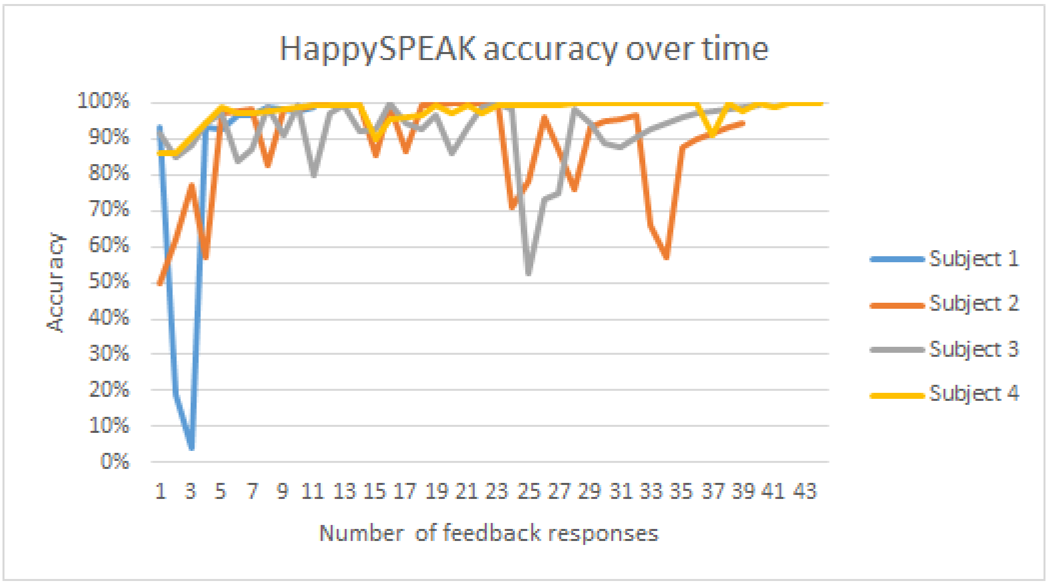


Figure 45-HappySPEAK accuracy test results

Accuracy seems to be very high for all subjects; in the next table we can see the average accuracy for all feedback requests:

Subject	**Average Accuracy**
Subject 1	81%
Subject 2	92%
Subject 3	92%
Subject 4	98%

Table 21-HappySPEAK average user accuracy

Even though the results are very good (the ANN shows an average 90%+ accuracy for most users), the results are a bit misleading. If we observe the next figure, we can see where most of the feedback from the users was allocated:

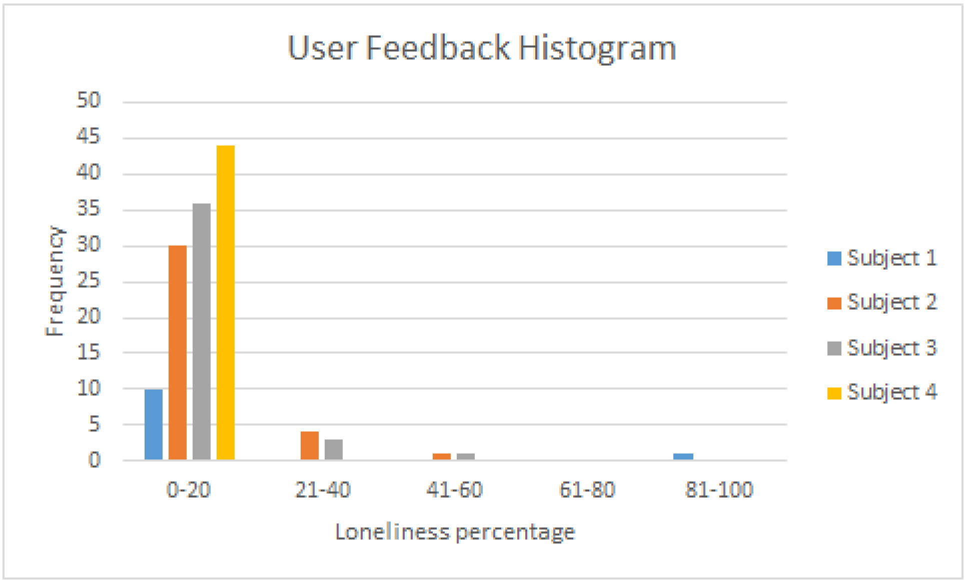


Figure 46- HappySPEAK user feedback histogram

The histogram above shows that almost all of the feedback requests were used to describe that the user does not feel lonely (between 0 and 50%).

After analyzing these results we come to same conclusion as the HappyWalk tests: the app needs users to give accurate feedback about how they are feeling in the full spectrum of our metrics. The only way for this to happen is for users to experience these emotions and give the appropriate feedback, which may take a long time.

Due to time constrains we will have analyze this issue in the future.

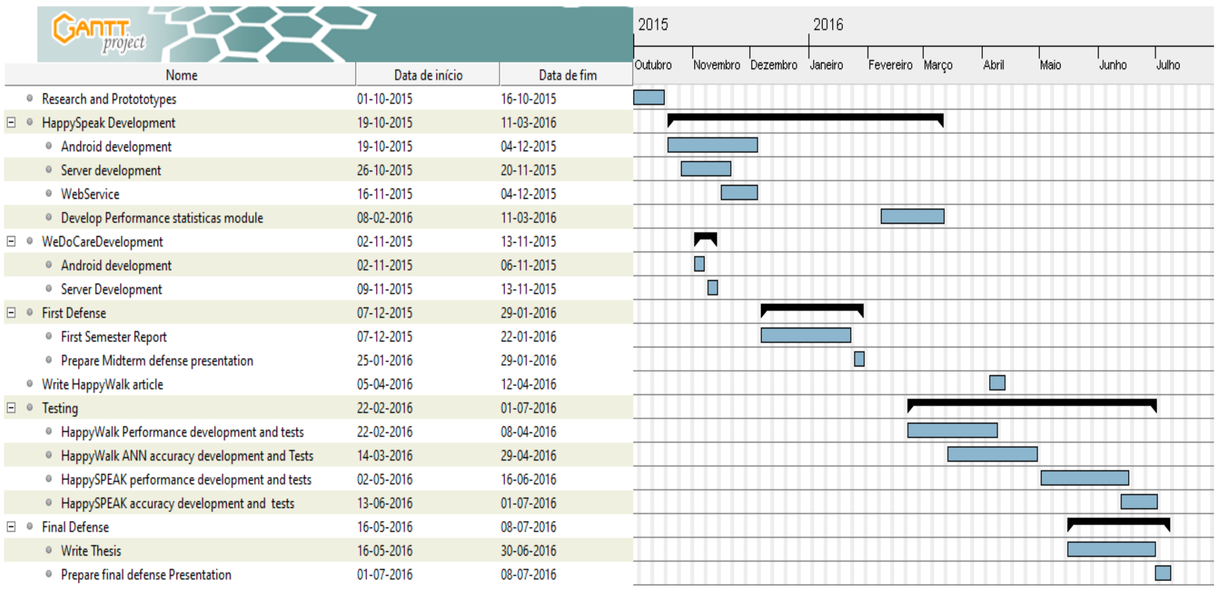
Chapter 6
Difficulties and Future Work

Most of the difficulties in the first semester where related with tight deadlines and the author's inexperience with some of the technologies used, namely Hibernate and PostgreSQL. These difficulties were surpassed by studying these tools, experimenting with several examples found online, and consulting with the supervisors.

By the end of the first semester, the apps were almost ready to be in production with just some minor bugs to fix and some small testing still required.

In the second semester we were able to fix most of those bugs and focus on power efficiency and neural network accuracy. Other difficulties slowed down this process, namely the problem with the phone entering a "sleep" state and not launching the tasks required for data collection and feedback requests. The time it took us to address this problem and fix it consumed a lot of our time and delayed the testing phase.

For a more detailed view of how the work described in this report progressed over the year, refer to the figure below:



Nome	Data de início	Data de fim
Research and Protototypes	01-10-2015	16-10-2015
HappySpeak Development	19-10-2015	11-03-2016
Android development	19-10-2015	04-12-2015
Server development	26-10-2015	20-11-2015
WebService	16-11-2015	04-12-2015
Develop Performance statisticas module	08-02-2016	11-03-2016
WeDoCareDevelopment	02-11-2015	13-11-2015
Android development	02-11-2015	06-11-2015
Server Development	09-11-2015	13-11-2015
First Defense	07-12-2015	29-01-2016
First Semester Report	07-12-2015	22-01-2016
Prepare Midterm defense presentation	25-01-2016	29-01-2016
Write HappyWalk article	05-04-2016	12-04-2016
Testing	22-02-2016	01-07-2016
HappyWalk Performance development and tests	22-02-2016	08-04-2016
HappyWalk ANN accuracy development and Tests	14-03-2016	29-04-2016
HappySPEAK performance development and tests	02-05-2016	16-06-2016
HappySPEAK accuracy development and tests	13-06-2016	01-07-2016
Final Defense	16-05-2016	08-07-2016
Write Thesis	16-05-2016	30-06-2016
Prepare final defense Presentation	01-07-2016	08-07-2016

Figure 47-Work progress over the year

There are still some things we could do to improve the projects developed throughout this semester like evaluating other types of classifiers to improve our system. There also some extra functionalities we could add to improve our accuracy even further:

- Exploring other classifiers like Support Vector Machines, Deep Neural Networks, Decision Trees, etc.
- Expand feature selection. Instead of using just messages and call logs, add information from other sources such as Social Network information and other messaging services.
- Change the classes we are using for emotion classification; HappySPEAK currently uses a slider to get its user feedback. It is possible that the ANN would be more accurate if we used specific classes to train our classifier.
- In the HappySPEAK app, we use an ANN in each user's smartphone because we are operating under the premise that each person is different emotionally and psychologically. The WeDoCare app manly monitors the user's environment, and as such, it would be interesting to crowdsource data from all users and tune all ANN's with it.

- Apply Speech and Text recognition algorithms to the HappySPEAK app. When the user makes or receives a call we could analyze his speech to infer his emotional state. We could also search for keywords in the content of text messages and use them as another source of information.

Chapter 7

Conclusions

After two semesters working on these systems, much was learned about how to correctly classify human emotional states in an accurate and power efficient way.

The results obtained by our tests also revealed that just having a lot of data of a user is not enough to accurately classify emotions; it is also necessary that the user experiences all the emotions in our spectrum in order to teach the ANN how to infer correctly.

Much has been learned throughout this year; mechanisms like Alarm Manager have proved to be very useful to extend battery life, and experimenting with machine learning concepts has given the author a new perspective on the field.

The first contact with technologies like PostgreSQL and Hibernate also proved to be challenging but the difficulties encountered and the way we surpassed them have contributed to the growth of the author's critical thinking and software planning.

Even though the problems we experienced in the second semester delayed the testing phase of our projects, we were still able to successfully execute our test suites and extract the desired results.

In the end, the objectives planned for this internship were accomplished with the two prototypes we built showing great promise; the testing results show that they are power efficient and the accuracy is quite high.

References

[1] (December 2016), About Speak" [Online] .http://www.speak.social/pt/sobre-o-speak/

[2] (January 2016), "UBHave Wiki", [Online], Available: http://ubhave.wikia.com/wiki/UBhave_Wiki

[3] (January 2016), "About Encog" [Online], http://www.heatonresearch.com/encog/

[4] (January 2016), "What is Hibernate", [Online], Available: http://www.tutorialspoint.com/hibernate/

[5] (January 2016), Rui Wang, Fanglin Chen, Zhenyu Chen, Tianxing Li, Gabriella Harari, Stefanie Tignor, Xia Zhou, Dror Ben-Zeev, Andrew T. Campbell, "StudentLife: Assessing Mental Health, Academic Performance and Behavioral Teends of College Students using Smartphones" Available: http://studentlife.cs.dartmouth.edu/studentlife.pdf

[6] (January 2016), Pedro Carmona, David Nunes, Duarte Raposo, David Silva, Carlos Herrera, Jorge Sá Silva, "HappyHour Emotion-Improving Mood With Emotionally Aware Application" Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7294480

[7] (January 2016), David Nunes, "HappyWalk HiTL Implementation".

[8] (January 2016), "Number of different Android devices" [Online] [Online], Available: http://qz.com/472767/there- are-now-more-than-24000-different-android-devices/

[9] (January 2016), "Key Facts and Figures" [Online], , Available: http://www.unhcr.org.uk/about-us/key-facts-and-figures.html

[10] (January 2016), "Quem ajudamos", [Online], Available: http://www.acnur.org/t3/portugues/quem- ajudamos/dmr-20110/

[11] (January 2016), David Sousa Nunes, Pei Zhang, Jorge Sá Silva, "A Survey on Human-in-the-Loop Applications Towards an Internet of All" Available: https://www.cisuc.uc.pt/publication/show/4253

[12] (January 2016), R. S. Desmond, M. F. Dickerman, J. A. Flemming, "A human in-the-loop cyber physical system: Modular design for semi-autonomous wheelchair navigation" Available:https://www.wpi.edu/Pubs/E-project/Available/E-project-041513-150903/unrestricted/MQP_Report.pdf

[13] (January 2016), D. Arney, M.Pajic, J. M. Goldman, I.Lee, R. Mangharam, O. Sokolsky, "Toward patient safety in closed-loop medical device systems" Available: http://repository.upenn.edu/cgi/viewcontent.cgi?article=1460&context=cis_papers

[14] (June 2016), "Recommender systems, Part 1; Introduction to approaches and algorithms" [Online], Available: http://www.ibm.com/developerworks/library/os-recommender1/os-recommender1-pdf.pdf

[15] (June 2016), Pedro Catré, Luis Cardoso, Luis Macedo, Almícar Cardoso, "Building Spatiotemporal Emotional Maps for Social Systems" Available: https://books.google.pt/books?id=4DNSzljn9u8C&pg=PA566&lpg=PA566&dq=Building+Spatiotemporal+Emotional+Maps+for+Social+Systems+conference&source=bl&ots=vYeP4MDV2v&sig=XVQCqcgHnnq_A2WfMYtUifqFlMg&hl=en&sa=X&ved=0ahUKEwju_a31iMvNAhXFHxoKHcmoBWcQ6AEILTAE#v=onepage&q&f=false

[16] (June 2016), Neal Lathia, Veljko Pejovic, Kiran K. Rachuri, Cecilia Mascolo, Mirco Musolesi, Peter J. rentfrow, "Smartphones for Large-scale Behavior Change Interventions" Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6562720

[17] (June 2016), Nicholas D. Lane, Petko Georgiev, "Can Deep Learning revolutionize Mobile Sensing?" Available: https://www.cl.cam.ac.uk/~pig20/hotmobile15.pdf

[18] (June 2016), R. Schoenen and H. Yanikomeroglu, "User-in-the-loop: spatial and temporal demand shaping for sustainable wireless networks" Available: http://userintheloop.org/html/publications.html

[19] (June 2016), About Yelp [Online], Available: http://www.yelp.com/about
[20] (June 2016), "Motorola Alert" [Online], Available: http://www.androidcentral.com/motorola-alert
[21] Dumitru Erhan, Aaron Courville, Yoshua Bengio, Pascal Vincent, "Why does Unsupervised Pre-Training Help Deep Learning?" Available: http://www.jmlr.org/papers/volume11/erhan10a/erhan10a.pdf
[22] (June 2016), "About Mapiness" [Online], Available: http://www.mappiness.org.uk/more/
[23] (June 2016) Stamatis Karnouskos, "Cyber-Physical Systems in the SmartGrid" Available:
[24] Mohamed R. Amer, Behjat Siddiquie, Coleen Richey, Ajay Divakaran, "Emotion Detection in speech using deep networks" Available: http://www.cs.umd.edu/~behjat/papers/ICASSP2014.pdf
[25] (June 2016) Kun Han, Dong Yu, Ivan Tashev, "Speech Emotion Recognition Using Deep Neural Network and Extreme Machine Learning" Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/IS140441.pdf
[26] (June 2016, "About WorldReader" [Online], Available: http://www.worldreader.org/about-us/
[27] (June 2016), "About CareMessage" [Online], Available: http://caremessage.org/about/
[28] (June 2016), "About Ankommenapp" [Online], Available: https://www.ankommenapp.de/