UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

TOWARDS AN INTELLIGENT BIOGRAPHICAL COGNITIVE PROSTHESIS

José Alberto Barreto Duarte Carilho

DISSERTAÇÃO

MESTRADO EM ENGENHARIA INFORMÁTICA
Especialização em Engenharia de Software

2014

UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

TOWARDS AN INTELLIGENT BIOGRAPHICAL COGNITIVE PROSTHESIS

José Alberto Barreto Duarte Carilho

DISSERTAÇÃO

MESTRADO EM ENGENHARIA INFORMÁTICA
Especialização em Engenharia de Software

Dissertação orientada pelo Prof. Doutor Tiago João Vieira Guerreiro
e co-orientada pelo Prof. Doutor Francisco José Moreira Couto

2014

Acknowledgments

My heartfelt thanks to my parents for all the support they gave me throughout the year, and words of courage and motivation. My sister who was always available to help and support me, and with her help everything was easier. Still to my brother in law who always showed interest in the project I was undertaking. To my girlfriend for all the support shown and for always being with me in the most difficult moments. To my faculty colleagues and friends especially to Carlos Barata, Tiago Aparício, Fábio Santos, Gonçalo Semedo, Claudio Saramagaio and Rita Henriques for their support, companionship and all the help they were always available to give me. To João Martins and Oliver Schnell my project partners, for the help which existed between all of us, and the group spirit created from the day one. I also want to thank the teachers Luis Carriço, Carlos Duarte and my co-supervisor Francisco M. Couto by all the ideas suggested during project creation. A special thanks to my advisor Tiago Guerreiro throughout his support and for guiding me throughout this year and in the development of the project.

I dedicate this thesis to all those who supported me throughout my life

Resumo

A demência é um problema mundial que está a aumentar à medida que a população mundial fica mais velha. Os pacientes de demência sofrem com um conjunto de problemas entre os quais problemas de memória e problemas cognitivos. Outro problema que também surge com demência é o fardo que os responsáveis pelos doentes têm que carregar quando lidam com os pacientes. Uma abordagem comum quando se lida com demência são as sessões de reminiscência. Mas um dos problemas no uso de sessões de reminiscência é a necessidade de toda a informação a ser mostrada ter que ser recolhida e organizada por quem vai auxiliar o paciente nestas sessões. Os responsáveis pelos doentes não podem ser incutidos de mais esta tarefa uma vez que eles já estão sobrecarregados. A solução para este problema é a recolha de dados e a organização destes de uma forma automática e sem carregar os responsáveis ainda com mais trabalho. Para isto foi construído um sistema chamado de RememberMe.

O sistema tem como objectivo a criação de uma prótese cognitiva da vida do utilizador. Usando dados recolhidos de diferentes fontes é possível criar um diário da vida do utilizador, para que este possa recordar os seus dias em sessões de reminiscência. Usando um smartphone para recolher informação da vida do utilizador, é extraída dos dados recolhidos informação de alto nível. O sistema faz uso de uma rede social, para com a ajuda de amigos do utilizador fazer uma validação e posteriormente um enriquecimento dos eventos da vida do utilizador. Ao usar a rede social a carga de trabalho associada à validação e enriquecimento dos eventos é divida por várias pessoas, em vez de ficar apenas sobre uma ou duas pessoas. Toda a informação recolhida é guardada numa base de conhecimento que irá representar a vida do utilizador através de eventos. Eventos estes que podem ser detectados automaticamente através dos dados recolhidos pelo smartphone ou que podem ser introduzidos no sistema pela rede social.

Este trabalho foca na criação da prótese cognitiva do sistema, onde é feita a extração de informação relevante para o utilizador, a qual é denominada de módulo de processamento de dados. Usando diversas fontes de dados para alimentar este prótese cognitiva é possível a detecção de eventos com base em dados recolhidos pelo smartphone ou em dados vindos da rede social. Ao usar uma ontologia para guardar informação recolhida, que podem

ser os eventos detectados bem como informação vinda da rede social, o sistema tem a capacidade para processar essa informação de forma a obter novo conhecimento sobre a vida do utilizador.

O módulo de processamento de dados é composto por três sub-módulos: detecção de eventos, gerador de questões e detecção de rotinas. O módulo de detecção de eventos usa os dados que foram recolhidos através do smartphone do utilizador para fazer a extracção de eventos. O objectivo deste módulo é fazer a transformação de dados de baixo nível para informação de alto nível e possível de ser interpretada pelos utilizadores. Este módulo faz uso de técnicas que são usadas na área da aprendizagem máquina para extrair os eventos da vida dos utilizadores com base em coordenadas GPS e com base nas fotos tiradas. Este módulo usa dois classificadores, um para as coordenadas GPS e outro para as fotos tiradas. Estes classificadores foram primeiro treinados com dados recolhidos pelo utilizador. Na fase de treino os utilizadores usam uma aplicação semelhante à que é utilizada num cenário real, mas com uma pequena diferença, que é a possibilidade de anotarem os dados que estão a recolher. Desta forma o classificador vai usar os dados recolhidos e anotados para criar um modelo para o utilizador e quando os dados reais forem processados ele usa esse modelo para detectar eventuais eventos na vida dos utilizadores.

Quando os eventos são detectados estes são guardados na ontologia. Quando os eventos são guardados na ontologia o módulo que gera as perguntas vai analisar os eventos que foram detectados e criar perguntas. Estas perguntas vão ser enviadas para o Facebook de forma a que os amigos do utilizador possam ajudar a enriquecer os eventos dividindo a carga pelas várias pessoas Este módulo foi desenvolvido de forma a que novas perguntas possam ser criadas, de forma a obter outro tipo de informação. Neste módulo foram criadas três perguntas, estas perguntas foram criadas com base na necessidade de completar a informação que não era conseguida através dos dados recolhidos pelo smartphone. Foram criadas perguntas para escolher qual o local certo de um evento, esta pergunta pede aos utilizadores para escolherem o local certo de um conjunto de possíveis locais sugeridos. Também foi criada uma pergunta para serem adicionadas fotos a um determinado evento, para isto os utilizadores só tem que comentar o post relativo à pergunta com uma foto e esta fica associada a esse evento. A última pergunta criada foi para identificar quem participou num determinado evento, para isto os utilizadores apenas têm que identificar uma pessoa no post relativo à pergunta, fazem isto utilizando as funcionalidades do Facebook.

O último módulo criado neste trabalho tem como objectivo a detecção de rotinas na vida de um utilizador do sistema. Para detectar estas rotinas em primeiro lugar é feita a identificação dos eventos semelhantes. Isto é feito através do uso da ontologia e da capacidade que esta trás ao sistema de comparar semanticamente os eventos. Usando esta capacidade da ontologia, vai ser criada uma lista de eventos semelhantes para cada evento e com o uso desses eventos semelhantes vão ser detectadas as rotinas. Cada conjunto de eventos semelhantes vai ser processado usando técnicas de aprendizagem máquina, do

qual vão ser extraídos os pontos em comum destes eventos. É usado um cluster para analisar os eventos semelhantes e a partir daí retirar a informação para criar a rotina comum a todos esses eventos. As rotinas identificadas usando este método podem ser consideradas como nós de uma rotina maior, como ir para o trabalho, ir almoçar, voltar ao trabalho e por fim regressar a casa. O que este método faz é identificar cada um destes nós, mas tendo mais um nível de processamento é possível criar estas cadeias de eventos.

Para testar esta abordagem à detecção de eventos e de rotinas, foi feito um estudo onde quatro utilizadores usaram o sistema durante uma semana. Nos dados dos restantes participantes foram detectados aproximadamente 65% dos eventos reportados nos diários de cada utilizador. No entanto apesar de este número ser relativamente baixo a detecção de eventos reportados pelos participantes pode ter ficado em algumas situações comprometida pois os participantes em algumas situações reportaram que não colocaram a aplicação a correr da forma correcta e no entanto reportaram esses eventos nos seus diários. Ao ver os eventos detectados pelo sistema e a respectiva análise feita pelos participantes do estudo pode-se confirmar que a maior parte dos eventos detectados foram confirmados pelos participantes. O que se traduziu num numero reduzido de falsos positivos, isto é eventos detectados mas não confirmados pelo utilizador.

Palavras-chave: Demência, Registo de vida, Aprendizagem Máquina, Base de conhecimento, Eventos de vida, Prótese cognitiva

Abstract

Dementia is a global problem which is growing as the world population gets older. Dementia patients suffer from a series of problems including memory problems and cognitive problems. A problem associated with dementia disease its the burden that caregivers have when trying to maintain dementia patients with a decent lifestyle.

The RememberMe system tries to create a prosthetic memory using data collected from a smartphone and data from a social network. This data is then used to create a personal diary to be used in a visualization tool.

This work presents an intelligent cognitive prosthesis which aims to extract high level information from low level data. It's achieved by extracting relevant life events from GPS data and pictures taken by the user and using an ontology to reason over the data in order to detect routines. The cognitive prosthesis consists on three modules, a life event detection module, a routine detection module and a questions generator module. The extraction of life events was made using machine learning techniques, using a classifier which was trained with tagged data from the user, the detected events are then saved on the ontology. Then using the ontology to compare the similarity between events, and using machine learning techniques it was possible to detect routines over the detected events.

A study with five participants was done to evaluate our approach to detect events and routines. By collecting data during a week and creating a diary of the relevant life events, it was possible to compare the effectiveness of this approach.

Keywords: Dementia, Life Logging, Machine Learning, Knowledge Base, Life Events, Cognitive Prosthesis

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As the world population gets older the number of older people with dementia disease increases. Statistics show that about 25 million people suffer from dementia worldwide in 2000 [33]. With the increase of life expectancy this number tends to grow even more. Dementia is a chronic or persistence disorder of the mental process caused by brain disease or injury. Main features are memory disorders, personality changes and impaired reasoning [2]. Dementia is irreversible and the only approach to deal with it, is control the development of the disease. Dementia has three main stages: mild stage, moderate stage, severe stage [1].

On the mild stage of dementia, patients will experience difficulty concentrating as they will suffer from a decline on memorize details of recent events. They may become confused when traveling to new locations by themselves. When leading with complex tasks they struggle to complete them on a efficient and accurately manner. On this stage start to isolate themselves and try to avoid their friends and family, because socialization with other people is awkward to them since they start losing their communication skills. On moderate stage dementia patients suffer from major memory deficiencies and this implies they need assistance to complete daily tasks. The memory loss is now very clear and the patients don't have capacity to remember major aspects of their current lives. On this stage they start to forget the names of close relatives, and recent life events become fuzzy as they fail to remember details of these life events. Also on this stage many of the patients only remember of some details of earlier life. Communication skills are now affected and they struggle to communicate with their relatives and friends. During conversations patients may suffer from difficulty in concentrating and don't understand the topic of the conversation and not fully understand the important information.

The most common approach to lead with dementia, is using caretakers to help patients on their daily tasks [23]. On mild stage of dementia, caretakers don't need to be constantly looking for the patients, giving patients some freedom. When dementia evolves to next stage, the caretakers should be much closer to the patients, helping them with daily tasks and making sure they are safe because patients can have unexpected behaviours.

1.1 Motivation

Mild stage dementia patients present a series of, such as mood changes or major memory deficiencies. Memory deficiencies can be seen when patients forget relevant aspects of their lives. Forgetting names of close friends and family members and aspects of earlier are lost. This will lead to a degradation of communication skills, as they begin to avoid participate on conversations [2].

Reminiscence therapy is an activity used when leading with dementia patients. As they enter in mild-stage they might suffer from depression and isolation. This approach has as objective to cause on the patients a state of well being [9] [10]. This is achieved by encourage the patients to talk about their lives, using text, photos or any object that can be used to stimulate the patient to talk.

The most common approach when leading with dementia is the caretaker approach. However the burden on the caretaker is huge, since they must provide help and assistance dementia patients on a big part of their time [2][18]. Caretakers aren't able to provide personalized information to all the reminiscence sessions that dementia patients attend. So we want to provide a solution to unburden caretakers with the task of collecting all the information used in these sessions.

A possible solution is to provide a system which is able to collect personal information about a patient's life, and use it to help patients in the reminiscence therapy sessions. In this case the patients will have information about their lives, helping them to talk with their families and friends about their days. This will help both patients and caregivers, because the patients will enjoy talk about their lives and the caregivers won't have the burden to collect all the information.

1.2 Objectives

The RememberMe project has the goal to provide a solution to unburden the caretakers with the task of collecting data to use in reminiscence therapy sessions. And helping patients, by solving two of the unmet needs, the need for general and personalized information and the need for social contact and company, identified by Lauriks et al. [21].

The objective of the work here presented is to create an intelligent cognitive prosthesis which is able to extract high level information using low level data. This main goal was divided into three sub-objectives:

1. Find relevant life events, using machine learning techniques over low level data such as GPS coordinates and metadata of pictures.

2. Build a mechanism which is able to find missing information on the events and generate questions, to be sent to a social network, to fill the gaps.

3. Find routines in detected events, combining the use of an ontology and machine learning techniques.

By achieving these three objectives, it's expected create an intelligent cognitive prosthesis which is able to learn from the past experiences.

1.3 Biographical Prosthesis

The work here presented is included in a bigger project, named RememberMe. This project result from a collaboration between three students. Each one of us was responsible to deal with specific parts of the project. João Martins was responsible to create the smartphone application and the visualization tool. Oliver Schnell created the enrichment framework, which makes the bridge between the system and the social network. And the work here presented is the data processing module.

The module where raw data is transformed into human readable information. This work is divided into three parts, detection of events, creation of a knowledge base and detection of routines. The process of detecting life events, based on raw data, uses machine learning techniques to extract events from the data sent from the smartphone. The knowledge base was used to store life events, used in the process of detecting routines and in the enrichment phase, where questions were generated by querying the ontology with SPARQL. The process of detecting routines, as told before uses the ontology to find similarities between life events followed by machine learning techniques, allowing routines to be found.

1.4 Contributions

This work focuses on providing an intelligent cognitive prosthesis to support projects dealing with biographical data for people with dementia.

1. We contribute with a semantic network that is able to store life events of the user. It will be used as a store mechanism which allows the system infer new knowledge about the user's life.

2. A classification module to detect life events, from raw location data and images. This module can be extended by adding new classifiers to run over different types of data.

3. A reason module that is able to identify routines from sets of events. Combining the use of the ontology and a cluster to identify the routines.

4. A module that is able to find missing information in the ontology and generate questions.

1.5 Publications

The work presented herein contributed to the publication of two paper, one in a international conference:

- João Martins, José Carilho, Oliver Schnell, Carlos Duarte, Francisco M. Couto, Luís Carriço,Tiago Guerreiro. Friendsourcing the Unmet Needs of People with Dementia, W4A 2014. [25]

- José Carilho, Oliver Schnell, João Martins, Luís Carriço, Carlos Duarte, Francisco M Couto, Tiago Guerreiro. Predicting Relevant Events in the Life of a Person with Alzheimer. Bioinformatics Open Days 2014

1.6 Document structure

This document is structured as follows, in chapter 2 is shown the previously work done in the area. Chapter 3 shows an overview of the system, and explains in detail the modules which are in focus in work. The knowledge base structure is shown and explained in the chapter 4. The module of detecting life events is revisited in detail in chapter 5 where are explained the algorithms used. Chapter 6 shows how routines are detected and how we arrive to the actual solution. An evaluation is detailed in chapter 7. Finally the conclusions of this work are in chapter 8.

Chapter 2

Related Work

2.1 Life Logging

Life logging can be seen as an activity of collecting data from a wide range of sensors and being able to represent a person's day. Life logging tools, can be used as external memory aids that help persons remember past experiences. These tools must use the large amount of data collected by sensors and transform it into useful information about users' days.

Appan et al. [5] present a tool that enables a group of people to create and share stories. This minimal authoring tool for everyday experiences, allows the user to insert images or text, or both combined. This tool places a minimal authoring burden on users. With this tool each user can create its own viewpoint of an event, and in visualization stage a user can see the event by different viewpoints.

Using a tool to create stories about an event, can be useful to see different points of view of the same event, however this approach won't unburden users with the task of creating stories. A different approach is used by Hodges et al. [15]. They present a computing device called SenseCam, this device is a wearable stills camera, which is able to capture the user's day. By recording images and collect data from its sensors. This device can be used to create personal life logs, without burden the ones who wear it.

Using tools to automatically collect and transform data is a common approach in life logging tools, because the task of searching for human readable information in the whole set of data would be very time consuming for the user. Different approaches to deal with collected data were created including the work of Lee and Dey [22] where they present the design of a new life logging system that captures data such as photos, ambient audio, and location information. The life logging system has a design that follows a three-step process of capture, cue selection, and review to support recollection of episodic memories. The first step is when the system passively records experiences chosen by the user, in the second step caregivers uses the CueChooser application to build a narrative of the experience by selecting and annotating the most salient memory cues from the recorded

data. Finally with the MemExerciser application, the system presents selected cues. Cues are shown in a way that maximizes opportunities for people with EMI think deeply about these cues. This will trigger memory recollection on his own without burdening the caregiver.

Location data is one of the main types of data used in life logging tools, using this type of data is possible to find places where users where throughout the days. This is what was done by Ferrari et al. [14] where they explore how to make a diary using location data. They did a search about the tools to collect data, and a way to extract human readable information from that data. They wanted to find the places where the users were and identify routines. They present a prototype which uses location data to build a diary. They start by doing a place identification, and end by identifying routines with the places identified. The work of Kim and Giunchiglia [20] also follows the concept of unburden users with the task of reviewing data. They present an open platform, named eLifeLog, that saves users' memories and experiences. The eLifeLog runs on a user's private cloud. Data is sent to the server and the server re-sends it to external data analyzers. Allowing the server saves data without interruptions, and have one analyzer for each type of data.

Different types of life logging tools, such as tools to identify what users do in house, are used to build a user diary of activities. Al Masum Shaikh et al. [3] explain their approach to record the users' lives, and to share daily events with family or friends using social networks or messages. Their approach uses environmental sound cues and common sense knowledge to detect human activity. 114 acoustic sounds of different types were collected, sounds produced by object interaction, by the environment itself or by deliberate action of an individual. These sounds were then used as underlying clues, to infer events in an individual's life. When an activity is detected the system sends a message to the family or to the friends.

In the work by Al Masum Shaikh et al. [3] was possible to see how a real time life logging tool can be useful. When life logging is made in real time, it can be used to help people by presenting useful information by knowing the context where the users are. This is what Kikhia et al. [19] did with MemoryLane, a context-aware life-logging system, with the goal to support independent living and improve the quality of life of people with mild dementia. The system has a mobile application that offers real time support, being useful to patients in their daily life activities. The mobile application offers navigation support, nearby person identification, current location identification and an activity guide, increasing the feeling of safety and help them to build episodic memory. Data is collected from a series of sensors, carried by the person or located in the environment where the person is. Data is stamped with time, and sent to the system's local storage. Once uploaded, data is processed and divided into context data and content data. Then

it's analyzed, filtered and aggregated automatically.

2.2 Semantic Networks

Semantic networks are used to represent relations between concepts, a common form of semantic networks are ontologies. An ontology is used to represent knowledge within a domain, it's possible to represent types, properties and relations between concepts of a domain. Ontologies are widely used in the area of artificial intelligence since they allow to reason over the saved data. Uschold and King [30] propose a methodology to build ontologies. They identified some main aspects that an ontology should have. Identify the purpose of the ontology, building the ontology, text definitions for each of the concepts and relationships and identify terms to these concepts and relationships. The final aspects referred were, evaluation and documentation of the ontology.

Davies et al. [11] used an ontology to create Popcorn, a system that tries to map users' brain knowledge to a database representing the user's knowledge. Knowledge in a person brain is made by associations, the system can represent associations between concepts and then store them in a database. To achieve this, they identified requirements for the application. These requirements are: Recording knowledge should be quick and painless; Returning to knowledge recorded in a different context should be quick and painless; Reorganizing knowledge should be easy; Knowledge should be express formally or informally; Public content should be easy to assimilate; The tool must be compatible with human memory. By following these design goals, they expect create an application that is easy to use, on both phases, of adding knowledge and knowledge searching. The main concept of the Popcorn system is the *kernel*, representing a concept of the real world. A *kernel* can be within other *kernel*, allowing a real-world concept be part of different contexts. *Kernels* can be related with each others, creating relations between them. The human brain doesn't have the capacity to work with all knowledge at one time, so *kernels* are the authors approach to deal with this issue. When exploring one *kernel*, users are working with a limited knowledge and without knowing what is outside the *kernel*. After some tests and results analysis, Popcorn showed that is a tool that is able to help users represent a lot of complex knowledge.

A different project which allows users represent their knowledge is the work of Cañas et al. [7]. Where they create CmapTools, a software environment that allows users to represent their knowledge, by creating concept maps. Concepts maps are a graphical display of concepts and relations. The concepts on a map are represented by circles or boxes and the relations are represented by direct arcs between the concepts. Creation of relations between concepts is simple, using drag-and-drop operations to create a relation between two concepts. This tool provides a simple and intuitive way to create a knowledge base, rep-

resented on a concept map, where users can see all the concepts and the relations between them.

Representing life events can lead to uncertainties about what to represent, Trochidis et al. [29] present an ontology model to represent life events on a public service. This model can be used in public administrations by domain experts to represent life-events. The proposed ontology has the following classes: life-event, public service, citizen, user profile, input, output, rules, public administration document and non-public administration object. This model has the advantage that a life-event is a concept that is well defined, the relations between concepts that are part of the life-event are easy to see.

In the work of Davies et al. [11] and Cañas et al. [7] it is possible to see how to represent knowledge in a human readable manner, however the users have to manually input the data in the knowledge base. Wang et al. [31] present a system which automatically creates a knowledge base. The system is composed by three connected components. The first component of the system the MADden, is responsible for knowledge extraction. MADden extract features from text and then uses it to analyze the text. The extracted features are used to perform inference. The next component is the ProbKB, a probabilistic knowledge base, with the objective to generate knowledge, based on entities, relations and rules that were previously extracted. The final component is the CAMel (Crowd-Assited Machine Learning), this component's main objective is to improve the uncertainty of the knowledge on the system. With these three components is expected have a semi-automatic system that generates knowledge and uses humans to deal with possible mistakes generated by the automatic part of the system.

Besides ontologies are good to represent knowledge, they are used as well to infer new knowledge. Using the capacity of ontologies to reason data was what Shi and Setchi [28] did in the development the system Life Story Book. This system goal is to provide access stored memories. It uses a semantic model, based in the use of ontologies and advanced algorithms for feature selection and dimension reduction. It includes modules of natural language processing, named entity recognition, semantic feature matching, clustering and matching. The information inserted in the system is done by users. Since the type of information may vary from user to user, they decided to implement a user-oriented ontology. The building process of the ontology is interactive and the users play a major role on it. To achieve topic identification based on semantic similarity they use the Onto-SVD algorithm, it's based on the idea that the combination of terms and named entities allows representing the main concept of the ontology as well as the semantic meaning of each information object.

Reason over ontologies can be used to find new knowledge but also to check the consistency of the knowledge base. Wang et al. [32] propose a Ontology Web Language (OWL) ontology, to model context in pervasive computing environments, and to support

logic-based context reasoning. Reasoning is used to check the consistency of the information and to derive high-level context by reasoning over the low-level context. The context model is divided into upper and specific ontology. The upper ontology is a general ontology to capture the general features of a context. The specific ontology is used to capture the details of a specific ontology. To infer new information about the context they divided the reasoning task was divided into two: ontology reasoning using description logic, and user-defined reasoning using first-order logic. The ontology reasoning is used to check the concept satisfiability, class subsumption, class consistency and instance checking, the ontology reasoning use properties like the transitive property to infer high level information, an example given is that is possible to infer that the user is at home by knowing that he is at his bedroom. The user-defined reasoning is used to know to what the user is doing based on the low-level information. The user defined rules allows a more flexible reasoning, using first order logic rules.

Finding routines in a person's life is also a feature which is able to be achieved using an ontology. Reasoning over the preferences of a person is possible to predict the next step on a person's life. This is what was made by Ngoc et al. [26] where they propose a way to find a behavior routine using a formal model. They propose two OWL based ontologies. The first is a user preference ontology, which includes vocabularies to represent user beliefs, desires and intentions. The second is a behavior routine ontology. Both routines are based on a context model created by the research group, named Contel. The users' preference ontology added the Preference class to the Contel model. A preference can be POS or NEG, representing the like or dislike. With this ontology is possible to represent users' likes or dislikes about a location or an activity. To represent routines, was created the behavior routine ontology. In this ontology the routines can be daily, weekly or monthly. A routine is composed by a sequence of routine items. These routine items have a location, a time and the activity that is made and these items will be linked between them by two properties: isAfter and isBefore. To infer new knowledge about the users' routines a OWL-support reasoning mechanism was used.

2.3 Events extraction

Extracting high level information from big collections of data is widely used in life logging tools. Because the collected data by sensors isn't human readable, it's necessary have a mechanism which transforms it into human readable information.

Using images to find life events is very used, since SenseCam is a very popular device for life logging. Doherty et al. [12] show a technique which allow extract life styles traits and characteristics from a data collection. SenseCam is used to collect data from users' days. Then a trait interpreter tool is used to extract the desired lifestyles traits. The first

step in this process was a feature extraction from the images collected by the SenseCam. The second step was select the traits. To train their concept detectors, they used manually annotated images from five of their users.

Conaire et al. [8] also used images collected by the SenseCam to cluster a day's data into meaningful events. To do this they extract features from the images collected. They have extracted two features from the images and have used accelerometer sensor readings to cluster images into events. Two methods to cluster were used. The first was to cluster via image classification, where they grouped the images using a Bayesian approach to classify each image into one of three classes: Static Person, Moving Person or Static Camera. The second method used, was clustering image groups via a statistical approach. In this method they used a modification of the statistical region merging algorithm of Nock and Nielsen, the algorithm is used to cluster images instead of pixels.

Machine learning techniques are very used to extract relevant information from large amount of data. The use of clusters is a common approach to extract relevant information, using location data. And classifiers are used to detect activities that require some previous training. Zhou et al. [34] present the ZhiWo system, a prototype human activity recognition system. It processes data, collected by smartphones, based on a machine learning model which is trained by the users. They use a smartphone application and tag the activity that are currently doing with one of the sixteen activities that are predefined in the application. The annotation process is the training set of the machine learning model. The data collected include photos, accelerometer readings, GPS, WIFI, bluetooth and ambient environmental measurements. When the model is trained, a predictive model to detect real-life human activities is ready to be used. Allowing recognize real life activities.

Machine learning techniques can be used in activity recognition. In this particular area the use of classifiers are very common, since the data from the sensors is used to build models capable of detecting activities. Hong et al. [16] propose a method to recognize daily activities using accelerometers and rfid sensors. The combination of these two methods urges from the necessity to find a method that is more accurate than the accelerometers and which doesn't detect so many false positives, as just using rfid sensors. So combining the use of the accelerometers to recognition of body states and rfid sensors to detect human object interaction, the recognition of daily life activities can be improved. Like in the work of Hong et al. [16], Bao and Intille [6] also propose a method to use accelerometers to collect data from daily activities, with the objective of detecting activity using classifiers. They identified four features to use in the classifiers. These features were mean, energy, frequency-domain entropy and correlation of acceleration data. The participants performed daily tasks and these activities were annotated into the accelerometer data, and then using the labeled data the classifiers were trained. The participants did tasks to per-

form each one of the twenty activities identified by the authors. In the end using decision tree classifiers was achieved an accuracy of 80% on activity recognition.

The use of machine learning techniques in real time applications can be very useful when dealing with older people since is possible to use them to detect when people fall. In their approach Luštrek and Kaluža [24] They use sensors to collect positions of body parts to detect activities such as lying down, sitting down, standing/walking, sitting and falling. They start by choose the best attributes to represent the state of the human body. The locations of body parts in the reference coordinate system, the locations of body parts in a body coordinate system and the angles between adjacent body parts were the attributes found that best represent the state of a human body. They used WEKA, an opening source machine learning suite, and compared eight algorithms available in WEKA, reaching an accuracy of 95%.

2.4 Routines detection

Routines are regular activities that a person does on a regular basis. It's possible to identify different types of routines, like routines of activities or going to the same place every day. These routines vary in type but also in the period which they occur.

Since routines won't bring a burden to the users in order to correct them, some assisted mechanisms are used to create the routines, which is the case of the RoutineMaker application. Antila et al. [4] created an application to detect daily tasks performed on smartphones. And by analyzing these tasks, suggest routines to the user by using clusters to detect the more relevant places and applications usage. The application collects data from the smartphone, such as locations, time and applications used and sends this data to the server, where it's analyzed by an algorithm. This algorithm is divided into two phases: geographical and application clustering. At the first stage the algorithm will cluster the most important places once these places are found the application clustering phase begins. The applications used in each of the locations are clustered to detect which are the most relevant applications for each location. Once a routine is detected, it's shown to the user, which can accept the routine or do changes that he found necessary.

Using automatic mechanisms to detect routines without burden the users is a different approach. And intelligent mechanisms use it to detect routines. Farrahi and Gatica-Perez [13] propose single and multi modal routines representations with multiple time scales. They address the problem of detect daily routines, by comparing weekday vs weekend routines. Data from 30 university students was used to collect location and proximity information. The location data represents the place where the students were and the proximity data represents information about the proximity of the students with each others.

Location and proximity were quantified in two levels: one fine-grain and one coarse-grain. In the one fine-grain, each day is divided into blocks of 30 minutes and each of this blocks is labeled with H,W,O,N. The letters represent the place where the student was at each block of time. H is for home, W for work, O for other and N to no reception. In one coarse-grain a day is divided into 8 coarse time slots each of them representing 1,5 hour intervals, using the one fine-grain labels they built words representing each block. Then a binary element representing each one of the labels is assigned to one if the location was presented in the word or zero if it isn't. To represent the proximity, the process is similar but instead of using labels, they use just binary elements to represent if were other students in the proximity. To find routines they classified the data using a support vector machine with a Gaussian kernel. Their training strategy was to train with all the students data but one, and test the model on the one student data that doesn't was used in the training, doing this for all students.

Detecting activity routines can be achieved using different methods, using accelerometers to identify the most common activity is one of them. Using high level information about a certain area is a different approach to identify the activities done in that same area. The work in the development of an activity-aware map is described by Phithakkitnukoon et al. [27]. This map will provide information about the most probable activity in a certain area of the map. The information will be used to construct individual patterns of daily activities. The first step was to prepare the data. They extracted the trajectories and the stopping points of each user. A trajectory is detected by measuring the distance between adjacent positions. If this distance is greater than a threshold, then it's considered a trajectory. A stopping point is detected when the user stays in a specific location for a long period of time. The working place of the users is detected as the most frequent stop during day hours. The map creation is done by dividing the map into square cells of 500 by 500 meters. Then each of the cells are labelled with activities that are expected in that specific area. An example is, if a certain area have many restaurants it'll be classified as a eating area. The concerning activities are: eating, shopping, entertainment and recreation. For each stop of the user an activity is identified according to the map position where the stop occurred. To infer the daily activity pattern for each user, a 24-hour time scale into eight 3-hour segments. So the daily activity pattern is a sequence of activities performed by the user on each stop. Each one of the eight segments will be assigned with the most common activity in that time interval.

Probabilistic methods are used to find potential routines in a person's life. Using these methods is possible to see which is the most probable activity to occur and predict the next one. Having the most probable activities in a day, it's possible to detect daily routines by detecting the most probable activity at each time of the day. Huynh et al. [17] propose

a novel approach to detect daily activities patterns using topic models. The use of topic models allows an automatic discovery of patterns in a user's daily routine. The idea behind the model is to detect how much an activity pattern is active at a certain point in time. They collected data from a individual which was wearing two sensor, they asked him to annotate his daily routines, and also his daily activities. The individual identified 34 activities and 4 daily routines. They used the topic models to discover routines based on activity recognition. The data collected by the participant was labelled and used to train a classifier, creating activity patterns. These activities patterns were then used to describe and recognize daily routines. To discover the routines from streams of low-level data, they used the framework provided by Latent Dirichlet Allocation. To use it they had to choose the nature and the size of the vocabulary, the size of the documents and the number of topics. To create the documents to be used by the topic models they create a document for each window of time, where the documents represent a mixture of activities of that time window. Then they just see what is the most probable activity in the time window.

2.5 Discussion

Life logging tools are widely used by reminiscence therapy tools, since they provide a solution to automatically collect data to be used in these tools. Studied approaches showed how different types of data can be collected, such as locations and photos. These types of data are the most useful to represent users' days It was shown how a real time life logging tool can be useful to older people, since it can alert the caregivers of unexpected activities.

In the section of semantic networks were demonstrated examples of how these storing mechanisms can be useful to represent knowledge. Semantic networks are used as a reason mechanism, leading to the discover of more knowledge. And it was shown how ontologies can represent routines and predict the next steps.

Detect events from big collections of data is a task which most of the times is performed using machine learning techniques. Clusters are widely used to detect relevant places from a collection of raw locations, while classifiers are used on activity recognition. Classifiers are trained with data from the users. This training can be done in real time where users identify the activity that they are doing or can be done in the end of the data collection process, where they identify the activities that they are doing by looking at the time of the activities.

Different approaches to detect routines were studied, like the use of an assisted method to create the routines or automatically identify routines using probabilistic methods or

machine learning techniques. A common point in all of the approaches is that all of them try to identify the routines in two steps. First identifying the most probable or common activity at a certain time and then creating the routine by linking the activities with each others.

Despite there are life logging tools which are able to represent a user's day and doing it automatically without burden users, they lack a method of validation. Since doing events extraction isn't 100% accurate, it's necessary to do a validation of extracted events. The solution achieved by the RememberMe project tries to solve this problem by using a validation mechanism through a social network.

Using clusters to extract events from raw locations are a common use, however most of these clusters lack in flexibility. The number of clusters to be found must be known at the beginning of the clustering process and the data should be all collected before this process starts. Using a classifier with trained data will allow to process the data in real time instead of waiting to process the data of a whole day, making possible the use of the module in a project which provides real time information.

The approach used to create an intelligent autobiographical cognitive prosthesis don't have the objective to substitute or fill the gaps of the approaches identified, instead it aims to combine different approaches to create a solution to extract high level information from biographical data.

Chapter 3

Cognitive prosthesis for People with Dementia

The processing data module explained in this work is part of the rememberMe project, a system created to help dementia patients and their caretakers. This cognitive prosthesis will transform raw data into human readable information. In figure 3.1 is shown a schema of the rememberMe system where is possible to see the flow of information throughout the system.

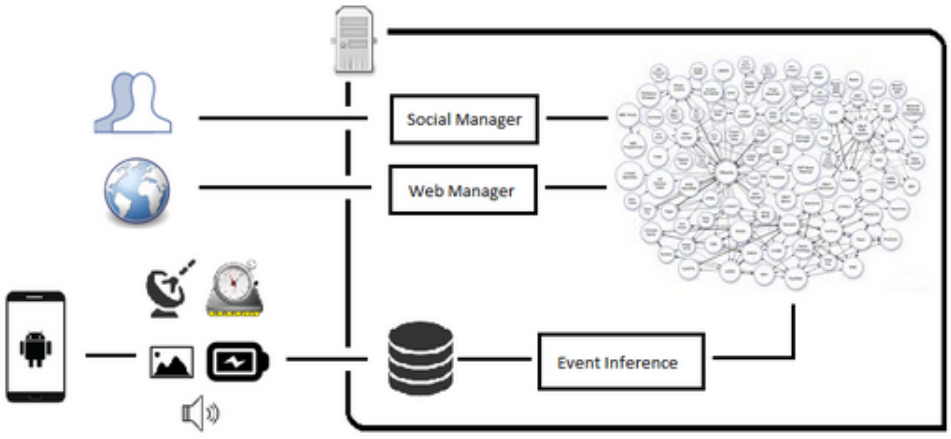

Figure 3.1: The rememberMe System

3.1 The rememberMe project

The rememberMe system helps dementia patients and their' families and friends dealing with the disease. This system takes into account the burden that the caregivers already have while leading with dementia patients. It provides personalized information about the patients without burden the caregivers. Data is collected automatically by the smartphone

and it's processed and enriched without the caregivers had to look to the information found. The rememberMe system is composed by three main components. A smartphone application responsible to collect data from the sensors available, a processing data module where data collected by the smartphone is transformed into information that is human readable. And an enrichment framework responsible to make the bridge between the system and a social network, so patients' friends can enrich the information collected. When the data collected by the smartphone is transformed into human readable information, a visualization tool can be used to reminiscence therapy sessions.

Figure 3.1 shows the whole system where is possible see and comprehend the flow of information through the whole system. The data starts being collected by the smartphone, then this data is sent to the server where it's stored in a database. The data processing module is monitoring the database and when new data arrives to the database, the module processes it. This module transforms the data into life events. These life events are used by the enrichment framework to collect more information from a social network. Patients' friends and family will be able to add information to each life event.

3.1.1 The smartphone App

The use of a smartphone was a natural choice since is possible to find a smartphone at a relatively low price and they have a set of sensors that can provide good information about patients' days, and most of people use them in daily life situations. The application collects GPS data, pictures taken by the individual, as well as the sound's amplitude collected by the smartphone's microphone. By using Google calendar the application is able to automatically retrieve events saved in the calendar. This application also allows collection of information about SMSes and calls received or sent by the user. In order to protect users privacy the information collected is just the metadata associated with the SMSes and the calls, just the time and the contact are collected.

The smartphone allows to remind patients of some tasks they must do. As patients may not have the ability to set the reminders by their own, a mechanism of setting reminders was built. The reminders can be set via Google calendar, SMSes or MMSes. Using Google calendar the responsible for setting the reminders creates an event and shares it with the patient. Different options to set reminders arises from the necessity to create reminders even when Internet isn't available. So this is achieved by sending an SMS or a MMS using a specific text. The application is monitoring the SMSes and MMSes, and when the specified format is detected, the application creates a reminder with the details inside the SMS or MMS.

3.1.2 Enrichment framework

The use of a social network to enrich the events arises from the need to unburden the caregivers with the responsibility to add all the contents used in the reminiscence therapy tools. The use of a social network as an information source about the users activities has the objective to split the load and the burden of adding information by a group of friends and family of the patient. This framework allows Facebook users to answer questions on the patient's wall. It offers a set of mechanisms to the patient's friends, so these can quickly and without effort enrich the events of the patient. The framework allows add pictures, quickly identify the place of an event or identify who were with patient on a event. The framework uses questions generated by the data processing module, and is able to detect the answers on Facebook.

Another feature of the framework is to allow Facebook users to add complete events to the system. Even with the data collected by the smartphone, the system may fail to detect some important events in the user's life. To minimize this, the framework is able to detect when a Facebook friend adds a life event on the patient's wall and retrieve information to create an event on the system.

The framework also collects information about the events in other platforms. It allows the system to collect pictures about a place identified as the place of an event. The pictures will be useful to help the patients, when using the visualization tool.

3.1.3 The visualization tool

A visualization tool was created as mechanism to help the caregivers in reminiscence therapy sessions. The visualization tool is a web application, which retrieves information from the server and displays it in way that patients' memories of recent events may be triggered. This tool is divided in two sections. First the routes taken by the patients are shown and the photos taken with the smartphone are shown in the map. Second, information is shown as events detected by the data processing module. The events in the visualization are displayed, after being enriched by patients' friends and all the details gathered by this framework are presented.

3.2 Data processing module

The data processing is where all the data processing occurs. The module can be seen as a piece of software which can be used to detect events and routines for general users. Making this module independent from the system allows it to be used on different projects. It's composed by three main components, the event detection module, the question generator

module and the routine detection module. All of this modules are directly connected with the knowledge base. Once the events are detected, they are stored in the knowledge base so the questions generator, which is monitoring the knowledge base, detect new events and generate questions to be answered on Facebook. The remaining module is responsible to detect routines, using users' events. This module analyzes the knowledge base, and by comparing the similarity between the events it detects routines. The data processing module is constantly monitoring all the storing mechanisms, the database and the knowledge base. Once it detects new data or new information, it triggers tasks to perform the desired actions.

In this module there are two particular classes that aren't included in none of the three modules identified earlier, they are the OntologyScheduler and the OntologyOperator. The first class is responsible to control all the tasks that make changes on the state of the ontology. Having all the tasks presented in this class is easier to add new tasks to the module and it also makes the debugging easier since all the tasks have this point in common. The OntologyOperator class is responsible by the changes in the ontology. By centering all the methods that make changes in the ontology in a single class, decreases the number of code lines and reduces code complexity.

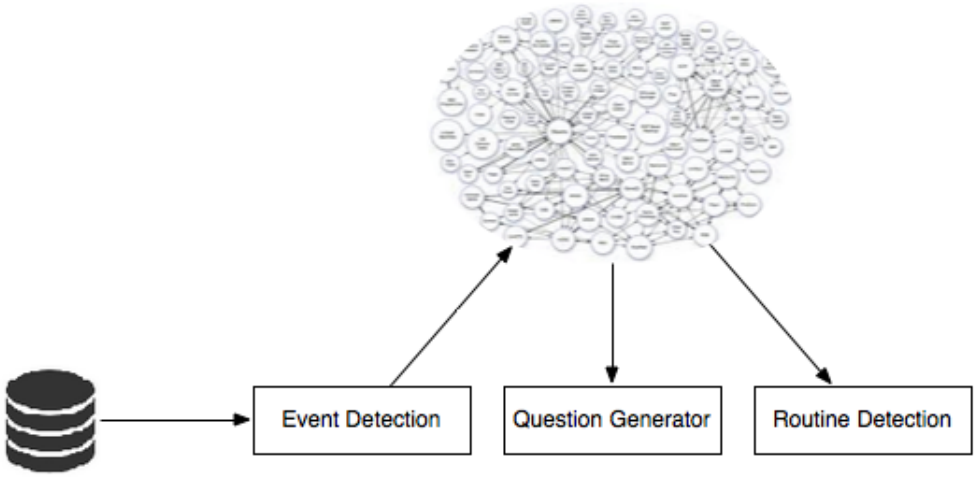

Figure 3.2: Data processing module

3.2.1 Event detection

This module is responsible to detect relevant events from data collected by the smartphone. The module includes two classes which are responsible for the detection of events based on the GPS coordinates or in pictures taken by the user. The classes will read data from the database, which was previously loaded by the smartphone application and then

using machine learning techniques to detect the user events. By using classifiers with trained data by the user, this module detects events and save them on the knowledge base by using the OntologyOperator class.

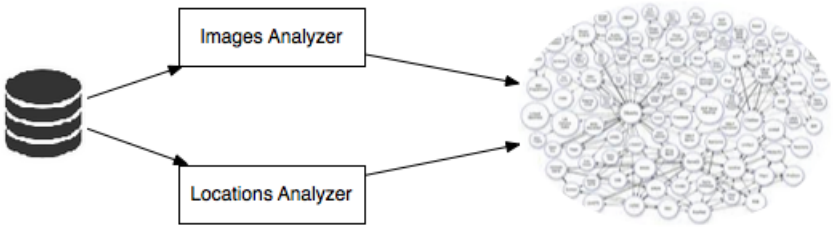

Figure 3.3: Event detection module

3.2.2 Questions generator

The questions generator is the module responsible to check missing information about the events and create questions to collect the missing information. This module uses SPARQL queries to query the ontology for missing information and then with the results create questions, which are sent to Facebook through the enrichment framework. This module is composed by three classes, DetectingMultiplePossibleNamesTask, DetectingMissingPhotosTask and DetectingMissingFriendsTaks. Each of the classes represents a type of question, this module is prepared and designed to be possible to add other classes representing different questions, using the enrichment framework.

Figure 3.4: An example of a question asked on Facebook

The DetectingMultiplePossibleNamesTask class generates a question, for users respond with the correct place of an event. This class analyzes the ontology and collects possible places of an event to create a multiple choice question on Facebook. The DetectingMissingPhotosTask class is responsible to detect events, which haven't pictures associated with them, asking Facebook users to add a picture to the event, by comment with a picture on the post associated with the question. The DetectingMissingFriendsTaks class in this module is where events which haven't persons besides the user are identified, and is asked to Facebook users to tag a friend on the post. Once the users identified a person on the post, she will be associated with the event.

In figure 3.4 is shown an example of a question generated by this module. This question was posted on Facebook and the user's friends can answer it on this social network.

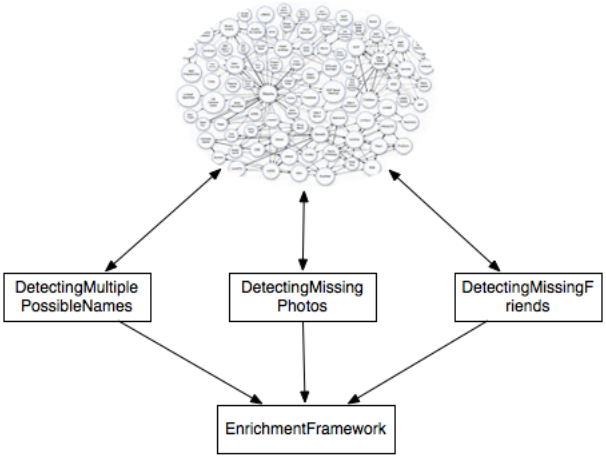

Figure 3.5: Questions generator module

3.2.3 Routine detection

The routine detection module is where user's routines are detected. This module combines the use of a semantic network to find similar events and the use of machine learning techniques to detect routines. This module is composed by two classes the SimilarEventsDetector and the RoutineDetector. The first class detects similar events within the knowledge base and save this information in the ontology. Once each events present in the ontology has his own similar events identified, the second class, the RoutineDetector will run, using a machine learning technique to detect the user's routines.

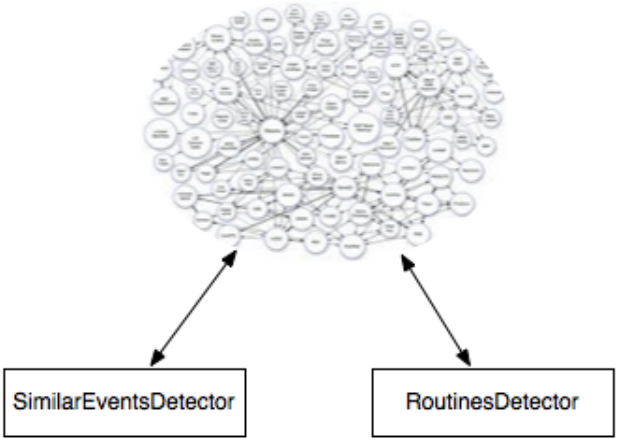

Figure 3.6: Routines detection module

Chapter 4

Semantic Network

4.1 The use of an ontology

In a first place the system was design to detect life events and display them in a visualization tool. To perform these tasks the system was projected having a relational database to store all the information. The database saves all information from the smartphone and social network. The database is fully capable to store and organize information collected, and important information to the system run. It was decided the system should allow the detection of routines and store semantic information about the events. As the database had already been implemented and as it was nuclear part of the system, it was decided to add a new layer on the system. This layer is an ontology, which allows store events and semantic information. This layer will be on top of the database and will be used as a database for events. The data used for the system run will stay on the database, because it has easier access from the java code and it's best structured on a relational database. This ontology will be used to store events, and these events can be used to detect routines from the individuals' life.

An ontology is a model that describes a particular domain. In this case the domain of the model are the events. An ontology is composed by two parts. Axioms that describe the structure of the model and a set of facts that are used to assert information. These facts use axioms to build a world for a domain. Axioms are composed by classes and properties these classes have. The facts are what the model expresses after the individuals' creation and the use of the properties applied to each individual. In this case it was proved very useful have an ontology to detect missing information. Using the ontology allows to reason over the information.

In figure 4.1 is possible to realize how the ontology will relate with the system. The data stored in the database, will be processed by the event detection module, every time an event is detected by the module, it's stored in the ontology. Then a module will run over the ontology to infer new information and to detect the individuals' routines. This module will be explained in chapter 7.

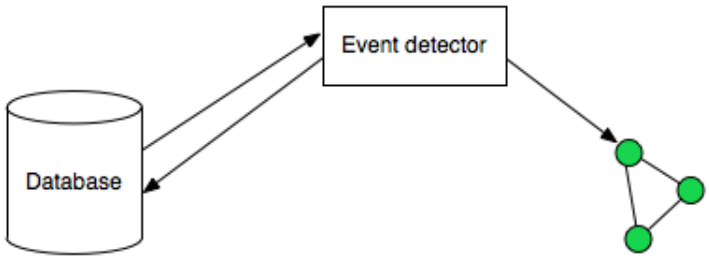

Figure 4.1: System after ontology implementation

4.2 RememberMe OWL Ontology

A knowledge base was modeled to represent the individual's life events. The knowledge base was defined using the Protege-4.3 application. The ontology uses OWL to define classes and properties. At the time of the creation of knowledge base, the goal was to answer four questions defined as the most important to represent a life event. What?, When?, Who?, Were?. To anwser these four question were four classes on the ontology: Event, Time, Person, Place. Another class named Media, was created to represent media information associated with a life event.

The interaction with the knowledge base is done through JENA, a Java framework. With this framework is possible to query and add new facts and relations to the knowledge base. The queries to the knowledge base are made using SPARQL, a query language for databases, able to retrieve data stored in Resource Description Framework.

4.2.1 Main Ontology

The concepts that are part of the knowledge base are: Events, Persons, Places, Media and Time. These five classes aim to answer the four questions identified as the most relevant to individuals' life event. A Event represents a life event on patients life, this event have participants that are represented by the class Person. A location where the event occurred is represented by the class Place. To increase the expressiveness of the class Place, Google Places categories were transformed into sub-classes of the class Place, allowing to know the type of place where the individual had gone. The time when the life event occurred is represented by the class Time. The classes Event, Person, Place and Time, have Media associated with them.

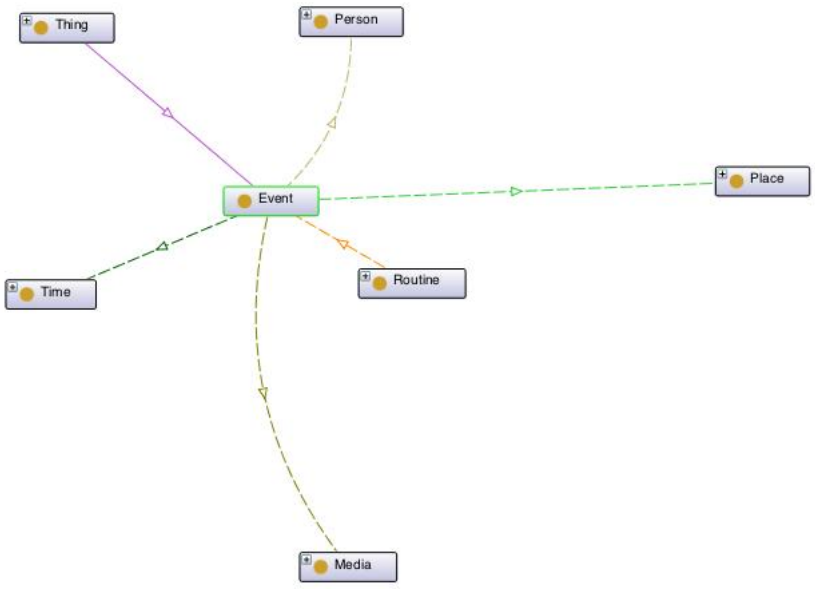

Figure 4.2: RememberMe ontology main classes

Event Class Event class is the one that represents an event, this can be considered the main class of the ontology, since all other classes are linked to this one. The class is linked to other classes by the following object properties: hasPlace, hasTime, hasPerson and hasMedia. The class has the following data properties: EventId; OntologyEventId; EventDescription. The EventId is a number type property that stores a unique ID, with the time when the event took place. The OntologyEventId is an unique ID that represents the event both on the ontology and database. This ID is created when a new event is detected. The EventDescription property is where a description of the event is stored.

Person Class represents who participated on an event. The persons are added to an event via Facebook. There are five data properties in this class, PersonClass that represents the person degree of kinship or if he/she is a close friend to the individual. This relations are obtained trough Facebook.
PersonFacebookId represents the Facebook ID of that person. The PersonID uniquely represents the person through whole system, both in the ontology as in the database. The PersonName represents the name of the person, and it's collected from the Facebook. The PersonTrustWorthiness is a number type property that represents the trust value of this person. The values vary between 0, when the person should not be trust, and 1 when

the person is very trustwhorty. This value is updated over time.

Time Class This class is used to represent the time when an event happened. This class uses the TimeProperties class to represent the weekday and the month when an event took place. The TimeProperties class has two subclasses, the Month class and the WeekDay class. These two subclasses have all the months and week days predefined by system. To relate the Time class and these two classes, two object properties hasMonth and hasWeekDay were used. To represent the exact time when the event took place the data properties BeginTime and EndTime are used to store the beginning and the ending time of an event. In order to facilitate the comparison of the time of different events, data properties were created to represent the year, the day, the hour and the minutes of the event. These properties are: BeginYear, BeginDay, BeginHour, BeginMinutes. The properties were very useful since they allow to know the events which happened in a certain hour of the day.

Media Class represents all the information collected through the enrichment mechanisms. There are four sub-classes: Audio; Photo; Text; Video. Pictures that were added to an event by Facebook will be represented in this Media class by creating and individual of the type Photo. Class Video is used to represent videos collected from Youtube or Facebook, the class Text is used to represent texts from comments made by individuals' friends on Facebook and Wikipedia articles.

4.2.2 Places Ontology

To be possible to compare the events on a semantic manner, a sub-ontology was created. A Places ontology was created to represent different types of places. The first option to create this ontology was use a database defined at http://www.geonames.org. However the use of this database won't answer the needs of the system since the goal of the Places ontology is to identify the type of place where the individual was, and the database isn't able to provide this information. As the enrichment module already uses Google Places, the next step was explore what Google Places can provide. To do it, we start explore Google Places API, to search what can be queried on Google Places API, when retrieving the nearby places of a specified latitude and longitude. The Google Places API is able to provide the place type and this was the main reason to be used as base to the Places Ontology. The ontology was created using the different places categories, available on Google Places API. The first step was gather all possible places, obtained after doing a query on Google Places API. Once all categories were gathered, the next step was to create general categories for the categories extracted from the Google Places API. Six general categories were created after this step, Amusement Establishment; Establishment; Place

of Worship; Private Place; Public Services; Transit Station. These categories have subcategories to obtain a higher level of specification.

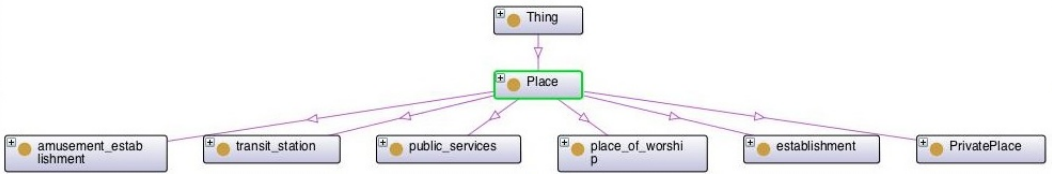

Figure 4.3: Places ontology main classes

In total sixty five leaf classes were created, and with these leaf classes and their upper classes the events can be compared semantically. With this ontology is possible to know if the individual been in two similar places, even if they are distant from each other.

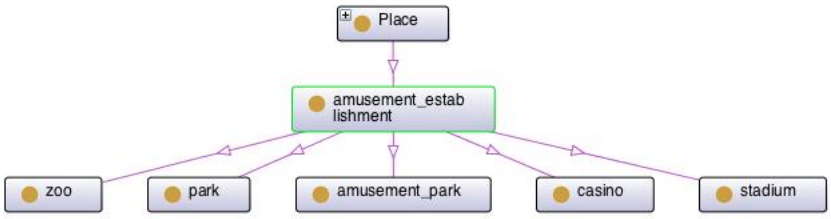

Figure 4.4: Amusement Establishment sub classes

Figure 4.3 shows the six main subclasses of the class Place. The Amusement Establishment class is the class that represents all the places where the individual can spend time to amuse himself.

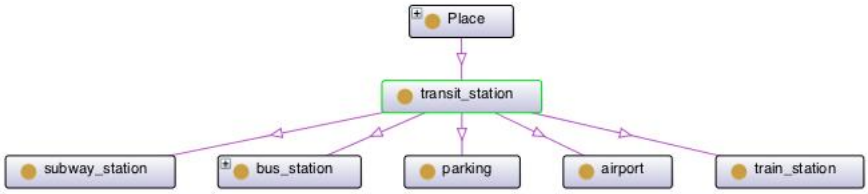

Figure 4.5: Transit Station sub classes

Transit Station class represents places like bus stations and other places where the individual can take a public transport, or places where the individual can leave his car. The classes that represents the places where the user can take a public transport are subway station, bus station, airport and train station. The parking class represents car parks.

The public services class represents places of a public service. These public services can be a post office, a political place like city hall or a embassy, a lawyer office, a courthouse or a police station.

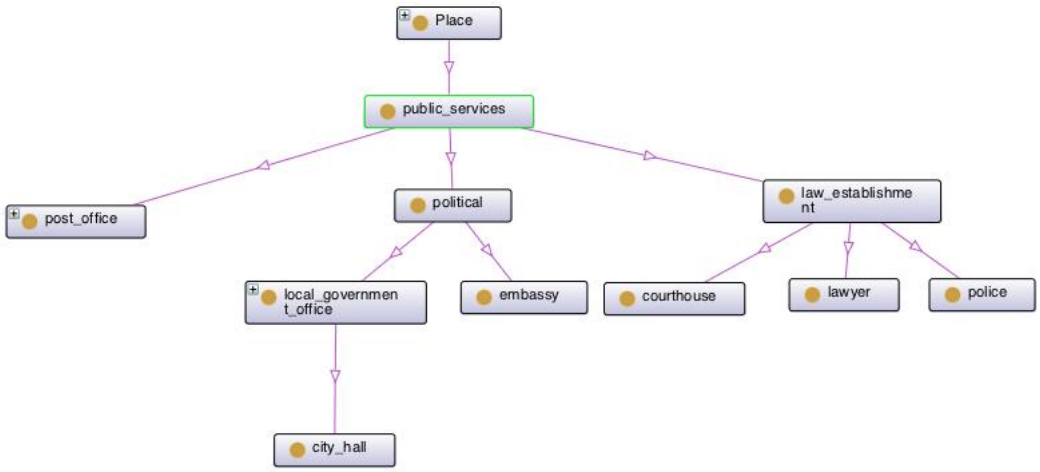

Figure 4.6: Public Services sub classes

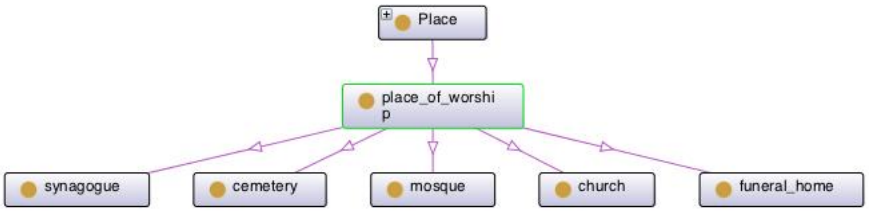

Figure 4.7: Place of Worship sub classes

The place of worship class is the class where religious places are represented. A synagogue, a Cemetery, a mosque, a church and a funeral home are the sub classes of the worship class.

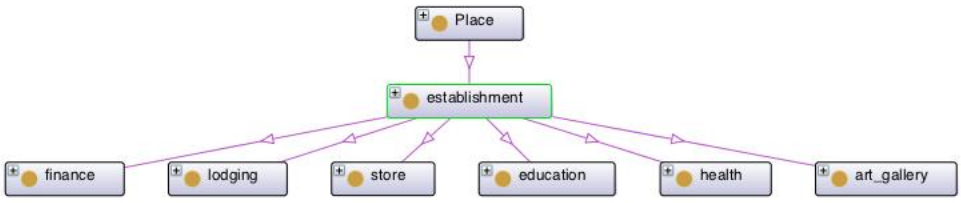

Figure 4.8: Establishment sub classes

Establishment class is one of the most relevant sub classes of the class Place. This sub class has sub classes representing different types of places. These sub classes can represent places like, finance establishments as banks and ATM machines, lodging places such as bars or nightclubs. The store subclass represents all types of stores available on Google Places API. It can represent auto stores, like car dealers, car repair, car rental and

car wash stores. It can also include bicycle stores. Clothing stores, convenience stores, electronics stores, florists, home goods stores, insurance agencies, pet stores, real estate agencies, book stores and shopping malls can all be represented on store class. Stores related with food and drink are also represented by the class food, a sub class of the store class. In this class are represented bakeries, bars, coffees, groceries or supermarkets, liquor stores and restaurants.

Private place class represents places that aren't public and aren't part of the Google Places API. This class is used to represent places like individual's home or friends' home. Places that aren't recognized by Google Places API are also represented by this class. It's known that a big number of this ontology's categories won't be used in this system, but was decided that should be created a general ontology for all the places available on the Google Places API, since this can be very useful for future projects.

4.3 Create questions using the semantic network

As explained in motivation section, we aim to unburden caretakers with the responsibility to collect all the information used in reminiscence therapy sessions. To help them with this task, we collect the data automatically from the smartphone and transform it into events. Once these events are detected they aren't ready to be shown yet, since they are incomplete. To help caretakers collecting information about events, a module which creates questions on Facebook, was created.

This module uses the ontology to identify events that have missing information. The missing information can be the exact place where the event occurred, the persons that were in the event or missing media contents to the events. In this module, each Java class is associated with a type of missing information. Were created three classes to detect missing information, one to identify when the exact place of the event is missing, one to find events which haven't people associated with them and one to identify events which haven't media. The fundamentals for each class are the same and because of that is very easy to extend this module to get other types of information about the events. Each class uses a SPARQL query to find a specific type of missing information. Each class is responsible for a single type of missing information. This decision made decrease the complexity of the queries.

The set of events returned by each query represents the events that have missing information, and questions will be created for these events. These questions are created using the enrichment framework, that allows create different types of questions, such as open questions, multiple choice questions, questions to add a picture and questions to tag a person. The question to identify the exact place of an event uses the multiple choice

question type, and the options are the possible places collected using Google Places API. When photos are missing in a event, a question to add photos is created on Facebook. And finally the class which detects missing persons on events, uses the question type that allows Facebook users tag someone on a photo. We use this type of question because is easier to identify persons on a event if there is a photo associated with.

Chapter 5

Detecting Life Events

To be possible to reconstruct users' days and create a diary of these days, it's need detect life events of the users. The detection of these events is the main goal of the project, since they are the main input source of the visualization tool and Facebook. It's with these events that caregivers and relatives are able to reconstruct individual's days. These are the events that Facebook friends will be able to enrich with photos and commentaries. In order to get life events to feed the visualization tool and Facebook, it was necessary to find a solution to detect events based on data provided by the smartphone application. The data provided by the smartphone contains latitude, longitude and time, and it was based on this data that life events were detected. The logical and simplest way to detect events with this data, was consider the amount of time that the individual spends in a place.
It's possible to retrieve information from the wide range of smartphones sensors. It's possible to collect ambient sounds and accelerometer data. However the data that proved to be more useful besides the data from the GPS was the data from photos taken by the individual. This data is used to detect events using the metadata of the photos.

5.1 Locations Classifier

The first approach to detect an individual life events was to consider the amount of time spent on a certain area. This was the simplest way to detect an occurrence of an event. To detect these life events was created a Java class, that analyzes instances on the database that correspond to locations of the individual. This Java class was able to detect if the individual stayed in the same position for more than ten minutes. If this was detected, a life event is created into the database and also in the ontology. However this method has the problem to become very complex and if it was need to do changes in the algorithm, these changes would be hard to do and very time consuming. It was proved that the method works on the data collected, however with the disadvantages mentioned we follow a different path.
The definition of a life event may be different for different people, with this in mind we

asked ourselves how can we build a system that adapts to each user. The answer for this question was the use of classifiers. Classifiers are used in the field of machine learning. A classifier is a mechanism that can learn from data and create a model to be used in different data. We used the ability of the classifiers to learn, to tackle the problem. The classifiers used for detection of events, based on locations and pictures taken by the individual were trained with labeled data. In a real use case scenario, the classifiers will be trained with data from the individual itself. The user will use a smartphone application to tag data, when he considers to be in a life event. This application will have the same features of the application that will collect the data in a real life scenario. This allows that the classifier learns what user considers to be a life event. In future the classifier will use this data as a model to predict if user is or not in a life event. The higher the volume of trained data, higher will be the accuracy of the classifier.

The process of event detection using classifiers was separated into different stages, a first stage of training, followed by a stage where data is collected. The third stage is where the classifier learns based on data collected on training stage. The final stages are the classification stage and normalization stage.

5.1.1 Algorithm choice

One of the first issues with data mining is to choose the best algorithm to each task. The first step was explore algorithms available on Weka package. An ARFF file was from the data collected in training stage. The algorithms available on the trees package were tested using a cross-validation mechanism. The cross-validation mechanism has a parameter that's the number of folds. This mechanism divides the test set in a n number of folds, and uses one of these folds as a training set. After the training set be created, the algorithm is tested with each one of the other folds. The results are then shown with a percentage of "Correctly Classified Instances". The chosen algorithm was the one with the higher percentage of correctly classified instances. In the case of classifying instances with GPS coordinates to detect life events, the chosen algorithm was the RandomForest.

Algorithm	Result
DecisionStump	86.6341
HoeffdingTree	87.4472
J48	88.813
RandomForest	90.8293
RandomTree	85.8211
REPTree	89.4309

Table 5.1: Results of the algorithm testing using WEKA

Table 5.1 shows that all algorithms had good results but the one with higher percentage of Correctly Classified Instances was the RandomForest.

5.1.2 Training

The training stage is where the classifier creates a model of what user consider to be an event. This stage aims to prepare the classifier the best possible way to the future stages. On this stage the individual uses a smartphone application to train the classifier, it collects GPS data. User just needs to press a button on the application to change the type of label associated with the data being collected. When user wants to identify an event he presses a button on the application, and the data which is being collected is identified as being part of an event. When user don't want to tag instances as being part of an event, he just need to press the button again.

5.1.3 Collection of data

This stage is where data is collected by the smartphone. This stage is similar to the previous stage but the user doesn't need to tag the data. The data is sent to the database on the server, while it's being captured.

5.1.4 Building the Classifier

To build the classifier first we need to define two parameters, the data collection frequency in seconds, and the minimum time, in minutes, that a individual should remain in one place to that be considered as an event. This two parameters will be used to create an ARRF to be used as a training source for the classifier. The ARRF file created will have deltas between the latitudes and longitudes of the instances tagged in training stage. To do this, in first place was calculated the time window, correspondent to the amount of minutes to analyze. The time window is created by the following formula, where m is the time in minutes, and s is the frequency of collection of the instances: $\frac{m*60}{s}$.

Once the time window is calculated, we start building the ARRF file. The first step of the file creation is to set up all the attributes. The file have three type of attributes, two to represent deltas for latitudes and longitudes and one to represent the class. Supposing the collection frequency of instances is 15 and the minimum time to detect an event is 10, the number of instances that represents 10 minutes is 40. By following the previous example, each instance will have 40 deltas of latitude and longitude representing the distance between the instance and the previous 40 instances. The class attribute of each instance will have the tag that was attached to the instance on training stage. After the ARRF file is created and all instances are processed, the classifier is trained using the file.

5.1.5 Classification

To classify an instance, first it need to be prepared to be used on classifier. It's necessary to calculate the deltas of latitudes and longitudes of each instance. The deltas will be created

using the same process explained in the previous point. Once these deltas are calculated, the instance is processed by the classifier. The result can be zero or one. When the result is zero it means the classifier didn't considered the instance as a part of an event. Otherwise, when the result is one, it means the instance is possibly a part of an event.

5.1.6 Normalization

A problem identified during the classification, was that the results produced by the classifier aren't constants. It was detected for example, that in the middle of a set of instances classified with zero, instances classified with the value one can pop up. This lead to the question, if those instances classified as being part of an example should be considered as a single event or should be ignored? The answer was to find a intermediate solution. Because it doesn't make sense to consider a single instance an event, since each instance just represents a point in the individual's life. However those instances shouldn't be ignored. With this in mind was created a normalization algorithm for the results. It calculates the average numbers of zeros and ones in the time window. Table 5.2 shows the results of the classifier in a time window with 10 instances. If the normalization process wasn't

1	2	3	4	5	6	7	8	9	10
0	1	0	0	1	0	0	0	0	0

Table 5.2: Results of classification before normalization without event detection

used, two false events would be created, an event regarding to the instance number 2 and another regarding the instance number 5. These events don't make sense since they would represent periods of 15 seconds on the individual's day. These results happen when user slows down its pace. Table 5.2 represents an ArrayList which is in memory during the process of classification, representing the results returned by the classifier, without the normalization process. The normalization's results are represented in Table 5.2, where we can see a list without anomalies. The first row is the instance number and the second row is the classification result. The instances two and five, were classified with one, meaning that these instances belong to an event, however if we see the whole time window we can clearly see that those instances represent anomalies in the classification. These anomalies can be eliminated using the normalization algorithm. The algorithm calculates the average value in the time window, if the number of instances classified with the value one, it's above the average is detected an event. The algorithm can be configured so the event detection can be more restrictive or looser.

Another example where without the normalization process could lead to false results is shown in table 5.4. With this example we would get three events, what wouldn't make sense since the all time window should be part of the same event. With this example the first event will be from the instance number one to the instance number three. The instance

1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0

Table 5.3: Results of classification after normalization without event detection

number five would also be considered an event. And the last event of this example will be an event which starts on instance seven and ends on instance ten.

1	2	3	4	5	6	7	8	9	10
1	1	1	0	1	0	1	1	1	1

Table 5.4: Results of classification after normalization with event detection

Using the normalization algorithm in results of the table 5.5 will result in the detection of a single event. These anomalies are common when user stays in a certain area and he moves a little and stops again.

1	2	3	4	5	6	7	8	9	10
1	1	1	1	1	1	1	1	1	1

Table 5.5: Results of classification after normalization with event detection

The other case where the normalization process is very useful is in the transition of an event state to a non event state. This is, when the classifier is producing a set ones and then the user starts to walk and the classifier starts to produces a set of zeros. However the transition from one state to the other isn't totally accurate, as shown in table 5.6.

Using the normalization algorithm on the instances from table 5.6, results in a clear transition between an event state to a non event state.

5.2 Pictures Classifier

Nowadays smartphones are used by everyone to capture relevant events in their lives. If an individual goes to a birthday party and takes several pictures with his smartphone, this could be used to detect an event. Using pictures as trigger of event detection has two advantages, first it's possible to detect a new type of events and second the photos taken are already associated with an event. An example of a new type of events able to be detected using pictures, could be, when a person visits a place on outdoors, he couldn't stop, but he can take pictures, and these pictures be used to detect an event. The classifier's objective is to recognize pictures taken close to each other, both in time as in place, and group them in a single event. The figure 5.1 represents the idea behind event detection using pictures, each vertical line represents a picture taken by the user and the horizontal

1	2	3	4	5	6	7	8	9	10
1	1	1	0	1	0	0	1	0	0

Table 5.6: Results of classification before normalization in a transition between states.

1	2	3	4	5	6	7	8	9	10
1	1	1	1	1	0	0	0	0	0

Table 5.7: Results of classification after normalization in a transition between states.

represent a time line when the pictures were taken. Group photos by time and place allows to detect events. The circles around vertical lines represent a detected event based on this approach. In this case there are four events to be detected. The single horizontal line that isn't inside a circle isn't part of any event, this happens because we consider that if an event is relevant, the individual will take more than one photo.

Pictures classification process is composed by a first stage of training, where data is collected and each picture taken was tagged with an integer number, and this number represents an event. The process of training and tagging the pictures is very simple. Users just need take pictures, every time he wants to change the event associated to pictures, he just presses a button on the smartphone application that was developed for this training, which changes the event to a different number.

Photo	Event
Photo1	1
Photo2	1
Photo3	1
Photo4	1
Photo5	2
Photo6	2
Photo7	2
Photo8	3
Photo9	3

Table 5.8: Photos tagged in the training stage

In table 5.8 is represented a situation, where user had taken nine pictures on three different events. Using the method explained earlier the individual teaches the classifier how he wants represent an event. The first four photos belong to the same event. The photos five to seven belong to event number two. And photos seven and eight belong to event number three. Using this model the classifier will create a model, using the metadata on each photo, to group future photos by events.

After collecting and tagging photos, instances that represent the photo, are created to

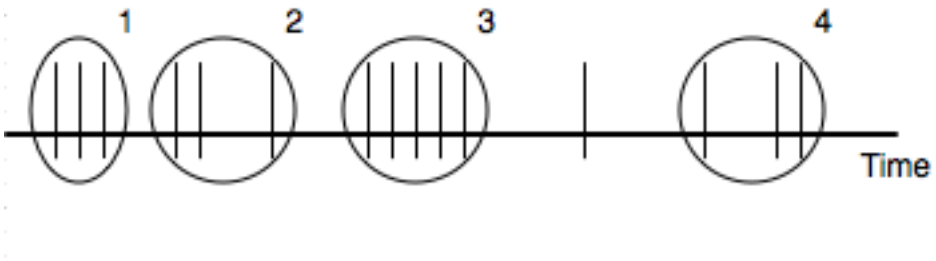

Figure 5.1: Picture Classifier

train the classifier. These instances are composed by latitude, longitude, time, accuracy value, picture name and event number.

Using the smartphone in a real life situation the individual takes pictures and these are send to the server. The module to detect events based on pictures is waiting for new pictures, and every time a new picture is sent to the server the module reads the metadata and creates an instance to be processed by the classifier.

The classification result of an instance is a numeric value which represents an event. When the classifier returns a result that is already stored in the database, it means that the instance classified represents a picture that belongs to an event which is already detected and created on the ontology. Then it's just necessary create the picture on the ontology and associate it, with the event. When the result isn't already in the database, it's necessary to create an event and associate the result number with the event recently created.

To classify the instances was used the FilteredClassifier, this classifier uses a filter and a classifier to classify the instances. This was the chosen classifier because there were some attributes, that we didn't want use in the classification. first we use the Remove filter, where the non desired attributes are removed and then, we use the AdditiveRegression classifier, this was chosen based on the success rate of the classifier.

Chapter 6

Reasoning and Routines

A routine can be seen as an activity that is performed in a regular basis, it can be diary, weekly, monthly or yearly. The detection of routines in dementia patients is useful since it can help doctors and caregivers learning patients' behaviors patterns.

The use of an ontology will give us a different type of tools to detect this patterns. Using an ontology as a store mechanism for detected events allow us to use reasoning mechanisms to detect routines in the individual activity. To try to identify a routine in the system, we start by looking at the events detected and we try to identify the common points between events. First and using the classes that belong to the ontology, we analyze the place where each event occurred. This had the goal to see if two events that happened on the same place could be part of a routine.

We tried identify simple routines in the events stored in the ontology. Getting in touch with this, we obtained clues how to identify routines using an ontology. The first type of routine identified in the ontology, was a routine which represents a place that the user visits multiple times. This was identified in the ontology when was noticed that a certain place was the place of many events.

In figure 6.1 we can see that the place P1 has many places associated with it. This was the first thing that we noticed when analyzed the ontology. But without looking to the event details, it's impossible to know if the user has gone to this place in just one day or if each one of the events happened on a different day.

The routines that we aim to detect were daily and weekly routines. The routines that are detected are what we could call as a place routine. A place routine is what we call, if the individual has the routine of going to a place on a daily or weekly basis. These are the only routines types which we are able detect, since the information stored in each event is very simple.

Analyzing each event details, in particular the temporal details, we were able to extract information that can help us to identify a routine. To detect routines, we began by explore what was the best option to detect them automatically, at first we began to explore protege 4.3 functionalities.

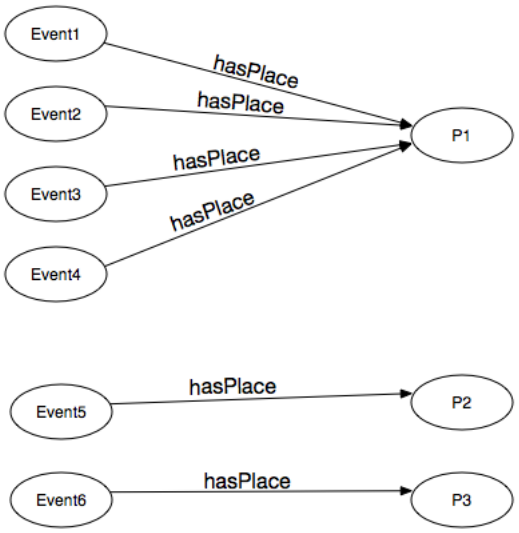

Figure 6.1: A place with many events associated

Protege 4.3 brings reasoners by default, and we started to explore how these reasoners could help us to identify individual's routines. We created rules that these reasoners can process, these rules have the objective to extend the expressiveness of the OWL language, and they are expressed in SWRL (Semantic Web Rules Language). By default protege 4.3 reasoners are FaCT ++ and the Hermit 1.3.8, and these reasoners have the capacity to process the SRWL rules.

Knowing this we created simple rules that can be processed by these reasoners, rules that express what we think a routine is. The first rules created were very simple, and they express daily routines. To detect if the individual goes to the same place every day, we created a rule that used event details to detect if there are events satisfying this rule. If the this is true, the rule is triggered when it's processed by the reasoner and then creates a routine in the ontology using events' details.

Besides this method worked using protege 4.3 application, it has a problem. Trigger these rules via Java code is tricky, and if is necessary to add or change the rules we need to do it via the protege application. To deal with these obstacles, JENA API was used, since it was already used to make the bridge between the Java code and the ontology. JENA API allows us to create our own rules and trigger them via java code, like a reasoner does in the protege.

6.1 Jena Rules

As explained earlier the use of rules to detect the individual's routines was the first approach. The problem of using SWRL rules in the protege is that this approach, brings

problems to trigger the rules using Java code. We started to explore the JENA API looking for reasoners that could solve this problem. JENA API have four types of reasoners available, the RDFS reasoner, the OWL reasoner, the Transitive reasoner and the generic purpose rule engine.

The first three reasoners aren't the type of reasoners that we wanted use, because they use properties that are expressed in the ontology to infer new knowledge. They are also very useful when the objective is to test the integrity of the ontology.

For the type of detection that we wanted to do, the reasoner that best fits to our needs was the generic purpose rule engine. This reasoner can process rules that are written in a similar language to SWRL. This was a great advantage when this reasoner was used, since we already have some of the rules created when they were used on protege. So the only necessary change was modify the syntax of these rules, in order to fit them to a language that can be processed by the reasoner.

6.1.1 Creating the rules

As some of the rules were already created on protege 4.3 we began by adapt them to the syntax of the generic purpose rule engine. In order to have a more expandable application, we opted to save the rules in a external file. This file is saved together with the Java files and when it's time to use the reasoner, it's read and the rules are loaded to the reasoner.

The process of creating each rule, it's started by identify what we want to detect, since with this mechanism is necessary to know what we are looking for. This is a disadvantage of this rules mechanism.

The rules that we created were rules to detect simple routines, as if the individual has a daily routine at a specific hour. This is, a routine where the user goes to the same place each day of the week. Other types of routines that were expressed in the rules, were rules to detect if the individual has weekly routines such as go to the same place every week. For that, rules were created to detect if the individual goes to the same place in each week of the month and if he goes exactly at the same day of the week. Seven rules were created, one for each day of the week. The Figure 6.2 shows a JENA Rule. This rule has the goal to detect a routine of a daily activity of the individual. This rule is triggered when the individual has five events, each one at the same place and each one of them happened at the same hour but on different days of the week.

As is possible to see, the complexity of these rules would increase rapidly if we wanted to detect more complex patterns in the individuals' events. The incapacity to expand these rules to more complex patterns and the impossibility to detect patterns that are unknown for us, was the reason that lead us to explore different paths to detect routines.

To achieve that, we used these rules just as a first approach to detect simple routines in people activities, and we used the power of classifiers to detect unknown patterns in people activities. The use of classifiers to detect routines will be explained later.

```
[hasDailyHourRoutine:
(?e1 rdf:type RMO:Event),(?e2 rdf:type RMO:Event), (?e3 rdf:type RMO:Event),
(?e4 rdf:type RMO:Event),(?e5 rdf:type RMO:Event), (?e1 RMO:hasPlace ?p),(?e2 RMO:hasPlace ?p),
(?e3 RMO:hasPlace ?p),(?e4 RMO:hasPlace ?p),(?e5 RMO:hasPlace ?p), (?e1 RMO:hasTime ?t1),
(?e2 RMO:hasTime ?t2), (?e3 RMO:hasTime ?t3), (?e4 RMO:hasTime ?t4), (?e5 RMO:hasTime ?t5),
(?t1 RMO:BeginHour ?bh), (?t2 RMO:BeginHour ?bh), (?t3 RMO:BeginHour ?bh),
(?t4 RMO:BeginHour ?bh), (?t5 RMO:BeginHour ?bh), (?t1 RMO:hasWeekDay ?dw1),
(?t2 RMO:hasWeekDay ?dw2), (?t3 RMO:hasWeekDay ?dw3), (?t4 RMO:hasWeekDay ?dw4),
(?t5 RMO:hasWeekDay ?dw5) -> (?p RMO:hasRoutine ?bh),
(?e1 RMO:isPartOfRoutine RMO:dailyRoutine),(?e2 RMO:isPartOfRoutine RMO:dailyRoutine),
(?e3 RMO:isPartOfRoutine RMO:dailyRoutine), (?e4 RMO:isPartOfRoutine RMO:dailyRoutine),
(?e5 RMO:isPartOfRoutine RMO:dailyRoutine), (RMO:dailyRoutine RMO:RoutineHasEvent ?e1),
(RMO:dailyRoutine RMO:RoutineHasEvent ?e2), (RMO:dailyRoutine RMO:RoutineHasEvent ?e3),
(RMO:dailyRoutine RMO:RoutineHasEvent ?e4), (RMO:dailyRoutine RMO:RoutineHasEvent ?e5)]
```

Figure 6.2: Example of a JENA rule

6.1.2 The use of the rules

As it was told before, JENA rules were written in an external file, to be easier to change and expand them in the future. When it's time to run the reasoner, the rules are loaded to the reasoner in running time, and then the reasoner use this rules to infer new knowledge in the ontology. This process can be seen in Figure 6.3.

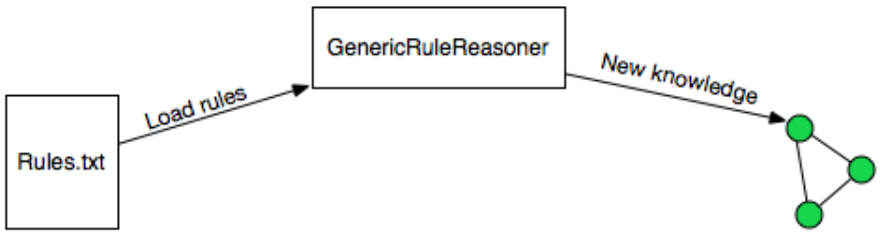

Figure 6.3: Loading and reasoning over the rules

6.2 Routine Classifier

The use of classifiers to detect routines arises from the necessity, as already explained, to detect unexpected routines in users' activities. As rules haven't the power to detect an unexpected routine, the use of classifiers to detect them is a possible solution.

The classifier allows the detection of these routines, and the complexity level won't increase when is necessary find more complex routines, since the classifier processes the events by it self and detect the routines. The process of detecting routines using classifiers was different to the one that we used to detect the events. We have the events in the

ontology, but we needed extract them from there and relate them, so we can use them in the classifier.

The process of detecting routines using classifier was tricky because it was necessary to find a way to relate each one of the events and find attributes that best fit to the classification. The main idea was compare each two events and see if they belong to the same routine. We defined that a routine should be composed by multiple events, and a routine should have the temporal information of the order of the events. In figure 6.4 is possible to see the idea of the concept of routine that we aimed to achieve.

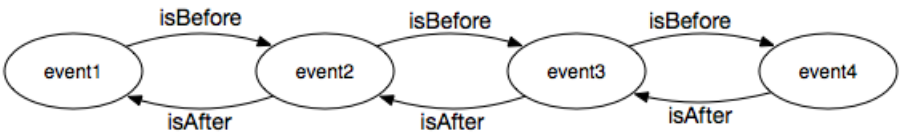

Figure 6.4: Routine Concept

We can see that a routine is composed by a set of events and each of them is linked to the previous and the next by the relation isBefore and isAfter. By doing this we tried to create a chain which represents a routine. In order to try to do this we create the relations isBefore and isAfter in the ontology and then we manually created the routines. These routines aren't expressed in the ontology as a class, but instead they are represented by these relations. Because when events are linked between them by these relations we can infer that these events belong to the same routine.

When we tried to implement this concept, we found that we had a problem, because each routine can have a different size, and this would bring a problem to the classifier. Since every routine can have a different size, it was impossible to create a training mechanism with the routines, because every instance to be processed by the classifier must have the same size.

Our goal was to create instances each representing a routine with the events that compose it. As this didn't worked we try find another solution. The solution found was to compare two events, and then give the information, if they belong to the same routine or don't, to the classifier. To do this, we crossed every two events in the ontology and marked them as being part of the same routine or don't. The objective of doing this, was to create a model using the classifier, that the system could use it on a real life situation, and detect if the two events are part of a routine. And by doing this, we sought to create the chain that represents a routine.

After a process of training the classifier, where each two events that we considered to be part of the same routine were marked, we tested this solution. However the results were disappointing. When used in a real life situation, the classifier didn't produced the results that we expected it would produce, and it didn't found routines in the individual's events. Because of this, it was necessary to find a different solution.

As it was already stated in previous attempts, we knew it was necessary a mechanism to compare events, so they can be related between them and later be analyzed to detect the routines.

Use the places ontology was a vital part of event comparison process. We create a mechanism, to compare events and then create an ARRF file for each event. These ARRF files will be used by WEKA to find the routines. So the process of detecting routines is composed by a first process of events comparison and a second process of clustering, which it's used to find a routine in similar events. In Figure 6.5 is shown how this process is done. The events are read from the ontology, they are compared and an ARFF file is created with the similar events of each event. These files are processed by WEKA and routines are extracted and then saved in the ontology.

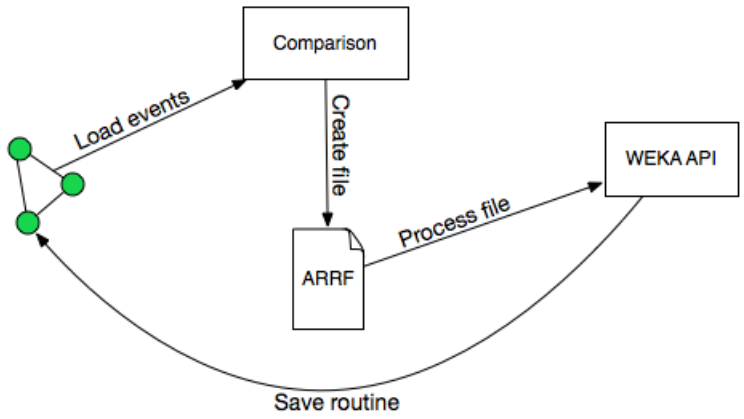

Figure 6.5: Finding Routine process

6.2.1 Events Comparison

The event comparison process starts by reading events from the ontology and compare them in pairs, beginning by calculate a value for each event. The calculation of this value is made using details of each event, like temporal and location details. Participants' details aren't used in the calculation because the ontology of persons isn't implemented, however in future when relations between the individual and the participants of an event are implemented, these details can also be used in calculation.

To calculate the difference between the places of two events, first we calculate the difference between its coordinates, by subtracting the latitude and the longitude of one event with the latitude and the longitude of the other. But as we use a places ontology, we can do better than that, and we can compare the similarity between the places, using the type of a place. If both places have the same type, then the difference between the places type doesn't exist and the value associated with this detail is zero. If the places have different types, we use the ontology to find the first common node between the two places. By doing this we can know how different two places are, just by knowing their types.

The value of the difference between two places type is calculated by measuring the distance between two nodes. This can be seen in the figure 6.6 where the distance between the place type University and the place type Library is two. This value is the number of edges necessary to go from one node to the other. In case of we calculate the difference between an University and a Bus Station this value will be five.

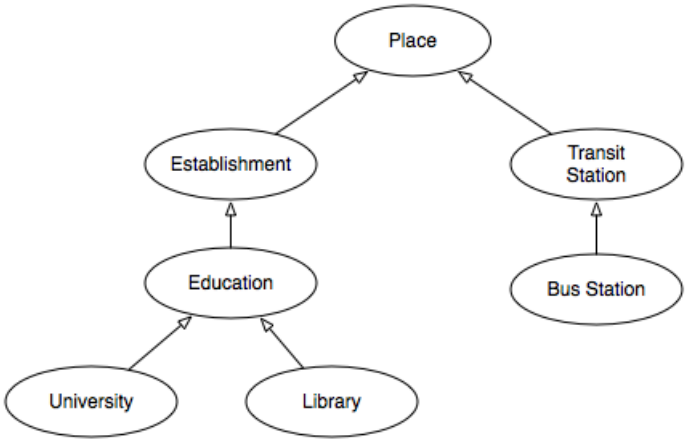

Figure 6.6: Calculating the difference between places types

This calculation is done using the JENA API. The process is simple, it starts-up by getting all upper classes of the two places types which we want compare. When obtained, the upper classes are automatically inserted into two lists, each list represents the path from the root to the place type we want compare. In our example if we want compare the University and the Bus Station we will have two lists. The list that represents the path from the root to the University will be the following: [University, Education, Establishment, Place]. The list that represents the path from the root to the Bus Station will be like: [Bus Station, Transit Station, Place].

Once we have these two list, it's time to find the first common node between them. The process of finding the first common node between the two lists is an iteration between both lists, it starts by taking the first element of the first list and compare it with the all the

elements of the second list, if there isn't no match, then it takes the second element of the first list and tries to find a match in the second list. This is done until a match is found.

In the worst case scenario the common node will be the Place class which is the root node. When the common node is found, we calculate the distance between the two types, by calculating the distance between the beginning of the lists to the common node and then sum-up the two values. Using this process we get the distance of the two places types.

Using this value and the difference between the coordinates of the two places, we calculate the difference between two places, a value that we called the places' delta value. The formula to calculate the value is the following: $placeTypeValue * 0.8 + latitudeSimilarityValue * 0.1 + longitudeSimilarityValue * 0.1$

The weight associated to the difference between place types is 80% and the weight associated with the difference between the coordinates is 20%, equally divided between latitude and longitude.

The other part of the formula that is used to calculate the similarity between events is what we called the time's delta value. This value represents the difference between the time of the two events. We use the temporal details of each event to calculate this value.

Each time have details associated with it, like the month, the week day, the day and the hours. The calculation of this value uses a number associated with each month to calculate the difference between the months, and the same method is used to calculate the difference between the weekdays.

The calculation of the difference between the values associated with each month and week day is circular. It's necessary, since when it's necessary to compare the difference between a Sunday and a Saturday if we didn't calculate using this method, we will have a difference of six, because the Sunday has the value of one and the Saturday has a value of seven. But actually the difference between the Sunday and the Saturday isn't this, because the two days are really close. So in order to do this we use the circular method that will give us a difference of one.

When comparing the weekdays and weekend days, we can insert a special value of difference because the weekend can be seen as two specials days that have their own types of events. In weekends the individual probably does some types of events that doesn't during the week. The rest of the calculation of the time's delta is done by calculating the difference between the days and the hours of the two events. The formula of the calculation of the time's delta value is the following: $monthDifference * 0.1 + weekDayDifference * 0.3 + dayDifference * 0.2 + hourDifference * 0.4$

The attributes that have higher weight are the weekdays and the hours, with 30% and 40% respectively. This means that when two events have close hours and have close weekdays, they tend to be more similar. The value of the weight can be changed to achieve

better results. The types of events can change from individual to individual and if we have that information we can tune this formula to get a better accuracy.

After the time's and place's delta be calculated, they are used to calculate the difference between the two events. A weight of 50% was assigned for each delta. Once the calculation is made, it's time to see the similarity between the events, if the result of this calculation is lower than a predefined value then the two events are considered similar and the details of both events are saved on a ARFF file. This ARRF file will have the name of an event, and inside will be saved all the details of the similar events.

6.2.2 Events Clustering

When all the files are created, they are processed using the WEKA API. The WEKA API allows us to use clusters, these clusters were useful to detect the common points between similar events. We used a cluster to extract the common points between events and with that we identified a routine within the similar events.

We use the SimpleKMeans cluster that is available in WEKA package. In the options of the cluster, it was specified that we wanted that the result of the clustering was divided into two groups, these two groups will be created based on an algorithm that uses the euclidean distance. One group will have all the events with the higher number of common points, and the other group, that will be discarded, will have all the other events that don't belong to the first. After the group with all the common events is created, we wanted to extract vital information of this group which will allow us to extract the routine from these events.

Each group created by the cluster have the same attributes of the events that were analyzed, like the month, the weekday, the day and the hour in the case of temporal details, and also have the place details which include the place type and the coordinates where the event had occurred. The values associated with each of this attributes are called cluster centroids.

The cluster centroids represent the center point of the values of all the group members. In case of nominal attributes the value will be the most common value, in case of numeric attributes it will be the average. These cluster centroids was what we used to create the routines in the ontology. Instead of creating routines using a chronological approach, where we need to specify the order of the events, we use an approach that extracts the information of a set of events and then, we use that information to create a routine on the ontology. This routine will have the information, that the individual has the habit of going to a certain type of place on a certain day of the week at a certain hour.

The routines detected don't follow the approach that we identified in the beginning of the chapter. With the routines detected we can't identify a pattern in the individual's life,

like the following example: The individual goes to drink a coffee every morning and then he goes to the work.

The methodology that we use does not allows to extract this directly, but as the detected routines are saved in the ontology, in the future it's possible to extract this information from the saved routines. This is possible to do, by analyzing the saved routines, and using temporal and places details is possible to extract the patterns of the individual's life using the routines nodes that we identified.

Chapter 7

User Evaluation

In order to validate the methods applied, we run some tests on the application. These tests had the goal to explore fails of the system and also to validate our approach. The process of testing, had the goal to test the detection of life events and the detection of routines. To do the test, we installed the mobile application on participants' smartphones, so they could collect the data of theirs days. The testing process of life event detection and detection of routines, uses the same data collected by the individuals.

7.1 Detecting Life Events

7.1.1 Procedure

The process of testing life event detection began by installing the application in the smartphones of the participants. We performed the tests during one week in order to get information about every day of the week.

Then the participants were instructed to use their smartphone as they used to, and to do their normal activities. Was asked to participants to do two tasks on end of each day. The first was to send the text file, to an email that was provided to them, the second task was to fill a form, where they describe their day. In this form it was asked them to indicate the day they are about to describe and then to create a diary of their day.

In the description, we asked them to do it in a chronological order, so the interviews at the end of each day can be simpler. Using a chronological order, was more easy for them relive their days. We didn't wanted to impose any other restrictions in the process of filling the form because we wanted to see how users see their days and if we created many rules in the filling of the form, the descriptions might be too much formatted, and we didn't wanted that. Also we had the care to create a form that is simple and fast to fill, because we didn't wanted to bother the participants, since they have to fill a form in the end of each day, and if this form is hard to fill, the participants might become unmotivated.

The text file which the participants were asked to send via email, was created and filled by the smartphone's application. The text file is saved in the SD card of each smartphone

and it will contain all the data collected by users. Asking the participants to send their files at end of each day, we also guarantee the backup of the data. At the end of the week, we have a file for each day of the week and we can then ask the participants if the events detected were what they really had done and if we find events that they didn't reported.

At the end of each day data of each participant is processed in separated. This processing will result on an ontology with the events of the day. Then this ontology is saved together with the data of the day. At end of each day we also merged the files sent by the user. The objective of this merging is to have a single file with all the data of a participant at the end of the week.

At the end of the week this global file will be processed and will be used to compare if the events detected on a file of a single day are also included in the events detected using the file with all the data. And also to see if there is events that are detected in the processing of the global file that didn't appear in the processing of the data of the days. In the end of the test week the participants were interviewed one by one to confront them with the results obtained.

7.1.2 Goals

The goal of this process of testing was to check if the life event detection algorithms, are able to detect the individual's life events. We also have other objective, which it's find if the life event detection algorithm detects events that the individuals didn't consider relevant. This is, events which the individuals didn't write down in their daily diaries.

7.1.3 Participants and Material

In total six participants were used in the study, each participant used his own smartphone to run the mobile application. The universe of the population had range of ages between the 23 and the 55 years old. Each one of the participants used its own smartphone and all of them use smartphones on a regular basis and they are familiar with the use of the applications in the smartphone. Also all of them have a regular access to Internet to send the files at the end of each day and to fill the forms. The participants' profile was chosen to be like this so the participants don't have problems to perform their tasks. However two of the participants had problems during the week, so their data was considered inadequate to be used in the study.

7.1.4 Results

To analyze the events extraction module efficiency, we wanted to know how much events the module can detect and how much events the module isn't able to detect. This will help us to identify the types of events which aren't detected and help us in the search for a possible solution.

The first analysis that was done to the process of extracting life events, was to see the amount of detected events from the ones which were reported by the participants.

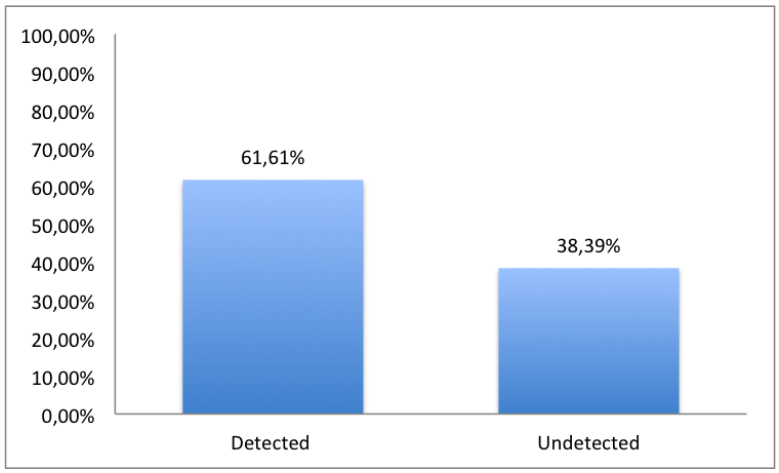

Figure 7.1: Percentage of events detected and undetected

In figure 7.1 is possible to see relation between detected events and undetected events. From a total of 113 events reported by the participants 69 of them were detected and 43 were undetected. These numbers suggests that the module isn't detecting a sufficient amount of events to represent users' days. However it was noticed that some participants didn't used the smartphone application in some periods of the day and in some cases the application stopped the collection of data. This made that the data collected isn't a totally correct representation of the participants' days. A factor which increases the number of undetected events is the fact that the participants when were writing their diaries, didn't take into account if the data they collected really represented the day they were writing down.

But the efficiency of the module cannot only be measured by this comparison. To analyze how this module is useful to extract information which is able to represent the users' days is necessary to see how the events detected by the module are seen by the participants. To do this, a list of detected events was shown to each user, and they with the help of Google Maps, to see the exact place, confirmed if that was a real event or not.

During the process of confirmation the events, the participants stated that some events happened at the time identified by the module but the place isn't exactly the one identified in the map. However the users were able to identify the correct place in the surrounding area of the point identified in the map. This incorrect place identification by the module, happened because GPS technology isn't 100% accurate and we need to take into account the surrounding area of a place.

In figure 7.2 is possible to see how this module is accurate when detecting users' events. From a total 120 detected events just 16 weren't confirmed by the participants.

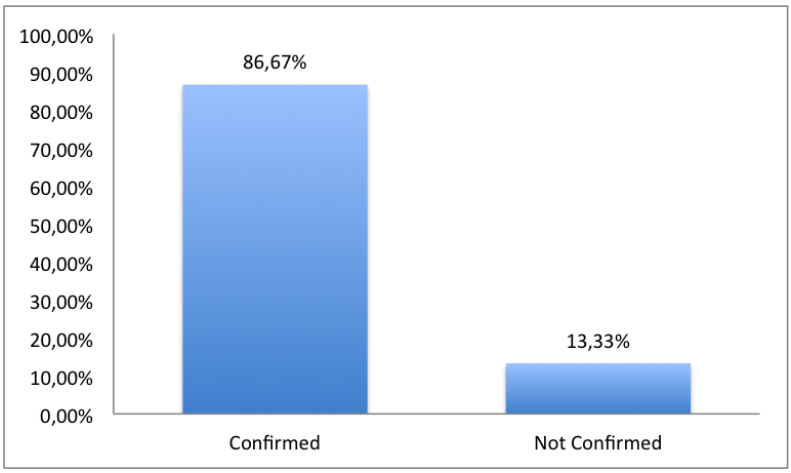

Figure 7.2: Percentage of confirmed events by the participants

The participants stated that using the events detected by the module it was possible to represent their days. These numbers proved that the module is able to detect and extract useful information from raw data and it can be used to represent users' days.

To see how the events detected can be used to represent users' days, we will show how the events of participant number one represents its day, with high accuracy. Using the events of 16th of July, we can rebuild the user's day, and see that he was at Faculty of Science from 7:47 am to 12:05 pm. Then he lunched from 12:15 pm to 12:39 pm, near the bus station of University City. It was possible to identify an event from 12:52 pm to 13:11 pm, this event was confirmed by the participant, and he stated that the event was when he was at the Faculty of Science bar, drinking a coffee. After that, he gone to the room where he spends most of his time working. He was in this room from 13:13 pm to 18:54 pm. And finally with the last event detected, we were able to identify that he arrived home at 19:20 pm.

A different type of information that is useful to extract from the results is how much events, which were detected by the module and confirmed by the user as real life events, but that weren't reported by the users in their diaries. This type of analysis will help us understand what are the types of events that users consider relevant to report.

In figure 7.3 we can see that 28% of the detected events weren't reported by the participants. This percentage represents the number of unexpected events by the participants. Events which they didn't consider relevant when writing the diaries but that they confirmed when confronted with them.

An analysis that is possible to do is to analyze the results of each participant. This type of analysis is useful to extract information how the detection of events works with different types of users.

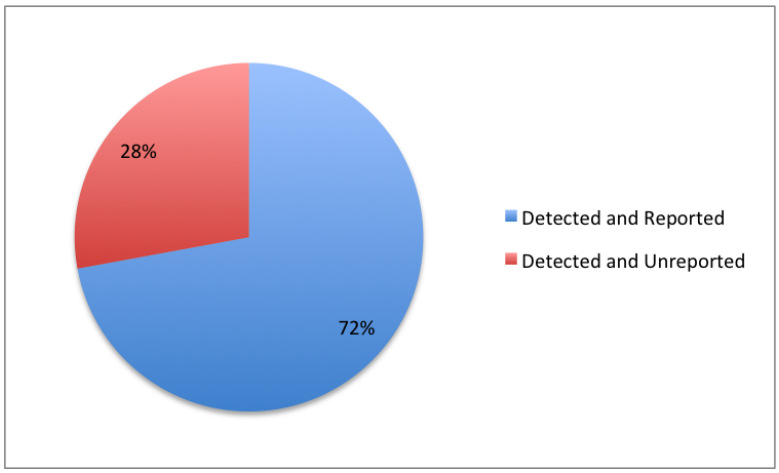

Figure 7.3: Events detected and reported vs Events detected and unreported

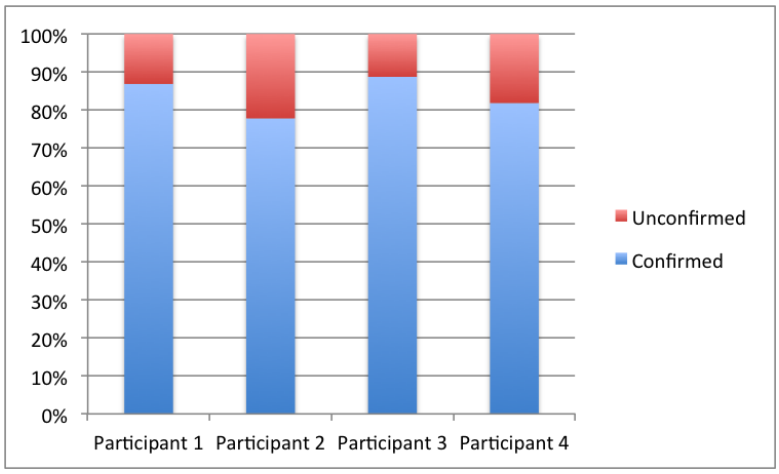

Figure 7.4: Percentage of confirmed events by participant

The figure 7.4 shows the percentages of confirmed events by the participants in the study. As we can see, participant one and participant three were the ones which have the higher percentage of confirmed events. The participant three which was the one with the higher value of confirmed events with a value near 90%, stated that the events that he confirmed could be used to create an accurate representation of his day.

The participant one and the participant three were also the most active participants in the study. These two participants had a big difference of detected events compared with the participants two and four. This lead us to the conclusion that the detection of events is more accurate when the participants are more active.

The participant two was the one with the lower confirmation of events. However, after the interview with this participant, this result was expected since the participant told us that in some cases he didn't used the smartphone application correctly. In the interview

he said that the smartphone application sometimes had stopped working and he didn't noticed that. He also said that sometimes in the morning he forgot to start the application and just started it after he was remembered by the leader of the study. This contributed to a lower percentage of confirmed events, because the events detected with this participant weren't such accurate as the events from the others participants.

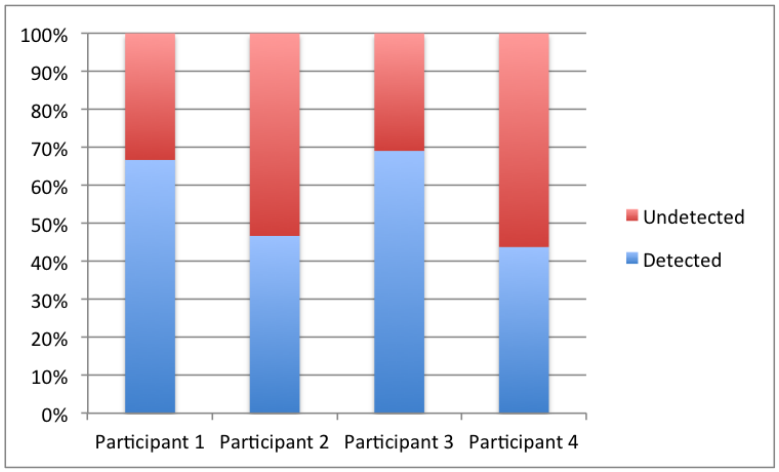

Figure 7.5: Percentage of detected events by participant

The other type of individual analysis that is possible to do with each individual's results is to compare the percentage of detected events for each participant. As in the previous analysis the participants with higher results are the participant three followed closely by the participant one. Was noticed that most of the undetected events are close to each other, this suggest that the GPS coordinates weren't accurate at that time or some kind of problem with collection of data happened. A common undetected event, was the time when the participants were waiting for the bus. They reported this as an event but they didn't wait almost no time for the bus, so the system wasn't able to detect that as an event.

The participant two as explained earlier, said in the interview that sometimes his application wasn't working correctly. And this was one of the reasons to the lower percentage of detected events in this participant. This happened because besides the application wasn't working correctly he didn't noticed that, and wrote on his diary all the events without take into account that the application wasn't collecting data.

The participant four also experimented some problems with the application. He noticed that sometimes the application wasn't writing the collected data, and that lead to a higher value of undetected events since the data to detect those events wasn't saved on the file of the day.

7.2 Reasoning and Routines

7.2.1 Procedure

The process of testing the routines uses the same data that was used in the previous test. At the end of the test week when all the data was collected and all the events were detected, it was time to test the detection of the routines. This test was done by loading the ontology, which result from the processing of the global data, to the system and then run the system in order to find the routines of each participant.

The results of this testing are saved on the ontology, like they would be done on a real life situation. After these results have been saved in the ontology, the participants were confronted to see if the results that we had obtained are a routine in their lives.

7.2.2 Goals

The objective in this test is obviously, see if the mechanism of routines detection is working and if it detects the right routines of the participants. The second but no less important objective is to see if the routines that were detected are the same routines that the individual identified in the interview at the end of the week.

7.2.3 Results

A total of nine routines were detected in all participants, the routines identified were useful to find patterns in participants' lives. Although the routines identified aren't totally accurate, they allow us to understand common actions in the participants' lives. Routines like lunching at same time every days and arriving home around the same hour each day of the week are examples of routines identified. The participants were confronted with the routines and they used Google Maps to help them identifying the place where they have a routine.

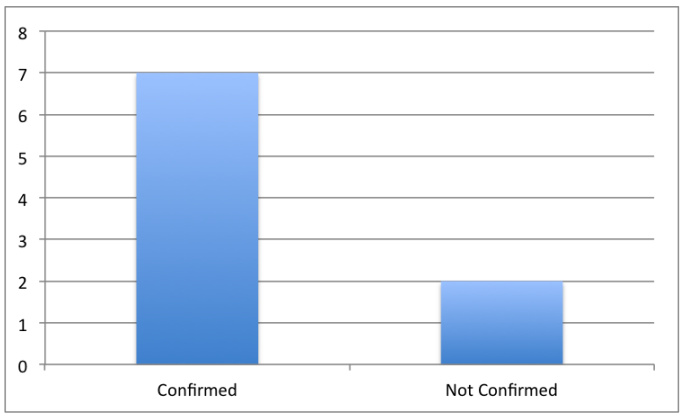

Figure 7.6: Number of confirmed routines

Using the Google Maps, they confirmed a total of seven routines. The places identified by the routines were all them correct but some participants stated that the hours on the routines could be more accurate. After search for what could be a possible problem, we found that the method to compare events using the ontology should be improved. The conclusion that we got was that this method isn't strict enough and because of that the events had many similarities between them. Doing some adjusts in the formula's of events comparison should lead us to better results in the events' comparison, leading to a better and more accurate detection of routines.

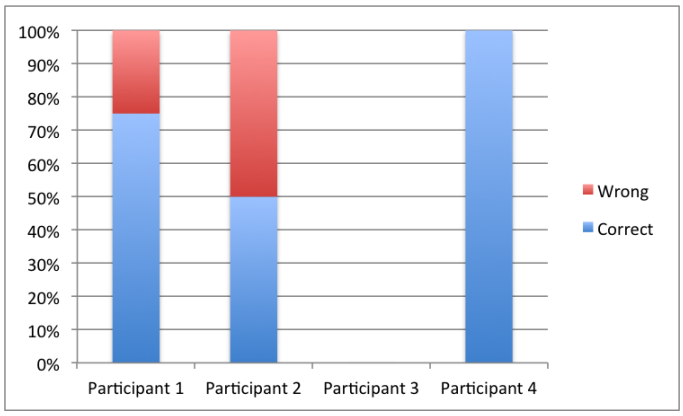

Figure 7.7: Percentage of correct routines

In figure 7.7 are shown the percentages of correct routines for each participant in the study. And it's possible to see that the most of routines identified were correct.

Analysing the routines of all participants, we can clearly see that the participant three didn't had any routine detected. This was a expected result, since this was the participant with higher number different types of events. When we look for his events there isn't no routine that we can identify.

The routines identified in participant four were all confirmed, this happened because this was the participant which had the same type of events on each day of the week of study. The days of this participant were, going to the university, lunching near there, go back to the university and then return home. Since the types of events were very similar each day, the system was able to identify routines, that were all correct.

To conclude these results, we can see that the system was able to identify correct routines on a user, where his days' activities were very similar to each others, like the case of the participant four. And also that the system didn't detect any routines in the case of a user which activities were very different from day to day.

Chapter 8

Conclusion

The RememberMe project started with the motivation, of create a system, which allows dementia patients to collect data automatically, to be used on reminiscence tools and to unburden caretakers with responsibility of collecting that data. The work presented in this document represents a part of the whole project RememberMe.

An ontology was created to represent events on a user's life. This ontology was built with the goal to store life events and to be used in the future as a mechanism which allows to find unexpected information about the users events. This ontology includes a sub-ontology within, the places' ontology, created using the types of places available in the Google Places API. This sub-ontology was very useful to find similarities between the different places which user had visited.

It was presented the data processing module, a module where data is processed and transformed into human readable information. This transformation is done by detecting life events using GPS coordinates and pictures taken by system users. The detection using machine learning techniques proven to be useful to detect life events.

In the review of the work done in this area was noticed that many approaches to detect life events use these techniques, but instead of using classifiers some approaches use clusters. The use of clusters was taken into account to detect life events based on GPS coordinates, but as the system was built with the objective to be a real time system, where events are detected while they occur in real life, the use of clusters became more difficult. A reason to this, was because we use the WEKA package, an open platform of machine learning, and the use of clusters in this platform implies that we had to have all the data and had to know the number of cluster to create, before we could do the processing.

Using a classifier instead of a cluster, we were able to classify the data on the fly, each time a location was sent to the server, it's classified and it's checked if it belonged to an event or not.

The routine detecting module was created with the goal to detect patterns in users' days. The first approach was try detect these patterns directly, using machine learning techniques. While we were trying to find the best solution to this problem, we found that a better approach to this problem was to divide it into two phases. The first stage, the one which was developed in this project, was to detect places where the users usually go and when they are on these places. This was achieved through the use of the semantic network to find similar events, and then use a cluster, from the WEKA package, to find the common points of these events and extract the nodes of a routine. The second stage the one to be built in future work, is to create diary routines using the nodes detected on the first stage of this process.

The presented approach was tested and evaluated on a study which included four participants. The results of this study showed that the system was able to find events reported by the participants, as well as detect events didn't reported by the participants.

These unreported events were confirmed by the participants, when they saw the time and place of the events. To help participants confirm the events, we showed the locations of these events in Google Maps. The accuracy of this approach can be increased, if the classifiers were trained with the data of each participant, since this would allow the creation of a model for each participant. Another point that was noticed was that all the participants used different devices, and the accuracy of the GPS coordinates wasn't the same for all participants, what made the trained model more accurate in some participants than in others.

8.1 Future Work

In order to improve the project we identified future work for the project rememberMe as well for each module of the work here presented. There are work to do in each of the modules that compose the cognitive prosthesis.

The event detection module should be update with more classifiers in order to detect different types of events. These new classifiers will help the system identify new events and with that have a greater understanding of the types of events in the users' lives. The classifiers that should be the next implementations of this module should be a classifier to identify events based on data collected from the smartphone microphone and a classifier which uses data collected from the accelerometer of the smartphone. This classifier should be use to detect the body state of the users. It can be used to identify if the person is walking and also can be used to falling detection.

The classifier to detect events based on data from the microphone can be trained with a conversation between the user and a specific person. Once the classifier is trained, it can

be tested to detect if it detects when the user is talking with the person used in the train.

Once these new classifiers are implemented, should be created a mechanism that uses the information of all classifiers to create a higher level classifier. This classifier will be a hierarchical classifier which uses the information from each one of the implemented classifiers and extracts a very high level of information about the users' days.

The questions generator module should follow the strategy of create new types of questions on Facebook. These new types of questions should be used to extract different types of information from the social network.

The new questions to be generated should use the information extracted with the new classifiers. An example of a situation where a question should be interesting to be posted on Facebook is the following: Using the hierarchical classifier, the system detected that the individual was walking on the street and when he stopped he was talking with some one. A question on Facebook asking for the person, who talked with the individual, is an interesting scenario for a new type of information to be collected from the Facebook. However the conversation isn't recorded and if the person don't want to be identified her privacy is secured.

The detection routines module, must be improved in order to obtain a better accuracy when detecting routines. This improvements should start by the algorithm already created. This algorithm should be updated so the nodes of the routines are more precise.

Once this algorithm is improved, the next step should be create a mechanism which is able to transform all the nodes identified, on a routine, like a diary routine.

The project rememberMe itself should evolve to an application that can be used by specialists, so they can track the behavior of the patients, using a simple and fast interface.

If the specialists could use this system as a tool to see the behavior of the patients it will reduce the time that they spend collecting the information from caretakers.

Also it can be used to detect activities that trigger unexpected behaviors in patients. If a patient has a unexpected behavior every time he perform an activity detected by the system, it will be easier to the specialists detect the trigger of these stressful situations.

Appendix A

Tables with the events detected and reported by the participants

Event Description	Confirmation	Report
15th of July From 10:57 to 18:55 at Faculdade de Ciencias, Universidade de Lisboa	C	DR
15th of July From 19:06 to 19:07 at Bus Station Campo Grande Metro	C	DU
15th of July From 19:11 to 19:22 at Bus Station Campo Grande Metro	C	DR
15th to 16th of July From 19:38 to 07:43 at Private Place (38.7974051,-9.1643273)	C	DR
16th of July From 07:47 to 12:05 at Private Place (38.755819,-9.1578476)	C	DR
16th of July From 12:15 to 12:39 at Bus Station Cidade Universitaria	C	DR
16th of July From 12:52 to 13:11 at Faculdade de Ciencias, Universidade de Lisboa	C	DU
16th of July From 13:13 to 17:19 at Faculdade de Ciencias, Universidade de Lisboa	C	DR
16th of July From 17:24 to 18:54 at Faculdade de Ciencias, Universidade de Lisboa	C	DU
16th to 17th of July From 19:20 to 07:42 at Private Place (38.7974051, -9.1643273)	C	DR
17th of July From 07:44 to 18:37 at Private Place (38.755819,-9.1578476)	C	DR
17th of July From 18:41 to 18:45 at Bus Station Campo Grande	C	DR
17th to 18th of July From 19:01 to 07:36 Private Place (38.7974051, -9.1643273)	C	DR
18th of July From 07:56 to 12:32 at Faculdade de Ciencias, Universidade de Lisboa	C	DR

18th of July From 12:46 to 18:30 at Faculdade de Ciencias, Universidade de Lisboa	C	DR
18th of July From 18:33 to 18:46 at Bus Station Campo Grande Metro	C	DR
18th of July From 18:51 to 18:42 at Private Place (38.794361,-9.1689162)	NC	DU
18th of July From 18:54 to 18:54 at Private Place (38.7974051,-9.1643273)	C	DR
18th of July From 18:54 to 18:55 at Grocery or Supermarket mini-mercado Quinta da Ponte	NC	DU
18th to 20th of July From 18:57 to 12:52 at Private Place (38.7974051,-9.1643273)	C	DR
20th of July From 13:10 to 13:12 at Subway station Cidade Universitaria	PC	DU
20th of July From 13:22 to 13:25 at Unidentified Place	PC	DU
20th of July From 13:41 to 15:12 at Shopping Mall Vasco da Gama Center	C	DR
20th of July From 15:20 to 15:26 at Train Station Lisboa Oriente	PC	DR
20th of July From 15:34 to 15:51 at Unidentified Place	PC	DR
20th of July From 16:22 to 16:22 at Unidentified Place	PC	DU
20th of July From 16:22 to 16:23 at Unidentified Place	PC	DU
20th of July From 16:23 to 17:30 at Unidentified Place	NC	DU
20th of July From 17:36 to 17:46 at Subway station Campo Grande	C	DR
20th to 21st of July From 17:56 to 07:46 at Private Place (38.7974051, -9.1643273)	C	DR
21st of July From 07:46 to 18:51 at Bus Station Campo Grande Metro	NC	DU
21st of July From 18:52 to 18:53 at Private Place (38.7974051, -9.1643273)	C	DR
21st to 22nd of July From 18:53 to 09:49 at Unidentified Place	PC	DR
22nd of July From 10:05 to 17:29 at Faculdade de Ciencias, Universidade de Lisboa	C	DR
22nd of July From 17:43 to 17:44 at Subway Station Campo Pequeno	C	DR
22nd of July From 17:45 to 19:54 at Unidentified Place	PC	DU

22nd of July From 20:05 to 20:05 at Bus Station Campo Grande Metro	C	DR
22nd of July From 20:06 to 20:12 at Unidentified Place	NC	DU

Table A.1: Participant 1 detected events

Event	**Detection**
15th of July at 11:00 at Faculdade de Ciencias, Universidade de Lisboa	D
15th of July at 12:00 at Canteen	UD
15th of July at 13:00 at Faculdade de Ciencias, Universidade de Lisboa	D
15th of July at 19:20 at Bus Station Campo Grande	D
15th of July at 19:45 at Home	D
16th of July at 08:00 at Faculdade de Ciencias, Universidade de Lisboa	D
16th of July at 12:00 at Canteen	D
16th of July at 13:00 at Faculdade de Ciencias, Universidade de Lisboa	D
16th of July at 17:20 at Faculdade de Ciencias C8, Universidade de Lisboa	D
16th of July at 19:00 at Bus Station Campo Grande	UD
16th of July at 19:30 at Home	D
17th of July at 08:00 at Faculdade de Ciencias, Universidade de Lisboa	D
17th of July at 12:00 at Canteen	UD
17th of July at 13:00 at Faculdade de Ciencias, Universidade de Lisboa	D
17th of July at 18:40 at Bus Station Campo Grande	D
17th of July at 19:00 at Home	D
18th of July at 08:00 at Faculdade de Ciencias, Universidade de Lisboa	D
18th of July at 12:00 at Canteen	UD
18th of July at 13:00 at Faculdade de Ciencias, Universidade de Lisboa	D
18th of July at 18:40 at Bus Station Campo Grande	D
18th of July at 19:00 at Home	D
20th of July at 12:50 at Subway Sation Odivelas	UD
20th of July at 13:30 at Shopping Mall Vasco da Gama Center	D
20th of July at 15:30 at Moscavide	D
20th of July at 16:45 at Subway Station Oriente	D

20th of July at 17:15 at Bus Station Campo Grande	D
20th of July at 17:30 at Home	D
21st of July at 08:00 at Faculdade de Ciencias, Universidade de Lisboa	UD
21st of July at 12:45 Lunch at Alvalaxia	UD
21st of July at 13:30 at Faculdade de Ciencias, Universidade de Lisboa	UD
21st of July at 18:40 at Bus Station Campo Grande	UD
21st of July at 19:00 at Home	D
22nd of July at 09:20 at Subway Station Odivelas	UD
22nd of July at 09:40 at Faculdade de Ciencias, Universidade de Lisboa	D
22nd of July at 12:15 at Canteen of Faculdade de Ciencias, Universidade de Lisboa	UD
22nd of July at 13:00 at Faculdade de Ciencias, Universidade de Lisboa	D
22nd of July at 18:40 at Sapo Picoas	UD
22nd of July at 20:00 at Bus Station Campo Grande	D
22nd of July at 20:20 at Home	UD

Table A.2: Participant 1 reported events

event description	**Confirmation**	**Report**
15th of July From 14:07 to 16:04 at Faculdade de Ciencias Universidade de Lisboa	C	DR
15th of July From 16:26 to 16:44 at Bus Station Campo Grande	C	DR
15th to 17th of July From 17:14 to 18:37 at Private Place (38.7960129 -9.1204573)	NC	DU
17th of July From 18:47 to 18:59 at Private Place (38.7531568 -9.1490212)	PC	DR
17th to 18th of July From 19:42 to 12:02 at Private Place (38.7934322 -9.1195837)	NC	DU
18th of July From 12:10 to 12:35 at Private Place (38.7511615 -9.160117)	C	DR
18th of July From 12:50 to 18:33 at Faculdade de Ciencias Universidade de Lisboa	C	DR
18th of July From 18:53 to 18:55 at Bus Station EncarnaĂ§ĂŁo Bombeiros	C	DU
18th to 22nd of July From 19:30 to 08:43 at Private Place (38.7934322 -9.1195837)	C	DR

Table A.3: Participant 2 detected events

Event reported	Detection
15th of July at 08:00 at Faculdade de Ciencias, Universidade de Lisboa	UD
15th of July at 12:00 at Canteen	UD
15th of July at 12:30 at Faculdade de Ciencias, Universidade de Lisboa	D
15th of July at 16:30 at Bus Station	D
15th of July at 17:15 at Home	D
17th of July at 08:00 at Faculdade de Ciencias, Universidade de Lisboa	UD
17th of July at 12:00 at Canteen	UD
17th of July at 12:30 at Faculdade de Ciencias, Universidade de Lisboa	UD
17th of July at 18:45 at Bus Station	UD
17th of July at 19:45 at Home	D
18th of July at 08:00 at Faculdade de Ciencias, Universidade de Lisboa	UD
18th of July at 12:00 at Canteen	D
18th of July at 12:30 at Faculdade de Ciencias, Universidade de Lisboa	D
18th of July at 18:30 at Bus Station	UD
18th of July at 19:20 at Home	D

Table A.4: Participant 2 reported events

Event description	Confirmation	Report
15th of July From 11:05 to 11:12 at Faculdade de Ciencias, Universidade de Lisboa	C	DR
15th of July From 14:36 to 15:26 at Faculdade de Ciencias, Universidade de Lisboa	C	DR
15th of July From 15:28 to 23:21 at Faculdade de Ciencias, Universidade de Lisboa	NC	DU
15th of July From 23:33 to 23:36 at Private Place (38.759488, -9.2482577)	C	DR
15th to 16th of July From 23:42 to 08:31 at Private Place (38.7619697, -9.2493368)	C	DR
16th of July From 08:31 to 08:32 at Private Place (38.7611293, -9.24892)	PC	DU
16th of July From 08:38 to 08:40 at Private Place(38.7608051,-9.2459355)	PC	DU

16th of July From 08:40 to 08:43 at Private Place(38.7626701,-9.2394935)	PC	DU
16th of July From 08:47 to 09:47 at Private Place(38.7844186,-9.22653745)	C	DR
16th of July From 10:09 to 19:00 at Faculdade de Ciencias, Universidade de Lisboa	C	DR
16th to 17th of July From 19:26 to 12:30 at Private Place(38.7608051,-9.2459355)	NC	DU
17th of July From 12:44 to 13:13 at Faculdade de Ciencias, Universidade de Lisboa	C	DU
17th of July From 13:20 to 16:22 at Faculdade de Ciencias, Universidade de Lisboa	C	DR
17th of July From 16:25 to 19:02 at Faculdade de Ciencias, Universidade de Lisboa	C	DR
17th of July From 19:03 to 19:03 at Private Place (38.7566077,-9.1944161)	NC	DU
17th of July From 19:03 to 19:12 at Private Place (38.7590785,-9.1956436)	PC	DU
17th of July From 19:18 to 21:20 at Private Place (38.7608051,-9.2459355)	C	DR
17th of July From 21:22 to 21:27 at Private Place (38.7608051,-9.2459355)	C	DU
17th of July From 21:29 to 21:38 at Private Place (38.7619697,-9.2493368)	PC	DU
17th of July From 21:39 to 21:39 at Private Place (38.759488,-9.2482577)	PC	DU
17th of July From 21:47 to 22:00 at Private Place (38.759488,-9.2482577)	PC	DU
17th to 18th of July From 22:14 to 12:52 at Private Place (38.759488,-9.2482577)	C	DR
18th of July From 12:55 to 13:05 at Private Place (38.7608051,-9.2459355)	PC	DU
18th of July From 17:04 to 17:19 at Private Place (38.7231989,-9.1524322)	C	DR
18th of July From 17:21 to 17:22 at Private Place (38.733355,-9.145317)	C	DR
18th of July From 17:22 to 17:34 at Private Place (38.733355,-9.145317)	C	DR
18th of July From 17:36 to 17:36 at Private Place (38.7341099,-9.1541786)	PC	DR
18th of July From 17:37 to 19:18 at Private Place (38.735616,-9.1554185)	C	DR
18th of July From 19:20 to 20:00 at Private Place (38.7274167,-9.2110859)	C	DR

18th to 19th of July From 20:12 to 10:40 at Private Place (38.7274167,-9.2110859)	NC	DU
19th of July From 10:46 to 12:44 at Private Place (38.7828232,-9.2226486)	C	DR
19th of July From 13:10 to 13:10 at Private Place (38.7608051,-9.2459355)	PC	DU
19th of July From 15:03 to 15:58 at Private Place (38.759488,-9.2482577)	C	DR
19th of July From 16:13 to 16:22 at Private Place (38.7822619,-9.222963)	PC	DU
19th of July From 16:53 to 17:02 at Private Place (38.6906463,-9.3158814)	C	DR
19th of July From 17:15 to 17:16 at Private Place (38.6995996,-9.2485255)	NC	DU
19th of July From 18:03 to 18:12 at Private Place (38.7075792,-9.1334972)	C	DR
19th of July From 18:24 to 18:49 at Private Place (38.7054755,-9.1452463)	C	DR
19th of July From 19:18 to 19:20 at Private Place (38.6906463,-9.3158814)	C	DR
19th of July From 20:19 to 20:46 at Private Place (38.7619697,-9.2493368)	PC	DR
19th of July From 21:07 to 22:08 at Private Place (38.8012566,-9.3104406)	C	DU
19th of July From 22:13 to 22:19 at Private Place (38.8012566,-9.3104406)	PC	DU
20th of July From 12:05 to 14:29 at Private Place (38.4448441,-9.1005235)	C	DR
20th of July From 14:47 to 16:50 at Private Place (38.4448441,-9.1005235)	C	DR
20th of July From 17:20 to 17:27 at Private Place (38.6261606,-9.1193779)	C	DR
20th of July From 17:40 to 17:50 at Private Place (38.669391,-9.1739833)	NC	DU
20th of July From 18:04 to 18:04 at Private Place (38.7608051,-9.2459355)	PC	DU
20th of July From 18:06 to 19:15 at Private Place (38.7608051,-9.2459355)	C	DR
20th of July From 19:29 to 19:39 at Private Place (38.723526,-9.1485796)	C	DU
20th of July From 19:47 to 20:27 at Shopping Mall Tivoli Forum	C	DR
21st of July From 15:06 to 19:58 at Faculdade de Ciencias, Universidade de Lisboa	NC	DU

21st of July From 20:42 to 21:10 at Private Place (38.7619697,-9.2493368)	C	DR
21st of July From 21:26 to 22:10 at Private Place (38.7608051,-9.2459355)	PC	DU
21st of July From 22:35 to 23:08 at Private Place (38.7557989,-9.2435794)	PC	DU
21st to 22nd of July From 23:40 to 14:01 at Private Place (38.7619697,-9.2493368)	C	DR
22nd of July From 14:52 to 16:54 at Private Place (38.7575688,-9.2433395)	C	DR
22nd of July From 17:23 to 17:30 at Private Place (38.7828232,-9.2226486)	PC	DR
22nd of July From 17:48 to 18:04 at Private Place (38.7846604,-9.2216209)	C	DR
22nd of July From 18:13 to 19:19 at Private Place (38.7828232,-9.2226486)	PC	DR
22nd of July From 19:45 to 20:01 at Shopping Mall Babilonia	C	DR
22nd of July From 20:11 to 20:18 at Private Place (38.7619697,-9.2493368)	C	DR
22nd of July From 20:25 to 21:08 at Private Place (38.7619697,-9.2493368)	C	DR

Table A.5: Participant 3 detected events

Event reported	**Detection**
15th of July at 10:00 at Faculdade de Ciencias, Universidade de Lisboa	D
15th of July at 12:00 at Canteen	UD
15th of July at 13:00 at Faculdade de Ciencias, Universidade de Lisboa	D
15th of July at 16:00 at Faculdade de Ciencias,C5, Universidade de Lisboa	UD
15th of July at 19:00 at Home	UD
16th of July at 09:00 at Vale Grande	D
16th of July at 10:10 at Faculdade de Ciencias, Universidade de Lisboa	D
16th of July at 12:00 at Canteen	UD
16th of July at 13:00 at Faculdade de Ciencias,C5, Universidade de Lisboa	D
16th of July at 16:00 at Faculdade de Ciencias,C5, Universidade de Lisboa	D

16th of July at 19:10 at Home	D
16th of July at 20:30 at Shopping Mall Dolce Vita Tejo	UD
16th of July at 23:00 at Home	UD
17th of July at 10:00 at Faculdade de Ciencias, Universidade de Lisboa	UD
17th of July at 12:00 at Canteen	UD
17th of July at 12:50 at Faculdade de Ciencias, Universidade de Lisboa	D
17th of July at 19:20 at Home	D
18th of July at 16:45 at Marques de Pombal	D
18th of July at 17:15 at Saldanha	D
18th of July at 17:40 at El corte Ingles	D
18th of July at 19:25 at Decathlon	D
18th of July at 20:00 at Restaurant	D
18th of July at 23:00 at Home	UD
19th of July at 10:30 at Vale Grande	D
19th of July at 15:45 at Vale Grande	D
19th of July at 16:30 at Oeiras	D
19th of July at 18:30 at Santa Apolonia	D
19th of July at 19:30 at Oeiras	D
19th of July at 20:00 at Home	D
19th of July at 20:45 at Restaurant at Tala	D
19th of July at 22:30 at Home	UD
20th of July at 12:00 at Sesimbra	D
20th of July at 16:45 at Amora	D
20th of July at 17:30 at Home	D
20th of July at 20:00 at Movie Theater	D
20th of July at 01:30 at Home	UD
21st of July at 14:30 at Oeiras	UD
21st of July at 15:00 at Faculdade de Ciencias, Universidade de Lisboa	D
21st of July at 18:20 at Home	UD
22nd of July at 17:30 at Unspecified Place	D
22nd of July at 19:45 at Chinese Restaurant near Shopping Mall Babilonia	D
22nd of July at 20:00 at Home	D

Table A.6: Participant 3 reported events

Event description	**Confirmation**	**Report**
15th of July From 11:27 to 14:55 at Faculdade de Ciencias, Universidade de Lisboa	C	DR

15th of July From 14:55 to 18:15 at Private Place (38.7558096,-9.1577713)	C	DR
15th of July From 18:21 to 18:28 at Faculdade de Ciencias, Universidade de Lisboa	C	DU
15th to 16th of July From 18:29 to 12:06 at Private Place (38.7558096,-9.1577713)	NC	DU
16th of July From 12:15 to 12:41 at Bus Station Cidade Universitaria	PC	DR
16th of July From 12:51 to 14:37 at Faculdade de Ciencias, Universidade de Lisboa	C	DR
16th of July From 14:38 to 14:42 at Faculdade de Ciencias, Universidade de Lisboa	C	DU
17th of July From 12:16 to 12:40 at Bus Statio Cidade Universitaria	PC	DR
17th of July From 12:45 to 16:13 at Faculdade de Ciencias, Universidade de Lisboa	C	DR
17th to 18th of July From 16:18 to 10:25 at Private Place (38.7550154,-9.1535574)	NC	DU
18th of July From 10:25 to 17:42 at Faculdade de Ciencias, Universidade de Lisboa	C	DR

Table A.7: Participant 4 detected events

Event reported	**Detection**
15th of July at 10:00 at Faculdade de Ciencias, Universidade de Lisboa	D
15th of July at 12:00 at Canteen	UD
15th of July at 13:00 at Faculdade de Ciencias, Universidade de Lisboa	D
15th of July at 19:30 at Home	UD
16th of July at 10:15 at Faculdade de Ciencias, Universidade de Lisboa	UD
16th of July at 12:00 at Canteen	D
16th of July at 13:00 at Faculdade de Ciencias, Universidade de Lisboa	D
16th of July at 18:40 at Home	UD
17th of July at 09:35 at Faculdade de Ciencias, Universidade de Lisboa	UD
17th of July at 12:00 at Canteen	D
17th of July at 13:00 at Faculdade de Ciencias, Universidade de Lisboa	D
17th of July at 17:15 at Home	UD

18th of July at 10:15 at Faculdade de Ciencias, Universidade de Lisboa	D
18th of July at 12:00 at Canteen	UD
18th of July at 12:45 at Faculdade de Ciencias, Universidade de Lisboa	UD
18th of July at 18:35 at Home	UD

Table A.8: Participant 4 reported events

Bibliography

[1] 7 stages of alzheimer's & symptoms | alzheimer's association. URL http://www.alz.org/alzheimers_disease_stages_of_alzheimers.asp.

[2] Dementia – signs, symptoms, causes, tests, treatment, care | alz.org. URL http://www.alz.org.

[3] M Al Masum Shaikh, Md Khademul Islam Molla, and Keikichi Hirose. Automatic life-logging: A novel approach to sense real-world activities by environmental sound cues and common sense. In *Computer and Information Technology, 2008. ICCIT 2008. 11th International Conference on*, pages 294–299. IEEE, 2008.

[4] Ville Antila, Jussi Polet, Arttu Lamsa, and Jussi Liikka. Routinemaker: Towards end-user automation of daily routines using smartphones. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, pages 399–402. IEEE, 2012.

[5] Preetha Appan, Hari Sundaram, and David Birchfield. Communicating everyday experiences. In *Proceedings of the 1st ACM Workshop on Story Representation, Mechanism and Context*, SRMC '04, pages 17–24, New York, NY, USA, 2004. ACM. ISBN 1-58113-931-4. doi: 10.1145/1026633.1026638. URL http://doi.acm.org/10.1145/1026633.1026638.

[6] Ling Bao and Stephen S Intille. Activity recognition from user-annotated acceleration data. In *Pervasive computing*, pages 1–17. Springer, 2004.

[7] Alberto J Cañas, Greg Hill, Roger Carff, Niranjan Suri, James Lott, Tom Eskridge, Gloria Gómez, Mario Arroyo, and Rodrigo Carvajal. Cmaptools: A knowledge modeling and sharing environment. In *Concept maps: Theory, methodology, technology. Proceedings of the first international conference on concept mapping*, volume 1, pages 125–133, 2004.

[8] Ciarán O Conaire, Noel E O'Connor, Alan F Smeaton, and Gareth JF Jones. Organising a daily visual diary using multifeature clustering. In *Electronic Imaging 2007*, pages 65060C–65060C. International Society for Optics and Photonics, 2007.

[9] Emily A Cook. Effects of reminiscence on life satisfaction of elderly female nursing home residents. *Health care for women international*, 19(2):109–118, 1998.

[10] Jacqueline B Cook. Reminiscing: how it can help confused nursing home residents. *Social Casework*, 1984.

[11] Stephen Davies, Scotty Allen, Jon Raphaelson, Emil Meng, Jake Engleman, Roger King, and Clayton Lewis. Popcorn: the personal knowledge base. In *Proceedings of the 6th conference on Designing Interactive systems*, pages 150–159. ACM, 2006.

[12] Aiden R Doherty, Niamh Caprani, Ciarán Ó Conaire, Vaiva Kalnikaite, Cathal Gurrin, Alan F Smeaton, and Noel E O'Connor. Passively recognising human activities through lifelogging. *Computers in Human Behavior*, 27(5):1948–1958, 2011.

[13] Katayoun Farrahi and Daniel Gatica-Perez. Daily routine classification from mobile phone data. In *Machine Learning for Multimodal Interaction*, pages 173–184. Springer, 2008.

[14] Laura Ferrari, Marco Mamei, and Franco Zambonelli. "all-about" diaries: Concepts and experiences. In *Proceedings of the 5th International Conference on Communication System Software and Middleware*, COMSWARE '11, pages 1:1–1:11, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0560-0. doi: 10.1145/2016551.2016552. URL `http://doi.acm.org/10.1145/2016551.2016552`.

[15] Steve Hodges, Lyndsay Williams, Emma Berry, Shahram Izadi, James Srinivasan, Alex Butler, Gavin Smyth, Narinder Kapur, and Ken Wood. Sensecam: A retrospective memory aid. In *Proceedings of the 8th International Conference on Ubiquitous Computing*, UbiComp'06, pages 177–193, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-39634-9, 978-3-540-39634-5. doi: 10.1007/11853565_11. URL `http://dx.doi.org/10.1007/11853565_11`.

[16] Yu-Jin Hong, Ig-Jae Kim, Sang Chul Ahn, and Hyoung-Gon Kim. Activity recognition using wearable sensors for elder care. In *Future Generation Communication and Networking, 2008. FGCN'08. Second International Conference on*, volume 2, pages 302–305. IEEE, 2008.

[17] Tâm Huynh, Mario Fritz, and Bernt Schiele. Discovery of activity patterns using topic models. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 10–19. ACM, 2008.

[18] Julia Kasl-Godley and Margaret Gatz. Psychosocial interventions for individuals with dementia: an integration of theory, therapy, and a clinical understanding of dementia. *Clinical psychology review*, 20(6):755–782, 2000.

[19] Basel Kikhia, Josef Hallberg, Kåre Synnes, and Zu Sani. Context-aware life-logging for persons with mild dementia. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pages 6183–6186. IEEE, 2009.

[20] Pil Ho Kim and Fausto Giunchiglia. The open platform for personal lifelogging: The elifelog architecture. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, pages 1677–1682, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1952-2. doi: 10.1145/2468356.2468656. URL `http://doi.acm.org/10.1145/2468356.2468656`.

[21] Steve Lauriks, Annika Reinersmann, Henriëtte Geralde Van der Roest, FJM Meiland, Richard J Davies, Ferial Moelaert, Maurice D Mulvenna, Chris D Nugent, and Rose-Marie Dröes. Review of ict-based services for identified unmet needs in people with dementia. *Ageing research reviews*, 6(3):223–246, 2007.

[22] Matthew L. Lee and Anind K. Dey. Lifelogging memory appliance for people with episodic memory impairment. In Hee Yong Youn and We-Duke Cho, editors, *UbiComp*, volume 344 of *ACM International Conference Proceeding Series*, pages 44–53. ACM, 2008. ISBN 978-1-60558-136-1. URL `http://dblp.uni-trier.de/db/conf/huc/ubicomp2008.html#LeeD08`.

[23] Rebecca G Logsdon, Laura E Gibbons, Susan M McCurry, and Linda Teri. Quality of life in alzheimer's disease: patient and caregiver reports. *Journal of Mental Health and Aging*, 1999.

[24] Mitja Luštrek and Boštjan Kaluža. Fall detection and activity recognition with machine learning. *Informatica*, 33(2):197–204, 2009.

[25] João Martins, José Carilho, Oliver Schnell, Carlos Duarte, Francisco M. Couto, Luís Carriço, and Tiago Guerreiro. Friendsourcing the unmet needs of people with dementia. In *Proceedings of the 11th Web for All Conference*, W4A '14, pages 35:1–35:4, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2651-3. doi: 10.1145/2596695.2596716. URL `http://doi.acm.org/10.1145/2596695.2596716`.

[26] Kim Anh Pham Ngoc, Young-Koo Lee, and Sung-Young Lee. Owl-based user preference and behavior routine ontology for ubiquitous system. In *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE*, pages 1615–1622. Springer, 2005.

[27] Santi Phithakkitnukoon, Teerayut Horanont, Giusy Di Lorenzo, Ryosuke Shibasaki, and Carlo Ratti. Activity-aware map: Identifying human daily activity pattern using

mobile phone data. In *Human Behavior Understanding*, pages 14–25. Springer, 2010.

[28] Lei Shi and Rossitza Setchi. User-oriented ontology-based clustering of stored memories. *Expert Systems with Applications*, 39(10):9730–9742, 2012.

[29] Ilias Trochidis, Efthimios Tambouris, and Konstantinos Tarabanis. An ontology for modeling life-events. In *Services Computing, 2007. SCC 2007. IEEE International Conference on*, pages 719–720. IEEE, 2007.

[30] Michael Uschold and Martin King. *Towards a methodology for building ontologies*. Citeseer, 1995.

[31] Daisy Zhe Wang, Yang Chen, Sean Goldberg, Christan Grant, and Kun Li. Automatic knowledge base construction using probabilistic extraction, deductive reasoning, and human feedback. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 106–110. Association for Computational Linguistics, 2012.

[32] Xiao Hang Wang, Da Qing Zhang, Tao Gu, and Hung Keng Pung. Ontology based context modeling and reasoning using owl. In *Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on*, pages 18–22. Ieee, 2004.

[33] Anders Wimo, Bengt Winblad, Hedda Aguero-Torres, and Eva von Strauss. The magnitude of dementia occurrence in the world. *Alzheimer Disease & Associated Disorders*, 17(2):63–67, 2003.

[34] Lijuan Marissa Zhou, Cathal Gurrin, and Zhengwei Qiu. Zhiwo: activity tagging and recognition system for personal lifelogs. In *Proceedings of the 3rd ACM conference on International conference on multimedia retrieval*, pages 321–322. ACM, 2013.