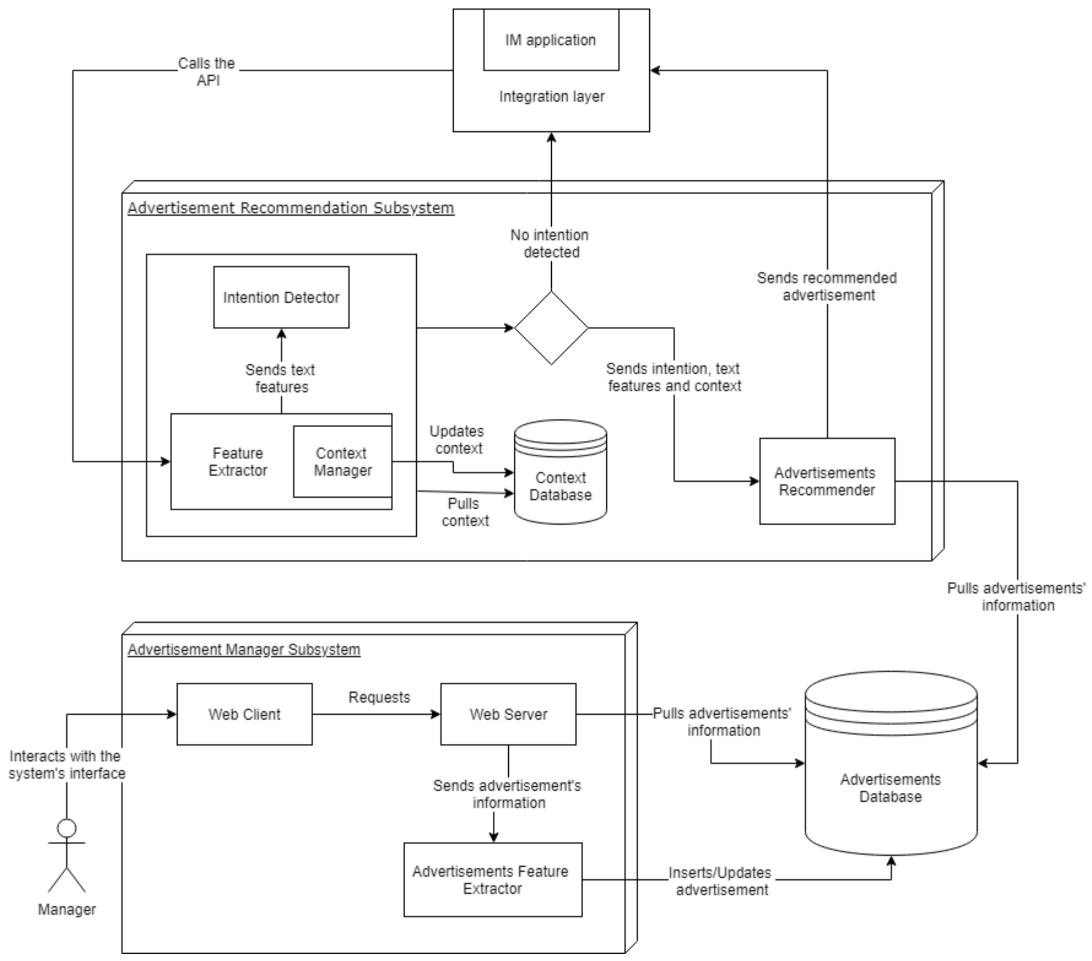

IM application
Integration layer
Calls the API
Advertisement Recommendation Subsystem
Intention Detector
Sends text features
Feature Extractor
Context Manager
No intention detected
Sends recommended advertisement
Sends intention, text features and context
Updates context
Pulls context
Context Database
Advertisements Recommender
Pulls advertisements' information
Advertisement Manager Subsystem
Web Client
Requests
Web Server
Pulls advertisements' information
Advertisements Database
Interacts with the system's interface
Manager
Sends advertisement's information
Advertisements Feature Extractor
Inserts/Updates advertisement

Gabriel Angel Amarista Rodrigues

Contextual Advertising Engine

Dissertation/Internship Report
Master in Informatics Engineering
advised by Antonio Dourado and Pedro Andrade
and presented to the Department Informatics Engineering
of the Faculty of Sciences and Technology of the University of Coimbra

July 2018

• U C •

Universidade de Coimbra

University of Coimbra, Faculty of Science and Technology

Contextual Advertising Engine

Master Degree Dissertation

Author:

Gabriel Angel Amarista Rodrigues

Advisors:

Pedro Andrade

António Dourado

UNIVERSIDADE DE COIMBRA

July 2018

Informatics Engineering Department (IED)
Faculty of Science and Technology, University of Coimbra
Polo II, Rua Sílvio Lima, 3030-790 Coimbra
Tel.: +351 239 700 600 | Fax: +351 239 700 688 |
E-mail: fctuc@fct.uc.pt

WIT Software, S.A.
Business Center of Taveiro
Estrada de Condeixa, 3045-508 Taveiro, Coimbra
Tel.: +351 239 801 030 | Fax: +351 239 801 039 |
E-mail: info@wit-sotware.com

Intern:

Gabriel Angel Amarista Rodrigues

grodrigues@student.dei.uc.pt

WIT Advisor:

Pedro Andrade

pedro.andrade@wit-software.com

IED Advisor:

Antonio Dourado

dourado@dei.uc.pt

Panel's Members:

Vasco Pereira

vasco@dei.uc.pt

Cesar Teixeira

cteixei@dei.uc.pt

Acknowledgements

My gratitude for achieving the submission of this work goes for those *capeless* heroes who made my evolution, as a person and as a professional, possible.

WIT Software took me in for a very challenging subject and they never gave up on me after the ups and downs I went through, especially during the first half of the internship. Always giving me perfect conditions to support my research and work. A special mention goes to my coordinator Pedro Andrade, who made up time out of thin air to have sessions with me about the progress of the project.

Friends, family and professors. All gave a tiny piece of themselves to me, which altogether gave me the capability to reach this milestone in my life.

Special regards to my grandmother and mother who always greeted me with a smile and open arms, even though I spent most of my time home in front of a screen, working.

Abstract

In the last two decades, society has been experiencing the ubiquitous intrusion of advertising; which has been increasing ever since expanding to novel communication channels, e.g. Social Networks. This constant intrusion led users to build a negative mental concept of “Advertising” as a whole and started ignoring it. Consequently, novel methods of advertising had to be explored, by the advertising industry, in order to adapt to this new behaviour.

Simultaneously, the flow of users’ attention has been increasingly shifting towards Messaging Apps, especially on smartphones, making it a highly desirable media for the advertising industry.

The objective of this project is to explore a new solution for advertisers having considering the aforementioned facts. More specifically, its purpose is to detect human intentions and context from instant messaging conversations between people and use it to recommend, or not, an advertisement corresponding to those intentions and context.

The current document describes, in great detail, the process and outcome of such exploration.

Keywords: Contextualised Advertising; Artificial Intelligence; Machine Learning; Deep Learning; Natural Language Processing; Word Embeddings; Recommender Systems; Information Extraction; Data Science.

Resumo

Nas duas últimas décadas, a sociedade tem experimentado a ubíqua intrusão da publicidade; a qual tem vindo a aumentar expandindo-se aos novos meios de comunicação, i.e. as redes sociais. Esta constante intrusão levou às pessoas a criar uma imagem mental negativa sobre a publicidade e começando a ignorá-la. Como consequência, novos métodos de publicidade tiveram de ser experimentados, pela indústria da publicidade, com o objetivo de adaptar-se a esta nova tendência.

Em paralelo, o fluxo da atenção dos utilizadores tem vindo a focar-se muito nas aplicações de *Messaging*, especialmente em *smartphones*, tornando este meio de comunicação muito desejável para a indústria da publicidade.

O objetivo deste projeto é o de explorar uma nova solução para a indústria da publicidade considerando os factos mencionados previamente. Mais especificamente, o propósito é o de detetar intenções humanas e contexto a partir de conversações de mensagens de texto instantâneas entre pessoas e usar esta informação para recomendar, ou não, uma publicidade que corresponda às intenções e contexto detetados.

O presente documento descreve, com grande detalhe, o processo e resultado de tal exploração.

Palavras-chave: Publicidade contextualizada; Inteligência artificial; Aprendizagem computacional; Aprendizagem profunda; Processamento de linguagem natural; Vetores de palavras; Sistemas de recomendação; Extração de informação; Ciência de dados.

Contents

List of Figures

List of Tables

Acronyms

AI – Artificial Intelligence

API – Application Programming Interface

CNN – Convolutional Neural Networks

CRF – Conditional Random Field

DL – Deep Learning

FMDS – Foundational Methodology for Data Science

IM – Instant Messaging

LSTM – Long-Short Term Memory

ML – Machine Learning

MLA – Machine Learning Algorithm

NER – Named Entity Recognition

PE – Phrase Embedding

POS – Part-of-Speech

RCS – Rich Communication Service

SVM – Support Vector Machine

WE – Word Embedding

WPEA – Word/Phrase Embedding algorithm

Chapter 1: Introduction

The present document has the purpose of detailing the process, studies and work done during the internship called *“Contextual Advertising Engine”*.

The internship was proposed and supported by the enterprise WIT Software, S. A. in the context of an academic internship for the Master’s degree in Informatics Engineering, strand in Intelligent Systems, of the Department of Informatics Engineering of the University of Coimbra.

It had a duration of 10 months. During that period the intern had to research and develop a prototype of a system capable of extracting relevant information from *Instant Messaging* conversations. Making use of it to suggest an advertisement concerning the detected context and desires of the users.

The current chapter is dedicated to the introduction of the motivation for the project, its context and, finally, a description of this document’s structure.

Section 1.1: Motivation and Contextualization

Originally, advertising was made with the exclusive purpose of selling. It used diverse strategies to achieve its goal. For instance, some advertisement campaigns were built around characters that supported a product, leveraging people’s empathy, making them feel identified with it and, consequently, boosting their sales. However, its blind focus was always *to sell* (Gallegos, 2016).

With the digital era, several diverse communication methods emerged, in which advertising also got in. Rapidly, people were bombarded with advertising that was present in everything they did. It was not long until people started to associate advertisements with annoyance, ultimately creating a negative image of advertising and its industry. At that moment, there was not much a single person could do in order to avoid advertisement, but it created a need, a demand (Gallegos, 2016).

In the decade of the 2010’s new services surged to satisfy the need of avoiding the constant intrusion of advertising. These services emerged with different strategies but the same objective: to free people from unwanted advertisements, and they came to be known as “*Ad blockers*”. Concurrently, services like Netflix and Spotify arrived at the market offering premium ad-free solutions. Short after, all the aforementioned solutions grew quickly in popularity revealing, not only that people were bothered by advertisements, but, more relevantly, they were actively choosing to pay to get rid of the of advertisements (Gallegos, 2016).

The growth of these services can be seen in *Figures 1*, *2* and *3*.

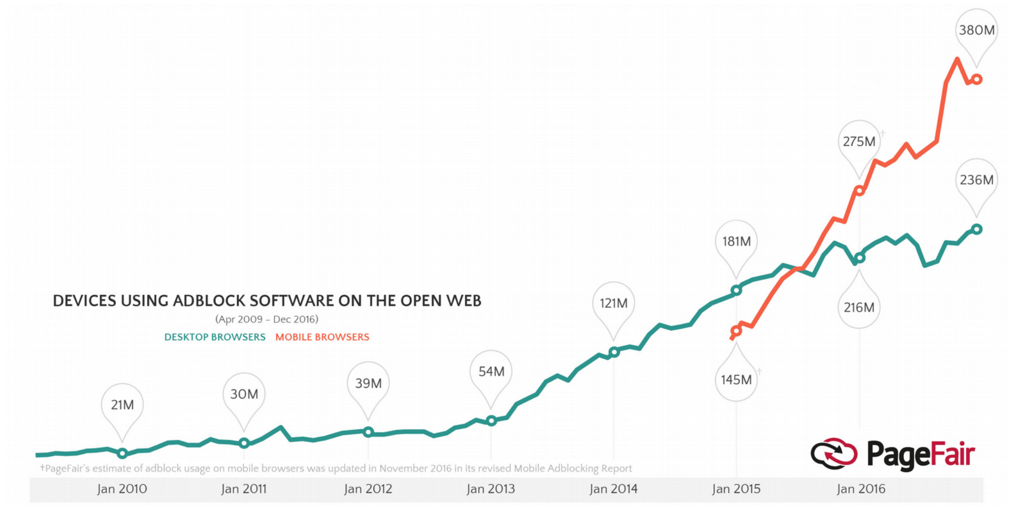


Figure 1: Adblock Software usage, from 2010 to 2016 (PageFair, 2017)

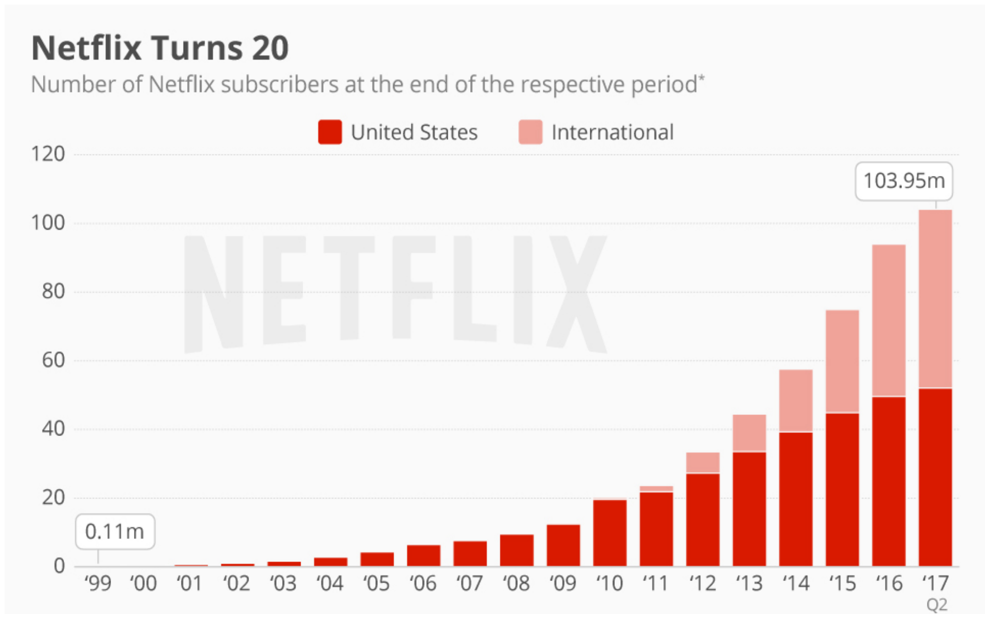


Figure 2: Netflix's subscribers growth in millions, from 1999 to 2017 (Richter, 2017)

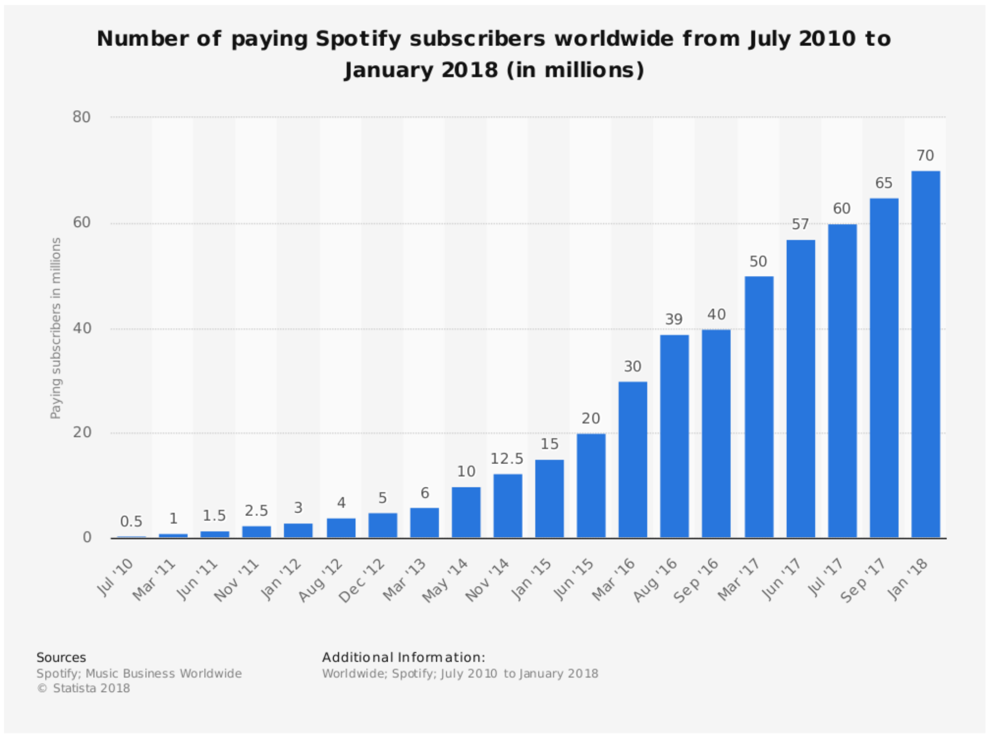


Figure 3: Spotify's subscribers growth in millions, from 2010 to 2018 (Statista, 2018)

As we can see in the previous figures, a drastic shift took place in the last decade. A shift that had a strong negative impact on the advertising industry.

As a reaction, the industry was forced to change its strategy. Nowadays, this industry no longer focuses directly on selling. Instead, its focus is on the creation of a community and a brand people can trust.

Different novel strategies emerged to make advertising more appealing to people, preferring friendliness over intrusiveness. One of these strategies is called “*Contextual Advertising*”.

The strategy in question holds the following idea as its cornerstone: “*Serve the right message to the right person at the right time*” (Dsquared Media, n.d.; Hoy, 2014). “*An advertisement succeeds when an advertisement resembles a friendly suggestion instead of a forced marketing agenda*” (Carty, 2017).

The advertising industry is already making use of these strategies to engineer solutions for their usual media channels. However, there is a channel, a communication channel, that has been growing and is rapidly taking over the users’ attention; this channel is the *Instant Messaging* (IM) applications.

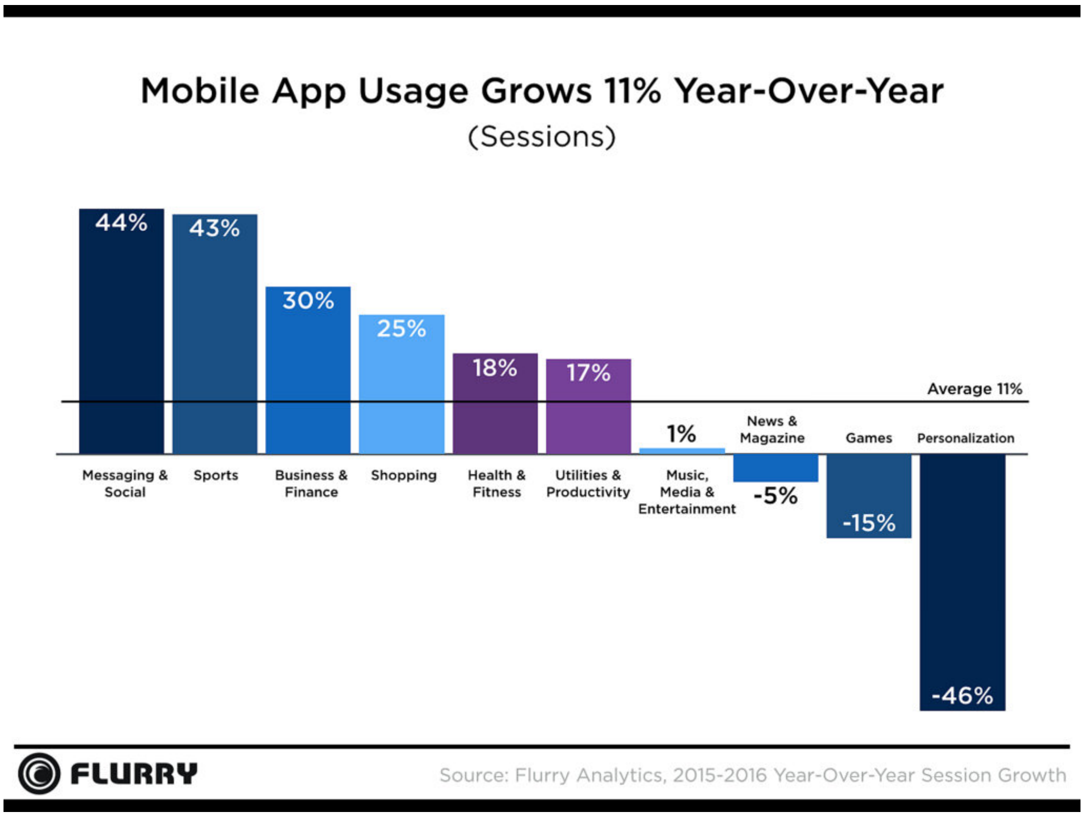


Figure 4: Mobile applications' usage 2015 - 2016 (Khalaf, 2017)

As reported in 2017 by Flurry Analytics (a company bought by Yahoo to, among other functions, measure mobile users' behaviour) IM applications have seen its usage increased, but more relevantly, the time users spent on them has grown as well (Khalaf, 2017). See Figures 4 and 5.

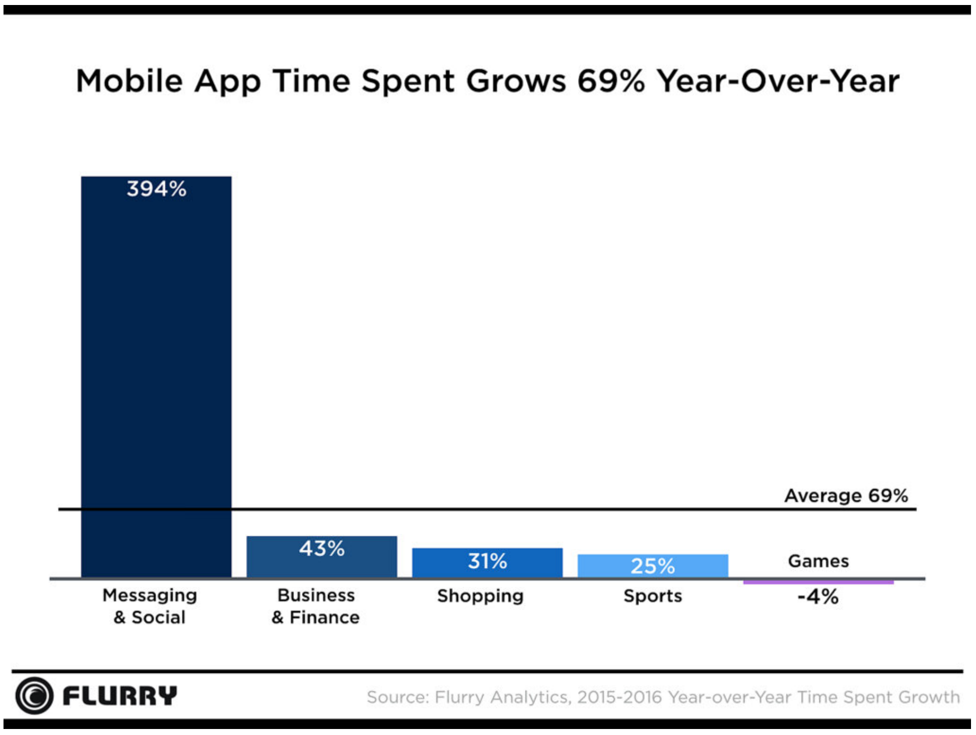


Figure 5: Mobile applications' time spent 2015 – 2016 (Khalaf, 2017)

IM applications have both an immense user base and users' attention, becoming an incredibly relevant channel, especially for divulgation purposes. Nevertheless, methods to profit this channel hadn't been much explored. The surface of it is being scratched by *Customer Support Chat Bots*, but there is still much work to do in order to understand and exploit this channel's full potential.

The crossing of the facts introduced made visible a non-explored business opportunity that may hold immense potential. Being this the motivation behind the project.

Section 1.2: Objectives

The purpose of this internship project is to research and develop a prototype solution regarding **Contextual Advertising** – as the technique – and **IM**

applications – as its channel – in order to verify its technical feasibility and the market's interest in solutions for this unexplored domain. More specifically, the prototype must be able to extract relevant information, from an IM conversation, and use it to recommend a fitting advertisement at a *convenient time*.

The aforementioned *convenient time* is associated with the most opportune moment to do a recommendation (send an advertisement). In this project, the *most convenient time* was empirically defined to be the immediate moment after an intention was detected and a good recommendation found.

Section 1.3: Thesis Structure

The current document follows the subsequent structure:

- **Chapter 1: Introduction**. This chapter presents the context in which the project is scoped and a deep understanding of its motivation.
- **Chapter 2: Requirements and Risks**. This chapter presents the analysis of both the requirements and risks associated with the project, as well as the introduction of this project's unique characteristics and limitations.
- **Chapter 3: Research and Definitions**. This chapter presents an overview of concepts, techniques and research associated with the approached solution.
- **Chapter 4: System's Solution**. This chapter presents the designed solution, its requirements and implementation details.
- **Chapter 5: Dataset Research and Construction**. This chapter presents the research, harvesting, development and cleaning of all datasets required for the approached solution.
- **Chapter 6: Project's Success Analysis**. This chapter presents the analysis of success of the system given its initial requirements and final state.

- **Chapter 7: Future Work and Conclusions**. This chapter presents a compilation of subsequent steps that could improve the implemented system, and presents the author's learnings and final thoughts.

Chapter 2: Requirements and Risks

This chapter is dedicated to the detailed explanation of the project’s most relevant characteristics and limitations; the specification of the objectives and requirements to be fulfilled throughout the internship; the limitation of the scope considering the time frame available; the risk analysis and, finally, the introduction of the methodology, work plan and milestones.

Section 2.1: Project’s Characteristics

In pursuance of the success of any project, the first step to be taken must be towards the understanding of the project itself. The motivation, problem, context and resources available must be well-known. The most relevant characteristics of the project are presented in the following subsections:

2.1.1: The language to work with

The IM conversations to be analysed and, consequently, to be supported by the project ought to be in the English Language. However, no specific English’s dialect was to be considered. The rationale behind this decision resides in the fact of the English language being the most spoken language worldwide, which translates into approaching a bigger market and more available resources.

2.1.2: The IM conversations

This project proposes a solution for the integration of Contextual Advertising in IM applications. This implies the analysis of conversations – text – from this specific channel. The kind of communication often found in this channel, and in social media applications overall, possess unique characteristics. It is asynchronous, erratic, informal and is loaded with abbreviations, ellipses (omissions of words), implicit references and *emojis* (Hu, Zheng, Yang, & Huang, 2018; Tagg, 2009). Furthermore, it has shown noticeable variations depending on the person’s personality, and to have the capability of transmitting non-verbal information (Roffo, Giorgetta, Ferrario, & Cristani, 2014).

The IM channel's flexibility allows people to express themselves beyond the language limitations. However, this fact imposes an increase in complexity since the formal language rules are valid no more.

In addition, NLP (Natural Language Processing) benchmarks comparing formal writing text and social media text have been done, which results indicate that social media text has to be studied separately. Solutions performant on one type of text does not necessarily perform well on the other (Rudrapal, Jamatia, & Chakma, 2015), leading to conclude that, although the format is the same, the representation of the information is different.

2.1.3: The need for specific datasets

Subsequently, since the data to work with has several unique characteristics, it is imperative to find representative datasets.

No dataset was provided initially. Hence, a supplementary thorough research for appropriate datasets had to be performed. Having a good dataset to work with is essential to both understand the nature of the data and to be used as the source for the AI algorithms.

Ultimately, the two core datasets – the Intention Detection and the Twitter datasets – had to be built from scratch. The processes of their construction are introduced in *Chapter 5: Dataset Research and Construction*.

2.1.4: The retrieval of context exclusively from text

Although the IM applications are hosted by systems possessing advantageous information regarding context, by the definition of the project, this information was to be seen as unreachable. In other words, it could not be considered as a resource for the solution.

The importance of this limitation resides in the impossibility to identify a user's location from the GPS information since it was unavailable.

The rationale behind this decision was the need for additional privacy policies, which we aimed to keep at a minimum.

2.1.5: The limitation of the scope

As mentioned previously, the objective of the project is to analyse a conversation, from an IM application, and recommend a fitting advertisement at a convenient time. However, by acknowledging the vast collection of advertisements that could be recommended, the limitation of the scope remained evident.

Additionally, a narrower scope would allow a much deeper analysis and understanding of the selected uses cases, opposite to a broader yet superficial understanding of generic cases. A smaller scope would also have a positive impact on the prototype's performance and adapt better to the limited time frame of the project.

Regarding context. The agreement was to approach and leverage only 3 types of context: **location**, **date** and **time**.

Regarding advertisements. The agreement was to leverage an existing system called: *Google Places*, using each single *place* as an advertisement and extracting its information via the *Google Places API*. This decision overlaps with the use cases defined later in this chapter, in *Section 2.3: Use Cases.*

Section 2.2: Requirements and Thresholds of Success

Prior to any kind of limitation of the project's scope, the specific requisites for the system must be identified, discussed and concerted by the stakeholders involved. In this case, the intern and his coordinators.

The technique chosen to do the requirements' analysis is called "MoSCoW" (Craddock, Fazackerley, Messenger, Roberts, & Stapleton, 2008). The rationale behind this decision was the need for a simple yet efficient approach to organise and prioritise requirements.

In simple terms, the *MoSCoW* technique prioritises the requirements assigning them into one of four categories:

- **MUST**: essential requirements that *must* be included in the project.
- **SHOULD**: important requirements that *should* be included, but are not critical for the project.
- **COULD**: desirable requirements that *would* be pleasant to have, but have a small impact on the project's success.
- **WON'T**: relevant requirements that *will not* be delivered on the actual iteration of the project due to the limited resources.

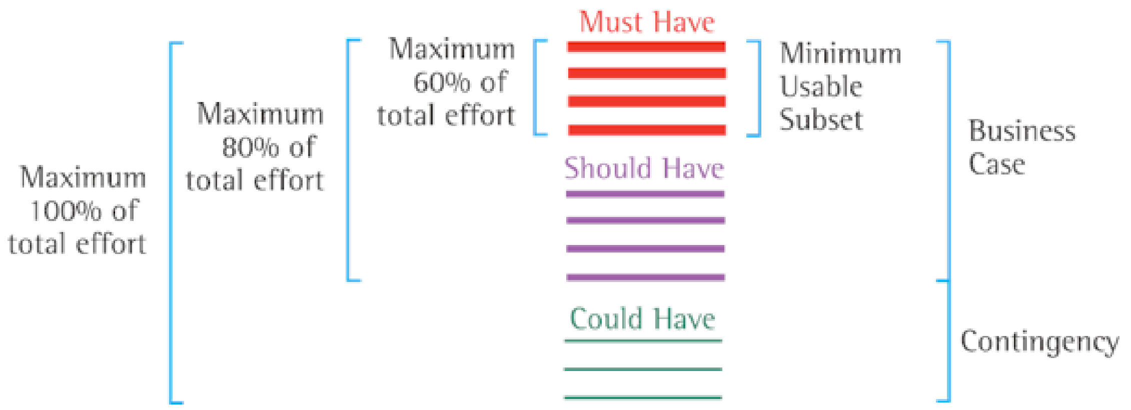

Must Have
Maximum 60% of total effort
Minimum Usable Subset
Maximum 80% of total effort
Maximum 100% of total effort
Should Have
Business Case
Could Have
Contingency

Figure 6: The MoSCoW technique, prioritisation boundaries (Craddock, Fazackerley, Messenger, Roberts, & Stapleton, 2008)

The requirements' analysis and prioritisation using the aforementioned technique is shown in *Table 1*.

Nº	**Requirement**
1	The system **MUST** be able to work with English conversations
2	The system **MUST** extract relevant features from the conversations
3	The system **MUST** detect and discriminate the user's intentions/needs/desires
4	The system **MUST** detect, extract and manage the context of the conversations
5	The system **MUST** recommend advertisements based on the conversation's most recent intention, text features and context
6	All information **MUST** be persisted in a database
7	The system **MUST** be available for consumption via an API
8	The system's API **MUST** be capable to identify conversations within calls
9	The system's API **MUST** always return a response
10	The system's API response **MUST** be within the user's conversational context
11	The system **MUST** have a visual interface to manage advertisements
12	The prototype **MUST** have a visual interface to demonstrate the system in action
13	Datasets **MUST** be developed to train and test the system

Nº	Requirement
14	All Artificial Intelligence algorithms **MUST** be evaluated
15	The system's performance **MUST** be evaluated
16	Alternative algorithms **SHOULD** be evaluated
17	Different tools and APIs **SHOULD** be evaluated
18	The system **SHOULD** detect and persist impossible recommendations
19	The system's performance **COULD** be monitored
20	The system **WON'T** persist user-related information and behaviour
21	The system **WON'T** have a dashboard with system or users metrics
22	The system **WON'T** be integrated with WIT's RCS Messaging application

Table 1: Requirements prioritisation

The final work delivered fulfilled most of the requirements presented in *Table 1*.

Besides the requirements, a set of thresholds that would determine the success of the final work were defined. The objective of this was to establish *SMART* goals to be used as guidelines throughout the project's development.

SMART goals are short statements clearly written that contain the important results one aspires to achieve in a project (University of California, 2016).

The thresholds of success are presented next:

Statement	**Time boundary**
All *must have* requirements must be fulfilled	By the end of the internship
The system must approach all proposed use cases	
The Intention Detection algorithm must reach an F1-score **superior to 75%**	
After a call to the *Advertisement Recommendation* API, a response must be given in **less than 1.5 seconds**	
The prototype **must have** a graphical demonstration of the system and idea behind it	

Table 2: Thresholds of Success

Section 2.3: Use Cases

As mentioned previously in this chapter in Section 2.1: Project's Characteristics, limitations were needful. As such, the uses cases were also limited in number and their interdependence with the Google Places system.

The use cases agreed upon are related to the kind of user intention the system must react to. These can be seen in *Table 3*.

Broad intention	**Intention**	**Google Places related**
Going out	… at night	Bars, clubs, pubs, etc.
Going out	… to eat	Restaurants, cafeterias, bakery shops, etc.
Going out	… to watch a film	Cinema, theatre, etc.
Delivery	… of food	Food delivery restaurants

Table 3: User intention use cases

Other use cases were considered. However, only the ones present in *Table 3* were taken into account. It was ultimately better to reduce the coverage of the system in exchange for potentially raising its performance. The importance of the demonstration of the developed system made this fact even more relevant. There is a preference for limited reliable systems, over complete unreliable ones.

Section 2.4: Risk Analysis

Once the *Thresholds of Success* are known, the identification of risks can be done. Risks represent the possibility of something negative happening. More specifically, the possibility of something happening that might prevent the project from reaching any of its *Thresholds of Success*.

The risks identified throughout the project’s development are presented in *Table 4*.

Nº	Risk statement
1	The absence of labelled datasets *might* require an immense effort towards their construction which cannot be properly predicted due to the intern's inexperience regarding this specific task
2	The absence of big datasets *might* compromise the performance of some AI algorithms
3	The absence of real data *might* difficult the understanding of its nature
4	The manual labelling *might* introduce a certain degree of bias since there is only one annotator involved in the project
5	The time frame given for the project *might* not be wide enough to reach the performance expected

Table 4: Risk statements

The risks identified were evaluated with respect to their **impact**, **probability** and **cost**, technique based on a risk identification framework developed by *The Software Engineering Institute* (Carr, Konda, Monarch, Ulrich, & Walker, 1993).

For this project's context, the cost will be represented by time, it being the most important resource. These aspects were selected to assist in the prioritisation of risks, which has to be done in order to focus on the most serious risks early.

Ideally, risks should be mitigated. However, this is not always possible. In those cases, thorough contingency plans must be designed to minimise the risk's consequences.

In *Table 5*, the analysis of the risks identified is presented, followed by *Table 6* were the handling strategies for each risk are described.

Nº	Impact	Probability	Delay (approx.)	Priority
1	Critical	High (p > 70%)	More than 1 month	9/10
2	Moderate	Very high (p > 85%)	Half a month	6/10
3	Critical	Medium (40% < p < 70%)	Half a month	6/10
4	Moderate	Medium (40% < p < 70%)	None	3/10
5	Critical	Medium (40% < p < 70%)	More than half a month	7/10

Table 5: Risk analysis. They are ordered by number, not by priority

Nº	Type of plan	Plan
1	Contingency	• Research for dataset • Attempt to adapt datasets with similar data • Unveil the risk as fast as possible • Seek council with experts in the area
2	Contingency	• Focus on approaches that perform well with small datasets
3	Contingency	• Study the nature of the data by other means. E.g. reading research associated with IM conversations
4	None	In order to mitigate the risk unavailable resources were needed, especially time. As a consequence, the risk was assumed and no plan was designed
5	Mitigation	• Evaluate and evolve • Study benchmarks of different solutions before implementing • Detect the weakest sides and focus there

Table 6: Risk handling. Risk analysis plans

The original proposal for the project did not mention the need for datasets, which was interpreted as a sign of ownership of the resources needed for its development. Nevertheless, it was a wrong assumption that led to a delay in the identification of ***risk 1****'s* occurrence, which recurrently unveiled ***risk 2*** and ***3*** as well. The three contingency plans were executed to minimise the cost and, ultimately, achieve the *Thresholds of Success*.

Several problems regarding the datasets raised, mainly as a consequence of the unique characteristics of the needed data, text from IM conversations. Ultimately, data from Twitter was harvested as a reference for the manual construction of another dataset. The process is further explained in *Chapter 5: Dataset Research and Construction*.

The mitigation plan of ***risk 5*** was immediately executed. Becoming part of the workflow of the development of the system.

Section 2.5: Work Plan and Milestones

The internship had a duration of 10 months, going through two semesters. As standardised by this kind of internships, the first semester was dedicated to the study and definition of the work that will be later executed in the second semester. Due to the activation of risks presented in the preceding section, and the unknown effectiveness of the contingency plans, the work plan had to include pessimistic estimations of the amount of time necessary to achieve this project's milestones.

Furthermore, a very simple yet efficient workflow had to be approached to guarantee the time was being invested in the right components of the project.

2.5.1: Methodology

Although *SCRUM* was proposed initially, several artefacts and meetings of this framework made no sense whatsoever since the *team* was composed by a single member (Schwaber & Sutherland, 2013). Furthermore, following its traditional iterative approach makes difficult the estimation of the project's conclusion date for projects with a fixed scope, as the present one.

Moreover, AI projects have different needs than typical *Software Engineering* projects. Therefore, a more specialised methodology was to be researched. In 2015, a senior IBM Data Scientist had been documenting these aforementioned differences and had designed a methodology for this type of projects called: "Foundational Methodology for Data Science" (FMDS) *(Rollins, 2015)*.

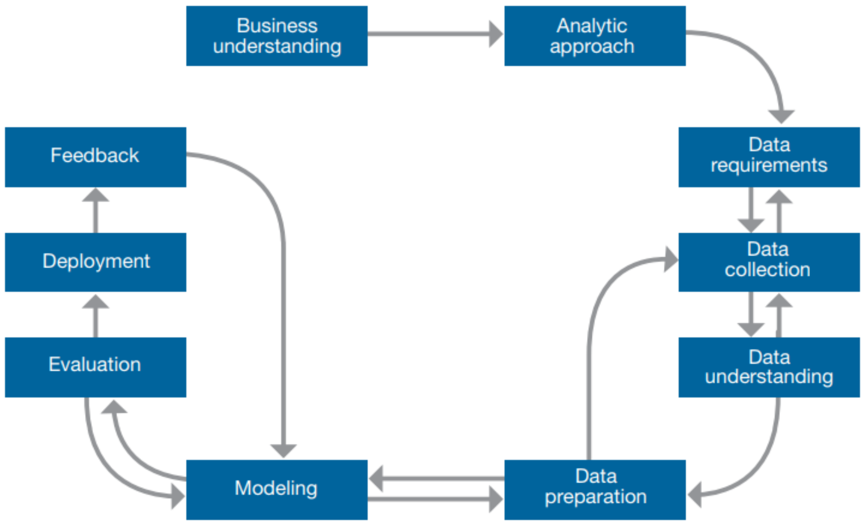


Figure 7: Foundational Methodology for Data Science, the 10 stages (Rollins, 2015)

In *Figure 7* the ten stages and interactions of the methodology are presented. This methodology also uses an iterative system but focuses on evolution rather than deployment. It also serves its purpose as guidance since it was specifically designed for this kind of projects.

The *FMDS* was used as guidance. On the other hand, the organisation and prioritisation of tasks followed a different pattern in order to guarantee the conclusion of the project.

The work plan designed was tailored with pessimistic time frames for each task towards each milestone. Within each task, the *FMDS* methodology was to be used together with an empirical prioritisation of subtasks.

2.5.2: Descriptive Work Plan

The project has a time frame of 37 weeks, from the 19th of September to the 18th of June. The first semester had a lighter weight, contributing only 20 hours of work per week. In contrast, for the second semester 40 hours per week were to be contributed. This unbalance is reflected in the work plan.

Milestones were defined in order to guide the process, establish deadlines and to assess the evolution and feasibility of the project over time. The milestones agreed upon were the following:

Milestone	**Deadline**
First presentation at WIT Software	31st of October of 2017
Presentation of the system's architecture for the designed solution	4th of December of 2017
Intermediate presentation and demonstration of the system, at WIT Software	25th of January of 2018
Intermediate defence at the University of Coimbra	29th of January of 2018
Presentation of the solution's performance	16th of April of 2018
Presentation of the functional prototype	22nd of May of 2018
Evaluation of the project's success	18th of June of 2018
Final report's delivery	2nd of July of 2018
Final presentation and demonstration of the system, at WIT Software	6th of July of 2018
Final defence at the University of Coimbra	10th of July of 2018

Table 7: Project's milestones

The descriptive work plan shows the contrast between what was originally planned, milestones set, and the actual execution of the project.

As can be noticed, no milestone regarding datasets was placed. The reason remains in the fact that the milestones were agreed before the 15th of November, the day when the absence of the datasets was confirmed and their construction was formally introduced as an imperative requirement for the project’s success.

The descriptive work plan together with the milestones can be seen in *Appendix A: Descriptive Work Plan*.

Chapter 3: Research and Definitions

This chapter is dedicated to the introduction of concepts, techniques and studies related to the project. The information researched go beyond the understanding of existing solutions. It focuses, mainly, on the understanding of explored approaches for another kind of problems handling similar data.

Once the characteristics of the project had been defined, the following step was to investigate if there were any existing systems attempting to solve the problem at hand.

Section 3.1: Contextualised Advertising in IM applications

As of today, there is no *public* solution to this problem. Nevertheless, the persistent growth of IM applications remains an unexplored business opportunity that is captivating companies' attention (Marketing Magazine, 2016).

Social Media is a similar channel to IM having to deal with analogous data, but it differentiates in the challenges faced to actually offer a contextualized advertisement service for companies. Namely, the information from the users' profile and their social interactions available in *Social Media* is much more easily analysed than the *raw text* found on the IM channel.

Besides *Facebook*, *Snapchat* has been doing tremendous work regarding the advertisement industry (Team Snapchat, 2018).

More relevantly, even though no public information has been explicitly revealed, the biggest technical companies are, secretly, working on approaches to integrate IM applications with the advertisement industry. Well beyond the personal assistants like *Alexa* and *Siri*, *Facebook* had been working on a project called: “Facebook M*”*. Google has also shown interest in the area, publishing their IM application called: *“*Allo" in order to collect and understand the data flowing in this channel.

Facebook started working on its *Facebook M* assistant back in August 2015 (Constine, 2015). However, as a matter of fact, on January 19th of 2018, *Facebook* decided to shut down the service. As explained in (Dillet, 2018) by a *TechCrunch* reporter, the secret behind the “success” of the project was not AI, but humans working behind the scenes. The rationale for the “M” project was for the AI agents to learn how humans behaved, so they could later substitute them. However, after two years of operation, Facebook decided it was enough, generating certain mystery over the matter.

In contrast, Google had shutdown *Allo*, for business reasons (Russell, 2018), but has been evolving another AI project of them called: “Smart Reply” (Perez, 2018). Firstly incorporated into their *Inbox* application to allow users to answer e-mails with a click, and currently evolving to any chat app an *Android* phone can have. Although *Smart Reply* is a service that does not seek to integrate the advertisement industry with IM applications, it surely has made great efforts into understanding and leveraging the data to provide a useful service.

Withal, no technical information is unveiled on this end, forcing the research to go deeper into the specifics of “How the problem could be solved?”.Then all was reduced to a single focus, the automatic understanding of text in IM conversations.

Although several cases of study leverage the automatic understanding of text, there is a specific and trending business service that has been on the rise lately, this service is called *“*ChatBots”.

ChatBots *"are computer programs designed to simulate conversations with human users, especially over the Internet"* (Drift, Audience, SalesForce, & myclever, 2018) Their relevance regarding this research does not reside in their use, but in the methods used to build them.

Most Chatbots are very similar to Interactive Voice Response systems (Wikipedia Community, 2002b). They possess a fixed decision diagram that guides the users through the possible hard-coded services it offers. No text analysis is done either since most of these system rely on buttons to obtain the users' input.

However, there is a smaller subset of Chatbots that have followed a different path, using NLP techniques in order to interact with users more humanly, understand and act considering accordingly. Many examples of AI-driven Chatbots exist, still, the focus of the research was not the bots themselves, but the techniques behind them.

The next section introduces the aforementioned techniques.

Section 3.2: Natural Language Processing

Much studies have been evolving around this topic. In this section they will be explained, as well as the complexity of the topic itself.

The first approach taken into the topic was the analysis of the language itself. The rationale was to detect patterns around the problem and, subsequently, apply pattern matching techniques to discriminate each case.

3.2.1: Pattern Matching

Although the pattern matching techniques do not entirely fit within the AI concept, they are widely used in systems claiming to be artificial intelligent systems. Take *chatbots* for instance. *Pandorabots* is the enterprise that built

Mitsuku, the world's best conversational chatbot (Plummer, 2017). Withal, *Mitsuku* uses in its code base a programming language tailored specifically for natural language software agents called: "AIML". This is a markup language consisting of the specification of text patterns and responses associated (Wikipedia Community, 2002a). In other words, the best existing *chatbots* are no more than an immense set of handmade pattern matching rules, which might sound disappointing yet it is highly effective.

Still, for this technique to be successfully used, it is necessary to understand the syntax of the text to be received and its context, all in order to manufacture the patterns.

Regarding the problem at hand, of detecting users' intentions throughout an IM conversation. The first step was to investigate grammatical constructions that relate to expressions of "desire", "needs" or "will". In the English language, there is a specific verb form associated exactly with the expressions of "desires", "urges", "proposals", "suggestions" among others. It is called: "The subjunctive form" *(Englishpage.com, n.d.)*. Although it is very interesting and opportune, it is everything but useful. This form is quite confusing, can be easily avoided by rephrasing a sentence and is rarely used. Conclusively, this information was discarded but the investigation of the understanding of the language continued.

Language is a complex system. Understanding it thoroughly is a tough labour. Nevertheless, there are some important key concepts that should be known.

Firstly, the levels of linguistic structure must be identified. *Figure 8* presents the six major levels and briefly describes them (Boundless Psychology, n.d.).

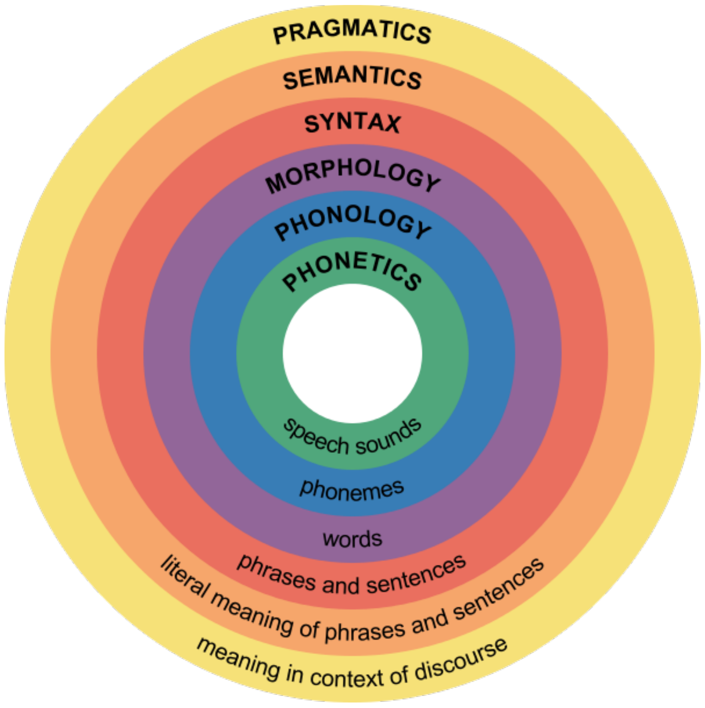


Figure 8: Major levels of linguistic structure (Boundless Psychology, n.d.)

Some of the linguistic levels present clear rules to assess their correctness, specifically: *phonology*, *phonetics* and *syntax*. Nonetheless, the other three levels present a much bigger challenge in their understanding due to their degree of subjectivity.

At the **morphology** level, words and their relations between them are studied. The complexity on this level, explained in (Liang, 2017), can be found in concepts such as:

- **Synonymy**, words with similar meaning to other words
  - For example the words "dark" and "obscure" hold similar meanings
- **Polysemy**, words carrying different meanings
  - For example the word "light" can mean "not heavy" or "a bright object".
- **Hyponymy**, words representing concepts in which other words are included.

    - For example the word “dog” is a hyponym of “canine”.

- **Meronymy**, words representing concepts associated as part of other words.
    - For example the word “tail” is a meronym of “dog” since a dog has a tail.
- **Multi-word expressions**, compounded words making up a more complex concept.
    - For example expressions like “break down”, “carry on”, “wake up”, etc.

The outer to the linguistic level’s circle, the more complex the understanding. Considering **pragmatics**, not only all the other levels apply their complexity on it, but it also adds its own. In (Liang, 2017), and especially in (Radev, 2016) this fact can be appreciated.

For instance, the sentence “He is an unbelievable worker” can hold two different meanings. Either the worker is a regular liar and should not be believed, or the worker’s quality is so high, that it is unbelievable (Radev, 2016). Nevertheless, the context given is not enough to undoubtedly identify which is the intended connotation.

Approaching **pragmatics** with **pattern matching** techniques is unthinkable. However, using **morphology** level concepts can be easily exploited. Gazetteers – dictionaries of entities with further information about them – could be created leveraging those concepts to be later consulted at the matching moment.

In conclusion, these techniques can be very useful if the case of study can be feasibly modelled in an approximately deterministic manner.

3.2.2: Ontologies

Ontologies are a more sophisticated technique that leverages dictionaries and linguistic levels such as hyponymy and synonymy to create a linguistic knowledge base. The formal definition is presented in (Moltmann, 2016) as “a branch of both metaphysics and linguistic semantics which aim is to uncover the existence of categories, notions, and structures that are implicit in the use of natural language”.

Accordingly, ontologies can be queried to understand better the concepts and relationships of a given word.

To build an ontology requires an immense amount of work, which cannot be usually done by a single person in useful time. Also, the extensibility of an ontology is crucial for its continuous evolution.

Ontologies have been developed and evolved through this millennium with the objective of enriching artificial intelligence algorithms. The most popular ontologies are:

WordNet (Miller, 1995), probably the most well-known and used ontology. It links English nouns, verbs, adjectives and adverbs to sets of synonyms that are linked between themselves following semantic relations of words. However, it is handcrafted, resulting in a slow-paced evolution.

YAGO (Suchanek, Kasneci, & Weikum, 2007), an evolution over *WordNet* that merges knowledge from it and Wikipedia to build more than 5 million relationships between more than 1 million entities. Moreover, it is easily extendable and it was populated using an automatic approach.

ConceptNet (Liu & Singh, 2004), with the same motivation, it represents words' knowledge and relationships but this time over a graph structure. Similarly to *YAGO*, it uses an automatic approach to extend itself, using as source the Open Mind Common Sense corpus collected by a crowd system. Furthermore, *ConceptNet* has been the ontology with the most notorious evolution, nowadays allowing the enrichment of word embeddings (Speer, Chin, & Havasi, 2017; Speer & Lowry-duda, 2017).

One of the most interesting uses of ontologies, besides finding word relations and concepts, is the usage of semantic similarities.

Semantic similarities offer the possibility of knowing how much two concepts relate to each other. Before having this technology the alternative was to compare the different texts by their syntactical differences, more specifically, the number of editions that would be necessary to transform one text into the other. One of these techniques is called: *The Levenshtein Distance (Wikipedia Community, 2003)*.

Several different designed semantic similarities exist. They leverage ontologies, language models (probability distributions of word usage in corpora) and information theories (such as entropy and information gain) (Elavarasi, Akilandeswari, & Menaga, 2014). Thorough studies and comparisons of these metrics were executed on (Slimani, 2013; Zhu & Iglesias, 2017) highlighting three specific measures:

The **Wpath**, is a hybrid similarity measure that leverages both ontologies and information theory. The ontology features extracted are: the *len* – the distance between two concepts within an ontology. – and the *LCS* (Least Common Subsumer) – the most specific ancestor common to two concepts –.

Regarding information theory, it computes information features from given corpora, namely the *IC* function – returning the probability of encountering a concept in a corpora –. Additionally, the contribution of the *IC* function is quantified by a *k* parameter whose optimal value depends on the specified domain.

In the *Equation (1)* the Wpath semantic similarity measure can be appreciated. The variables *c1* and *c2* represent the concepts (words) to be compared. A further explanation is detailed on (Zhu & Iglesias, 2017).

$$\mathrm{Sim}(c1,c2)=\frac{1}{1+len(c1,c2)*k^{IC(LCS(c1,c2))}} \qquad (1)$$

Secondly, the **Zhou**. Yet another hybrid measure with the introduction of a weight parameter *k*. Differently from the *Wpath k* parameter, this *k* parameter balances the impact of both ontologies and information theory. For instance, if *k=0.5* both the ontologies and information theory will have the same impact.

All the concepts used in the *Zhou* semantic similarity measure were previously introduced, with the exception of the *Depth* function. This function returns the distance between a concept and the ontology’s *root*.

Equation (2) presents the formula for this similarity measure. A further explanation is detailed on (Zhou, Wang, & Gu, 2008)

$$\mathrm{Sim}(c1,c2)=1-k\,x\,\frac{\log(len(c1,c2)+1)}{\log(2\,x\,max(Depth(ci))-1)} \qquad (2)$$
$$-(1-k)\,x\,((IC(c1)+IC(c2)-2\,x\,IC(LCS(c1,c2)))/2)$$

Finally, the **Li** is a structure-based semantic similarity measure that leverages both the *len* and the *LCS* between the two concepts through a non-linear function. See *Equation (3)*. Also, the *alpha* and *beta* parameters are both optimization variables whose optimal values found are “0.2” and “0.6” correspondingly, as explained in (Li, Bandar, & A. McLean, 2003).

$$\mathrm{Sim}(c1,c2)=e^{(-\alpha * len(c1,c2))} * \left(\frac{e^{(\beta * LCS(c1,c2))}-e^{(-\beta * LCS(c1,c2))}}{e^{(\beta * LCS(c1,c2))}+e^{(-\beta * LCS(c1,c2))}}\right) \tag{3}$$

3.2.3: Text to Numbers

The peculiarity of Deep Learning (DL) approaches is the need for numeric features. The first adaptation of NLP to DL was called: “Bag of words”.

The **Bag of words** is a technique that represents documents by identifying all different words on them and defining sentences by a vector word occurrences. Each vector position represents a specific word in the document. The limitations of these models are that no context, grammar or even word order was considered. Also, for corpora with extensive vocabularies, this method would not be very efficient, since all sentences will be represented by long vectors mainly populated by *zeros.* Applying DL was not feasible in those cases *(Wikipedia Community, 2007a)*.

In order to automate the solution of a problem, one must first understand how a human would approach it and break his process down into simple steps a computer could do. The conclusion from this reflection is the realization that if humans need certain information to solve a problem effectively, computers will surely need it as well. In a study from 1965 (Rubenstein & Goodenough, 1965), researches successfully proved that knowing the adjacent words of a word will highly facilitate the detection of synonyms from the similar contexts. Therefore, in order to achieve better performance, the context of words should be included in the models.

Conditional Random Fields (CRF), is a statistical approach that can leverage from the latter presented fact (Lafferty, McCallum, & Pereira, 2001). Basically, it uses feature functions to calculate the probability of a label. In its usage to solve NLP problems, a window of words is given as features, which, contrarily to the *Bag of words* techniques, consider words' order and neighbourhood. However, DL could not yet be applied.

Further models were developed until one specific breakthrough was made. The *Word Embeddings* (WE). They made feasible the use of DL in NLP related problems by creating a vectorised representation of words. These n-dimensional numerical vectors had interesting semantic properties, being the most relevant the fact that two similar words would have similar vectors. Hence, techniques such as the *Cosine Distance* could be used to calculate the semantic similarity between two words (Wikipedia Community, 2007b).

Word2Vec is an evolution of the *Skip-gram* model. It is a distributed representation of words in vectors. More importantly, the dimensionality of the vectors could be decided as will. Based on the vectors obtained by *Bag of words* techniques, or by the *Skyp-gram* models, it encodes the information through neural networks creating smaller and more compact vectors. The encoded information includes surrounding words inside a limited-sized window (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013).

After *Word2Vec*, other word embedding techniques emerged with time. Namely, *GloVe* (Pennington, Socher, & Manning, 2014a) and *FastText* (Bojanowski, Grave, Joulin, & Mikolov, 2016b) Besides new techniques, pre-trained vectors on massive corpora were also made available. These pre-trained vectors were used as Transfer Learning to solve NLP problems in different domains. Furthermore, they could be trained with different, domain-specific corpora to avoid long training times and leverage domain-specific information (Mikolov, Grave, Bojanowski, Puhrsch, & Joulin, 2017).

Phrase embeddings (PE) also emerged. *Doc2Vec* (Le & Mikolov, 2014) and **InferSent** (Conneau, Kiela, Schwenk, Barrault, & Bordes, 2017) were specifically tailored to represent sentences through vectors. Improving over the word embeddings that struggled in NLP problems requiring the classification of whole sentences.

From this stage on, much research has been done in developing the best phrase embedding possible. In (Stanford NLP Group, n.d.) we can appreciate a benchmark over the NLP problem of Natural Language Inference. Successful research takes advantage of different DL techniques. Convolutional networks (Zhang, Zhao, & Lecun, 2015) and Long Short-Term Memory Networks (LSTM) (Ma & Hovy, 2016), among others, show promising results.

Section 3.3: Recommender Systems

One crucial function of the project's solution is the advertisement recommendation. Therefore, the literature and state-of-the-art of this specific AI algorithms was also a target of study.

The objective of *Recommender Systems* is to use past behaviours and current conditions to suggest an action or item within a specific context. The challenge encountered by these systems is the ranking of matches between users and potential recommendations, which has to be done effectively to satisfy the highest number of users as possible.

In the literature, we can easily find 3 traditional recommender systems*:* **Collaborative Filtering**, **Content-Based** and **Knowledge-Based** systems (Aggarwal, 2016).

The Collaborative Filtering, in short terms, is a type of recommender system that leverages users' behaviour to match them with each other. Once the high confidence matches are selected, the system analyses the user's matches to make the recommendation. The recommendation is based on the rationale that if both users have similar behaviours, they will probably share preferences.

For example, assume this system is implemented on a music service and we have both *Alice* and *Bob* as matches. The system would recommend to *Alice* songs *Bob* listens to and she does not. Regarding *Bob*'s recommendations, they will follow the same logic.

These systems, however, have the tendency to suffer from *cold-start problems*, meaning, the initial insufficient amount of information. It is fairly impossible to match users if there is barely information about them. No matches translate into no recommendations.

The Content-Based system, similarity to the previously explained system, this one also matches entities in order to make its recommendation, but, instead of users, it matches actions/items. In other words, it analyses a user's past actions/items and seeks for other actions/items carrying similar characteristics.

Using the music service example, if *Alice* only listens to *rock*, the system will recommend her *rock* songs.

However, the same *cold-start* problem remains a challenge for new users who are yet to interact with the system. Again, no actions/items to match translate into no recommendations.

The Knowledge-Based system. This kind of system, in contrast to the previous, does not leverage past history. Instead, it leverages the actions/items knowledge. The idea is to improve or accelerate, the user's quest for something.

It is often used in scenarios where the history is irrelevant, for example buying a house. The current user's preferences are much more relevant than past behaviour or peer's houses.

Most importantly, this system does avoid directly the *cold-start* problem, but, it requires a bigger effort into categorising and defining attributes for the actions/items to be recommended.

In practice, most robust solutions include implementations of the three systems, which is a good decision since they do not overlap, they complement each other.

These 3 are not only the traditional *Recommender Systems* but also the baseline for most novel systems. Most improvements over the baselines enhance their implementation and/or features they use.

YouTube is one service that owns a very complex recommender system (Covington, Adams, & Sargin, 2016). Its system is hybrid, leveraging the three aforementioned techniques. It uses features obtained through deep learning techniques resulting in what they called: "a deep collaborative filtering model".

In practice, all the engineered features are concatenated into a fix-sized vector. The vector is then used as input for a *dimension reduction neural network* with *Rectifier Linear Unit* as activation functions (ReLU) (Glorot, Bordes, & Bengio, 2011). The architecture for this recommender system can be seen in *Figure 9*.

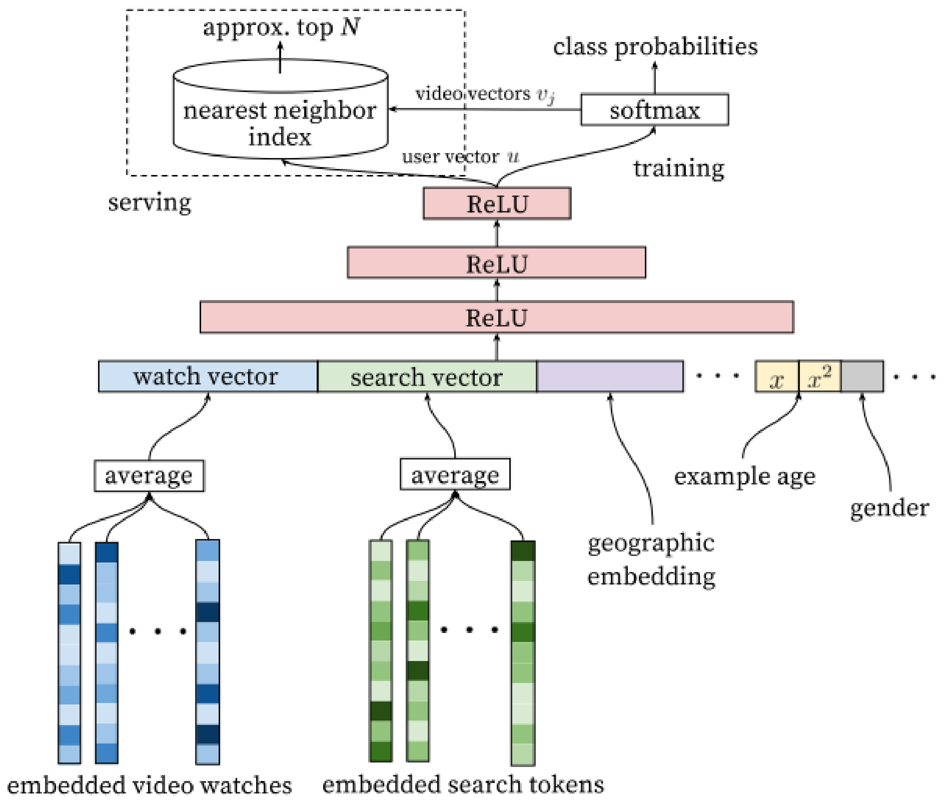


Figure 9: YouTube recommender system. Notice the features used and how they are encoded before reaching the ranking step (Covington, Adams, & Sargin, 2016)

Regarding recommender systems, good features are key. A further study based on the *YouTube*'s system focused its research on preserving the features' relevant information throughout the deep learning transformations (Beutel et al., 2018). The results achieved were empirically superior both in efficiency and accuracy.

Section 3.4: Evaluation Methods in Artificial Intelligence

AI algorithms are usually easily extendable and customisable, generating a vast variety of models. Hence, there is a need for comparison between alternative designs and baselines.

$$accuracy = \frac{Number\ of\ correctly\ predicted\ samples}{Total\ samples} \tag{4}$$

A very popular metric used for comparison is “accuracy”, seen in *Equation (4)*. Nevertheless, it might also be the most deceptive since it neglects both *variance* and *bias* in data. **Bias** describes the difference between real and predicted data while **variance** describes the difference between predictions resulting from different training sets. The selection of the best model is not entirely based on its performance but, also, on its generalisation capabilities. A model unable to understand new data is useless (Kohavi, 1995).

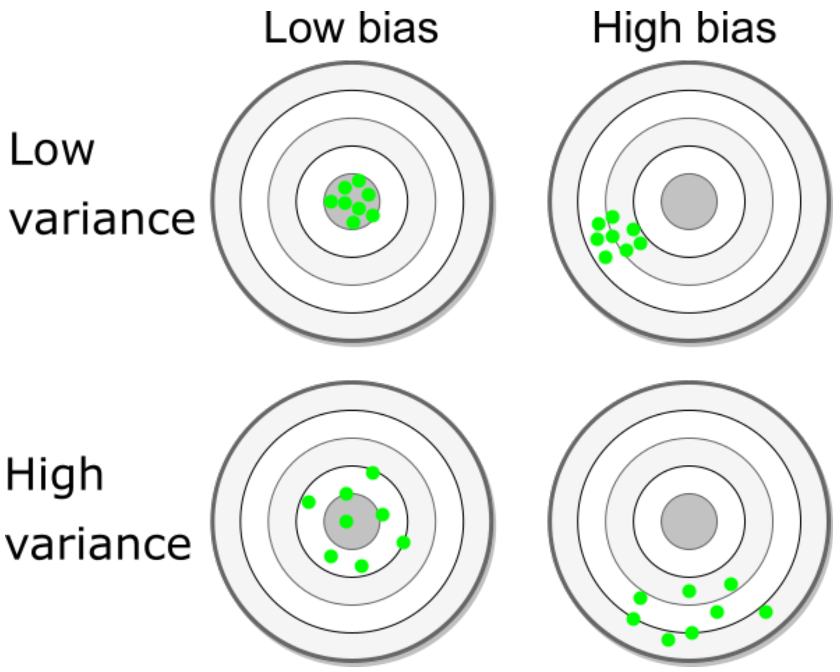


Figure 10: Bias vs Variance. The green points represent the predictions while the dartboards represent the proximity to the real values (Frank, 2017)

Neglecting bias and variance might rise the phenomena of *overfitting* or *underfitting*. **Overfitting** occurs when a model is complex enough to “memorise” each training sample. The result of it is a model with very low bias but very high variance. On the other hand, **underfitting** occurs when a model is not complex enough to create patterns with the training data. Its consequence is a model with high bias although its variance might be fairly low.

Before describing the standard metrics used in the comparison of models, the concept of *Confusion Matrix* must be first introduced. The **Confusion Matrix** is a table where all the predictions are compared with the real data and then classified (Caelen, 2017). There are four classifications: *True Positives*, *True Negatives*, *False Positives* and *False Negatives*. This matrix is presented in *Table 8*.

		Predicted data	
		Positive	Negative
Actual data	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Table 8: The confusion matrix

True Positives and **Negatives** are the classes assigned when the prediction matches the real data. The distinction between them is the value predicted. *True Positives* correctly identify the presence of a pattern while the *True Negatives* identify the absence of it.

False Positives and **Negatives**, contrarily, are the classes assigned when the predictions do not match the real data. The distinction, similarly, remains in the value predicted. If the erroneous prediction was the presence of a pattern, the classification would be *False Positive*, otherwise, it would be *False Negative*.

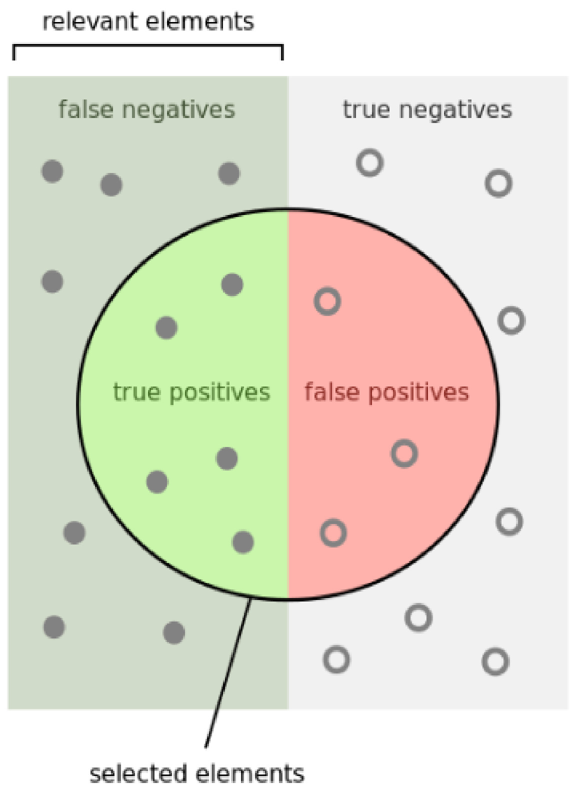


Figure 11: Visual representation of the confusion matrix's classes (Wikipedia Community, 2007)

These four concepts are the base for most standard metrics use to evaluate models in the AI field. These metrics are: *Precision* (also known as the *Positive Predictive Value,* Eq. 5), *Recall* (also known as *Sensitivity* or *True Positive Rate*, Eq. 6) and *F1score* (representing the harmonic average between the *precision* and the *recall* metrics, Eq. 7).

$$Precision=\frac{TP}{TP+FP} \tag{5}$$

$$Recall=\frac{TP}{TP+FN} \tag{6}$$

$$F1score = 2x \frac{1}{\frac{1}{precision} + \frac{1}{recall}} \tag{7}$$

However, as *variance* requires, several evaluations must be executed to obtain a good notion of the model's generalisation capability. This is crucial information to understand how the model will behave with new, unseen, data. The technique standardising this evaluation of generalisation is called: "Cross-validation".

This technique uses sampling methods to divide the population into two subsets: training and testing. The sampling is executed "k" times, each with different subsets. Every "k" time, the corresponding subsets are used to train and test a model. Lastly, the results of each iteration are summarised as the outcome of the *cross-validation* (Kohavi, 1995).

Varieties of cross-validation techniques exist, nevertheless, as recommended by (Kohavi, 1995), the *Stratified K-fold Cross-validation* technique with *k=10*, renders the best trade-off between *bias* and *variance*.

The differences between the cross-validation techniques are the sampling method. The one used by the *Stratified K-fold Cross-validation* is called: "Stratified Random Sampling". This method divides the population by its classes and then, in each iteration, samples separately from each class population. The class population samples are then joined together into a training subset for the current iteration and the rest are used as a testing subset. This method guarantees the ratio between classes is the same in all "k" evaluations. *Figure 12* presents a graphic representation of the sampling technique.

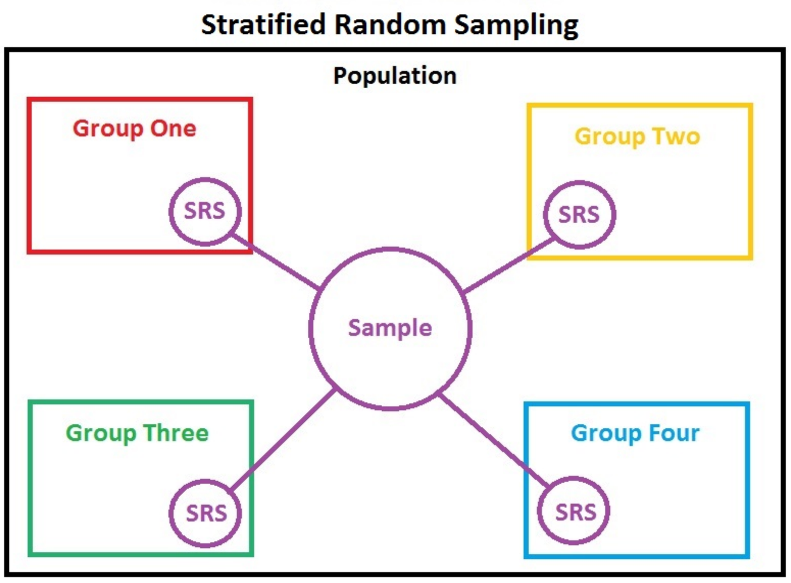


Figure 12: Stratified Random Sampling. Each group represents a different class (Wikipedia Community, 2001)

Chapter 4: System’s Solution

This chapter is dedicated to the description of the designed solution. The solution to be introduced is the result of the analysis done in the first semester towards the objectives of the project, and the implementation and experimentation of technology during the second. The solution is based on the project’s motivation, requirements and research presented in previous chapters.

The following sections present the technology used, the architecture of the system and the description of each module’s purpose, implementation and evolution.

Section 4.1: Technology

In order to build a solution, a collection of technologies capable to integrate with one another had to be selected.

The main programming language *Python 2.7* was chosen. The rationale behind this decision was the amount of experience the author has with it. Furthermore, *Python* is well-known for its ML and DL libraries that are easily integrated into an ongoing project. As a result, *Python* was used for all the implementation regarding algorithms, data manipulation, data management and request handling. The request handling functionality was implemented through *Flask*, an easy-to-use web application framework, ideal for the implementation of a prototype.

For the interaction and design of the web application, *HTML5*, *CSS*, *Bootstrap 4*, *JavaScript* and *jQuery* were used. Once more, the decision resides on the simplicity of use that would hasten development.

As persistence system, the databases used are *MongoDB*. It is a *NoSQL* database ideal for the execution of key-value queries, which is fitting for the designed solution presented in the next sections of this chapter (Chodorow & Dirolf, 2010). Moreover, the usage of a *SQL* database was not justified since no relationships between data wanted to be enforced or leveraged.

The remainder technology used, namely libraries, will be introduced in the following sections together with the explanation of their selection.

Section 4.2: The System’s Architecture

In *Figure 13* the system’s architecture is presented. The diagram describes the flow of data initiated by a request to the system.

The system is segmented into two separated subsystems exchanging data through a common database. The two subsystems are the *Advertisement Recommendation* and the *Advertisement Manager*.

Both the *IM Application* and the *Integration Layer* are not included in the project’s scope. Their presence in the diagram is for demonstration purposes only. The *IM Application* represents the client of an IM application and the *Integration Layer* an extension of the application prepared to communicate with the *Advertisement Recommendation API*.

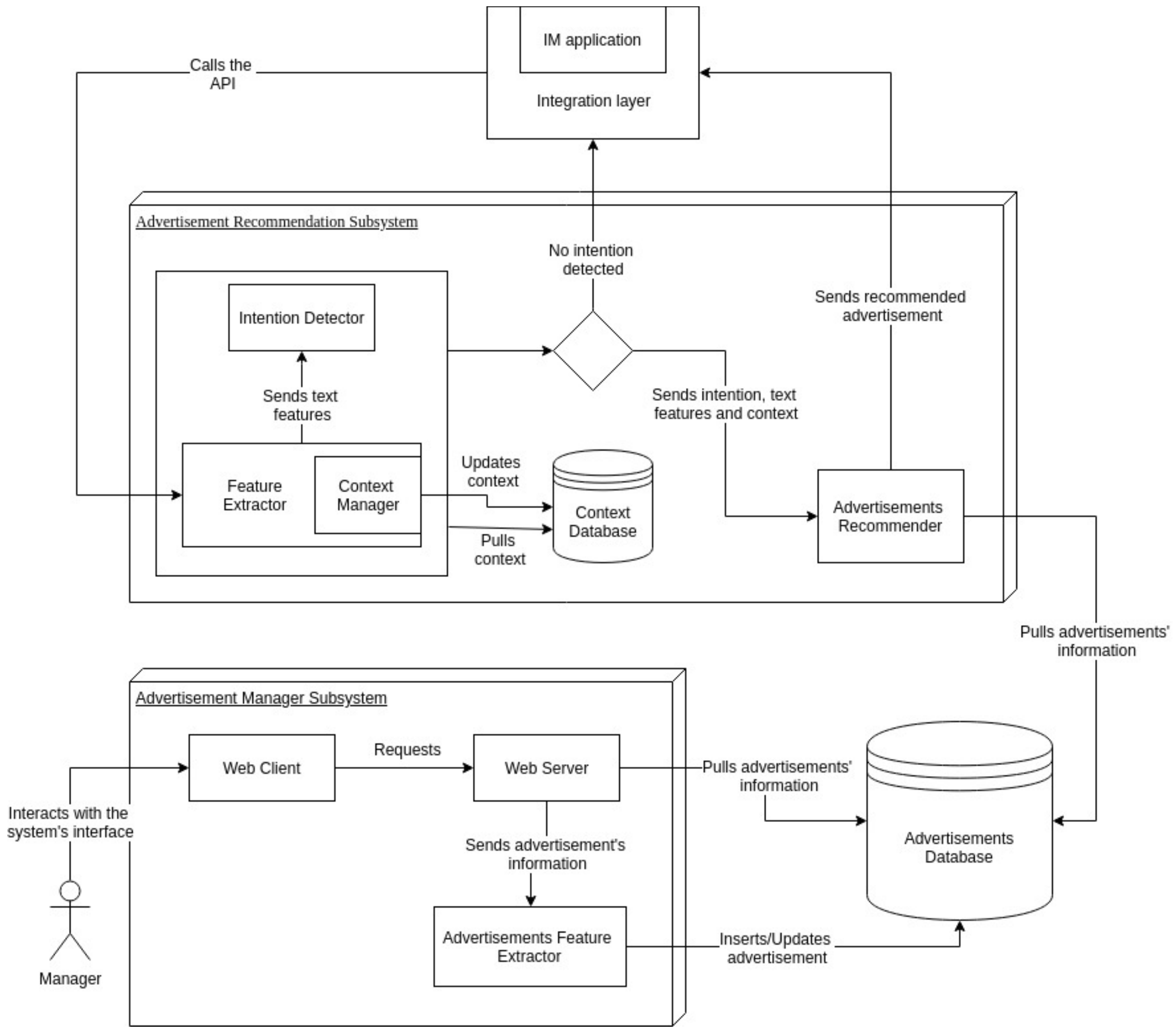

IM application
Integration layer
Calls the API
Advertisement Recommendation Subsystem
No intention detected
Intention Detector
Sends recommended advertisement
Sends text features
Sends intention, text features and context
Updates context
Feature Extractor
Context Manager
Context Database
Pulls context
Advertisements Recommender
Pulls advertisements' information
Advertisement Manager Subsystem
Web Client
Requests
Web Server
Pulls advertisements' information
Interacts with the system's interface
Sends advertisement's information
Advertisements Database
Manager
Advertisements Feature Extractor
Inserts/Updates advertisement

Figure 13: System’s architecture

Section 4.3: Advertisement Recommendation Subsystem

This subsystem is responsible for the *Advertisement Recommendation API* calls made from the IM applications. It must discriminate the sender’s identity, associate the request with previous calls, analyse the request and, finally, give a response with a recommendation if appropriate.

Given the subsystem’s objective, a set of modules were designed to divide the problem into smaller ones, following the ideology behind the well-known proverb: “Divide and conquer”. The modules in question are introduced in the subsections ahead.

4.3.1: Feature Extractor

This module encapsulates the extraction, cleaning and transformation functions to be used over the *raw text* received from the IM applications.

Its responsibilities are:

- The cleaning of raw text
- The extraction of text features
- The preparation of features to be sent to the *Intention Detector* and *Context Manager*

This module handles, mainly, data in text format. Hence, its cleaning is fundamental for the further usage of the data. In *Figure 14* the handling workflow of the *raw text data* is graphically explained. The validity of sentences rely on their length and content. Empty sentences and sentences containing exclusively single numbers are interpreted as invalid, hence, discarded.

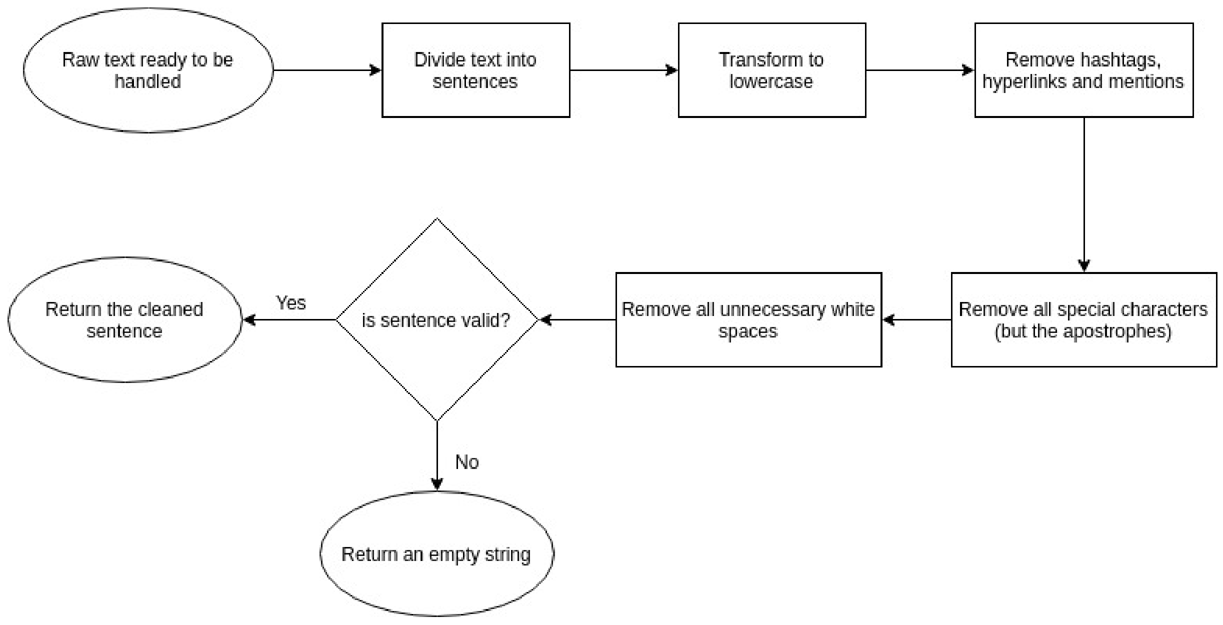

Raw text ready to be handled
Divide text into sentences
Transform to lowercase
Remove hashtags, hyperlinks and mentions
Remove all special characters (but the apostrophes)
Remove all unnecessary white spaces
is sentence valid?
Yes
Return the cleaned sentence
No
Return an empty string

Figure 14: Raw text treatment's workflow

Most cleaning and transformation steps were done through the use of pattern matching algorithms called: “Regular Expressions”. Nevertheless, the first step, the division of raw text into sentences, was done using the *SpaCy*’s library instead. The reason for this was the ambiguity detected in the segmentation of sentences originated from social media (or IM applications), which could not be effectively solved by pattern matching techniques (Rudrapal et al., 2015). Hence, a trained ML model was used instead. The library chosen was *SpaCy* due to its good benchmarking results and ease of use (spaCy, 2016b).

Once the information was properly cleaned, features could be extracted. These features are introduced in *Table 9*.

Feature	**Method of extraction**
Separation of sentences into individual words	Tokenisation using SpaCy
Named entities	NER using SpaCy
High-level part-of-speech tags	Coarse-grained POS using SpaCy
Verbs tags	Fine-grained POS using SpaCy
Sentence’s sentiment	Vader method using NLTK
Phrase’s embedding	Word embedding algorithms using the *Intention Detector* module

Table 9: Extracted text features

As mentioned previously, *SpaCy* is a NLP library with a good performance in traditional NLP benchmarks and simplified usage. Therefore, its selection as the text processing library. Regarding the extracted features, *Spacy*’s difference between *coarse* and *fine*-grained POS (Part-of-speech) tags is settled by the specificity of the tags. The *coarse-grained* POS uses the “Universal POS tags” defined in (Universal Dependencies, 2017), while the *fine-grained* uses the POS tags defined for the Penn Treebank project (Santorini, 1995)

Regarding the *sentiment analysis*, benchmarks were investigated to discover the better-performing algorithms. As seen in (Ribeiro, Araújo, Gonçalves, André Gonçalves, & Benevenuto, 2016), the *Vader* method is rarely the best performing method for the analysed datasets, however, it seems to be the better overall, having a consistently good performance across datasets when compared with the other methods.

The *verbs tags* feature is very relevant for the *Intention Detector* module. The reason relies upon the intentions themselves. Although there is a clear interest in detecting user intentions, the real interest is towards actual and future intentions. Past intentions are useless. No recommendation can be given. Consequently, negative reinforcement must be applied to sentences in past tense.

4.3.2: Context Manager

This module has an important yet not obvious purpose regarding *contextual information*. The detail with this type of information is its tendency to change periodically. Therefore, the relevance of this type of information fade over time. The objective of this module is to manage the fading of this information.

The responsibilities of this module are:

- The detection of *context* from features
- The persistence of context associated with a conversation ID in the *Context Database*
- The association of *context* with incoming conversation IDs
- The ageing/fading and renewal of *context*’s relevance
- The addition and update of *context*
- The retrieval of *context* associated with a conversation ID

The detection of location context. The *SpaCy*’s library provides a NER (Named Entities Recognition) service with the detection of location as GPE (Geographical Political Entities). However, after some empirical experimentation, the NER models had shown to be insufficient to detect locations. Cases where the name of the city was not capitalised or the city was not present in the corpus used to train the model, resulted in failure.

As a result, it was decided to build a gazetteer with the most important cities of the world and to use pattern matching techniques to deterministically detect cities in sentences. The gazetteer in question is presented in *Section 5.3: Other* *datasets*. This solution is possible due to the finite, and relatively small, amount of relevant cities in the world.

Nonetheless, an unaddressed problem emerged. Sentences where the cities are present but are not related to the user’s current location exist. If they appear, they will always be incorrectly detected. See *Table 10* for exemplifications of the problem.

Case	Is it correct?
"I will be in Coimbra next week"	True
"Are you in Leiria already?"	True
"I love Lisbon"	False
"My aunt lives in Oporto"	False

Table 10: Location context detection examples. Notice how the current method detects locations unrelated to the user's actual location

The detection of the location contextual information is the Achilles’ heel of the system, having a major drawback in detecting such important information.

The detection of date context. Leveraging the use of keywords, pattern matching techniques and a date manipulation library, a *date parser* was built. This *date parser* is able to receive text, extract date related information and use it to calculate relative dates. An example of its behaviour is presented in *Figure 15*.

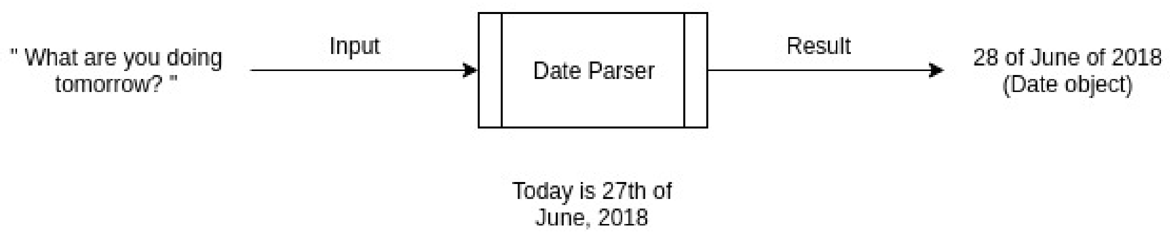

" What are you doing tomorrow? "
Input
Date Parser
Result
28 of June of 2018 (Date object)
Today is 27th of June, 2018

Figure 15: Date parser example

The detection of time context. The time context means to be extracted to comprehend “at what time″ the user desires to fulfil his/her intention. After a quick research about time references, it was concluded that the best way to approach this detection was by pattern matching techniques. Again, the amount of different values is finite, in this case extremely small. Moreover, unlike what happens with the location context, time references are rarely mentioned outside the desired context, therefore it is a more fitting solution.

Similarly to the detection of location context, the combination of a gazetteer and pattern matching techniques were used in the detection of time context. The gazetteer used is presented in *Section 5.3: Other datasets*.

The ageing algorithm. Contextual information often fades over time and the three types of contextual information used on this project are no exception. Information gathered at one point in time might not be true afterwards if enough time elapses. To simulate this effect, an *ageing algorithm* was implemented.

The idea is to, in addition to the context values, store their relevance bounded between “0” and “1”, and a timestamp of the last time the value was updated. Over time, it would be verified whether or not enough time has passed since the last update, a *time-lapse*. The variable *t* (number of time-lapses) is a discrete variable. For instance, if the *time-lapse* defined is of 1 day and 3 days had passed since the last update, the value of *t* would be “3”.

If enough time had passed since the last update, a new update is executed. The values remain the same, but, their relevance would “age” following a modified logistic function whose behaviour resembles the one of a logarithmic function. The *age* and *relevance* functions (see *Equations (8)* and *(9))* were designed to consider a parameter (*alpha*) to control the function’s growth. Furthermore, it was desired to have its growth diminished over time, ageing *t* in a logarithmic-like fashion. This design was sought due to the empirical conclusion of more active conversations having its *contextual information* changed quicker, which is represented by a higher decay rate for fewer *time lapses*.

The *age* function’s range is bounded between “0” and “0.9” for positive values of *t*. Notice that if many *ageing* steps occur the *relevance* can reach negative values.

$$Age(t,\alpha)=((\frac{1}{10})^{\frac{1}{1+\alpha*t}}-\frac{1}{10}) \tag{8}$$

$$Relevance(t,\alpha)=previousRelevance-Age(t,\alpha) \tag{9}$$

It is to be noticed that the three types of contextual information managed do not fade at the same pace. Therefore, different values of *alpha* (growth rate) were to be applied to each one. Furthermore, the periodicity (*time-lapse*) of their ageing is also different.

Context type	*alpha*	Time-lapse	Time-lapses until half faded	Ageing pace
Location	0.1	1 day	22 days	0.02328
Date	0.75	1 day	3 days	0.1682
Time	0.09	1 hour	24 hours	0.02093

Table 11: The ageing context configuration for each context type

Table 11 presents the configuration for each contextual information type. The location context ages slower than the other two, requiring *22 days* to fade under *0.5* relevance. On the opposite side, the time context is the fastest ageing, requiring only *24 hours* to fade under *0.5* relevance. The graphic behaviour of the parametrization of the ageing function can be seen in Figure 16.

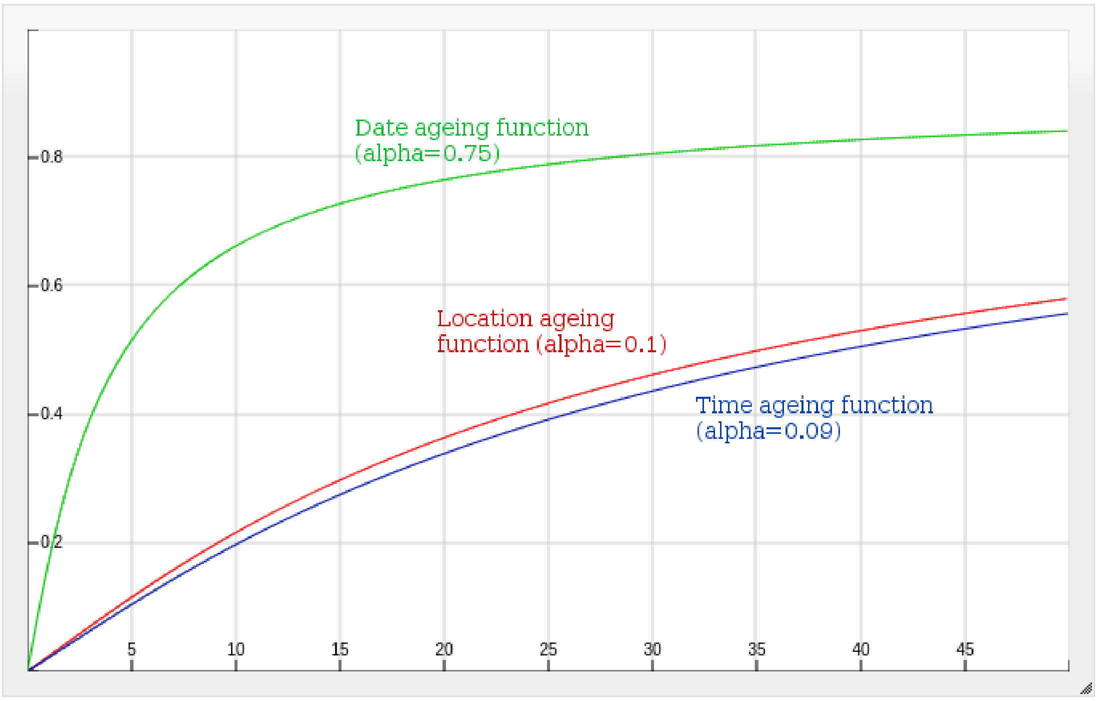


Figure 16: Parametrized age function. The ordinates represent the ageing value, while the abscissas represent the time lapses

Finally, The **Context Database** is a simple database designed for the storage of all contextual information by using the conversation IDs as primary keys. Its purpose is the persistence of context across API calls, which allows the association of stored context with requests. It also saves the context relevance and timestamps to support the contextual information *ageing algorithm*.

4.3.3: Intention Detector

This module analyses the features received from the *Feature Extractor* to determine whether an *intention* was expressed or not. More specifically, it consists of a binary machine learning classifier whose target is either “1” – an intention was detected – or “0” otherwise.

As seen in *Figure 13,* the result obtained from this module determines the subsequent actions of the subsystem. Either the *Advertisements Recommender* is invoked, or a response with no recommendation is returned to the requester.

The responsibilities of this module are:

- The detection of user intentions from features
- The inclusion of confidence in each classification
- The transformation of cleaned text into word/phrase embedding correspondingly

This module is purely built from AI algorithms. It is compounded by two different types of algorithms: word/phrase embedding algorithms (WPEAs), and machine learning algorithms (MLA) for binary classification of numeric data.

The Word Embedding algorithms. They are used to calculate the transformation of cleaned text into its equivalent vectorised form. The vectorised form – embeddings – are vectors of numbers with variable dimensions. These vectors are a key feature for the training of the machine learning algorithms.

Similarly to MLAs, the "No Free Lunch" theorem is present. Meaning, that no algorithms, or set of algorithms, is better than any other across all problems. Hence, different algorithms were studied and implemented in the seeking of the best embedding for the specific problem at hand.

Using the *Gensim* (Rehurek & Sojka, 2010) library and the *InferSent* and *GloVe* public implementations, several word embedding algorithms were implemented and tested. Furthermore, a common interface was built to uniformly train, test and use those algorithms.

The Phrase Embedding algorithms. Most techniques transforming text into vectors perform at the word-level. Nevertheless, since the input for this module is a sentence, a phrase-level transformation was sought. Word embedding algorithms were used by naively averaging all word embeddings contained in a sentence. However, other methods such as *Doc2Vec* and *InferSent* were specifically designed for this purpose, therefore, they were be used directly.

The Machine Learning algorithms. The role of these algorithms is to receive numerical features and use them to predict if there is any intention, or not. Their implementation was facilitated through the use of the *Scikit-learn* library (Pedregosa et al., 2012), which also simplified the implementation of the experimental setup for this module.

The experimentation and evaluation of the WE algorithms was done in cooperation with the ML algorithms. The experimental setup arranged different combinations of WE and ML algorithms, which were evaluated using the *Intention Detection Dataset* (introduced in *Section 5.2: The Intention Detection Dataset*). The experimental setup and results of this study reside in *Appendix B: The Intention Detection Experiments*.

In conclusion to the experimentation, the combination of **InferSent** (PE), **Polynomial SVM** (MLA) and the **verb tags** (text features) was selected to be the default model for the *Intention Detection* module.

4.3.4: Advertisements Recommender

This module is invoked exclusively when intentions are detected. It receives features and context – intention information – and uses it to rank advertisements. The ranking system is based on the proximity of the relation between the intention (represented by the features and context) and the advertisements (represented by their intrinsic information).

The responsibilities of this module are:

- The calculation of a matching measure between intentions and advertisements
- The ranking of advertisements regarding specific intentions
- The evaluation of matches proximity through a threshold mechanism
- The recommendation of the most proximal advertisements

This recommender system was based on a *Knowledge-based* recommender system. For the calculation of the matches, features from both the advertisements and intentions must be compared. Each comparison mechanism depends intrinsically on the features to be compared and how their comparison contribute towards the proximity between the intention and the advertisement associated. A total of 8 comparison mechanisms were implemented, each making a contribution to the final match’s proximity. The match’s proximity has a maximum value of “1”, having the following domain: *[0,1]*. The comparison mechanisms can be seen in *Table 12*.

The contextual information is very relevant for the recommendation. It would be nonsense to recommend a service located in a different city, or closed the day of the desired intention. Hence, these features have a higher contribution to the matching proximity.

Besides the verification of contextual information and proximity calculation of word (and phrase) embeddings, an additional type of comparison was made: the comparison by ontology’s semantic similarity. As a conclusion to the evaluation of different measures, the ontologies selected were *WordNet* and *YAGO*, with the *Wpath* and *Li* semantic similarities correspondingly. The details of the evaluation are presented in *Appendix C: The Similarity Measures Experiment*. The implementation of these measures was, also, simplified by the usage of a library called “Sematch” (Zhu & Iglesias, 2015).

After the calculation of the matches’ proximity, a simple ranking is done giving more importance to higher proximity values. Finally, the most proximal match is compared against the recommendation threshold set to *0.65*. If the threshold is not surpassed, no recommendation is done.

Relevantly, every situation when the threshold is not exceeded holds crucial information about the unsatisfied users’ intention. This information allows the manager of the system to identify trending needs, adapting the advertisement portfolio accordingly.

Intention feature	Advertisement feature	Comparison mechanism	Weight
Its location context	Its city	Verify whether the location and city are the same. The value is amplified by the context’s relevance	0.25
Its date context	Its working days	Verify whether the Google Place is open on the specified date. The value is amplified by the context’s relevance	0.2
Its time context	Its working hours	Verify whether the Google Place is open on the specific hours. The value is amplified by the context’s relevance	0.05
Word embeddings of its cleaned sentence	Word embeddings of its cleaned name	Calculate the cosine distance between each word embedding combination, and average the three higher values	0.1
Words from its cleaned sentence	Words from its cleaned name	Calculate the WordNet Wpath similarity between each word combination, and average the three higher values	0.1
Words from its cleaned sentence	Words from its cleaned name	Calculate the YAGO Lin similarity between each word combination, and average the three higher values	0.1
Phrase embedding of its cleaned sentence	Phrase embedding of its name	Cosine distance between the phrase embeddings	0.05
Phrase embedding of its cleaned sentence	Phrase embedding of its promotion	Cosine distance between the phrase embeddings	0.15

Table 12: Recommender system's matching features and comparison mechanisms

Section 4.4: Advertisement Manager Subsystem

This subsystem is responsible for handling the manager’s view of the system. It must allow *CRUD* (Create Retrieve Update and Delete) functionalities for the advertisements in an intuitive manner. Furthermore, since a graphical demonstration is within the requirements of the project, a User Interface (UI) must also be implemented for this purpose.

Similarly to the previously introduced subsystem, the *Advertisement Manager* was also divided into small modules. Withal, these modules are more traditional and easily understandable. The modules in question are introduced in the subsections ahead.

4.4.1: Advertisements Feature Extractor

This module extracts relevant information from advertisements and transforms it into features to be persisted at the *Advertisements Database*.

It is important to recall that, as a limitation of the scope presented in the *2.1.5: The limitation of the scope*, the *Google Places* service had to be used to obtain advertisements. Therefore, this module must also consume the *Google Places API* to acquire them.

This module responsibilities are:

- The retrieval of Google Places information through their *Google Place ID*
- The cleaning and transformation of Google Places’ information into features

The information pulled from the *Google Places API* was the following:

- The official name of the Google Place
- The place’s rating
- The place’s phone number and webpage
- The place’s city and exact address
- The place’s “types”, representing categories to which the place belongs
- The place’s working hours
- The place’s picture

Additionally, from these features, a *promotion* feature was also obtained from the UI, manually set. This *promotion* is to be used as a feature to further differentiate advertisements from each other. One example of a *promotion* could be: “Free drink with the purchase of a menu”.

Regarding the cleaning and transformation of the features. All text features were treated using the same method as the *Feature Extractor*. Also, the format for the place’s working hours given by the Google Places API is rather confusing, its format had to be significantly processed and adapted to a more clear and useful alternative.

4.4.2: Web Server

This module is a traditional component in software engineering that handles user requests received, executes the business logic of the system and manages the databases’ information.

For this project, its responsibilities are:

- The handling of user requests from the UI
- The management of advertisement’s information
- Host the *Advertisement Recommendation API*

In order to fulfil its responsibilities, the available endpoints and their usage had to be designed. This can be seen in *Table 13.*

Functionality	**Endpoint**	**Parameters**	**HTTP Method**
Navigation to the main page	“/” or “/index”	None	GET
Navigation to the add advertisement page	“/add”	None	GET
Addition of an advertisement	“/add”	Google Place ID and promotion	POST
Navigation to the inspect advertisement page	“/inpect/<id>”	Google Place ID	GET
Removal of an advertisement	“/inspect/<id>/ remove”	Google Place ID	POST
Navigation to the demonstration page	“/demo”	None	GET
Call to the *Advertisement Recommendation* system	“/demo”	Conversation ID and message	POST

Table 13: The Web Server endpoints

The response given by a POST request to the “/demo” endpoint is a JSON formatted text containing: the conversation identifier, the message sent, the context of the conversation, whether or not an intention was detected and the recommended advertisement if there was enough confidence.

Regarding the **Advertisements Database**, it was designed for the storage of advertisements’ information but, also, for the persistence of the advertisement’s performance. In other words, the history of recommendations is also saved, including the situations when no advertisement could be recommended due to the lack of confidence. This is valuable information for both the improvement of the system and for supporting business decisions regarding the advertisement’s portfolio.

4.4.3: Web Client

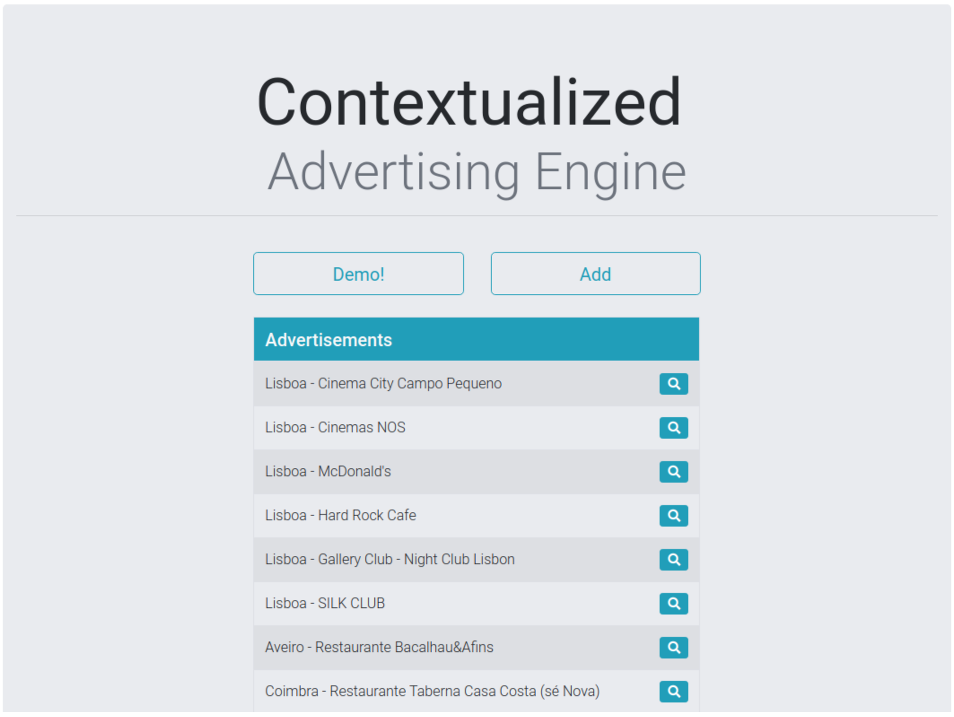
This module represents all the UI and interaction needed for both, the *Advertisement Manager* and the demonstration.

The responsibilities of this module are:

- The intuitive presentation of CRUD functionalities to the user
- The intuitive presentation of a demonstration of the system
- The interaction between the user and the *Web Server*

In order to fulfil its responsibilities, a series webpages had to be designed.

The Main Page. Mainly used to introduce the user with the system’s available functionalities in an intuitive manner. It presents three functionalities available through the UI: the addition of advertisement, the inspection of advertisement and the demonstration of the working system. In *Figure 17: System*'*s main page* the designed page can be appreciated.



Lisboa - Cinema City Campo Pequeno
Lisboa - Cinemas NOS
Lisboa - McDonald's
Lisboa - Hard Rock Cafe
Lisboa - Gallery Club - Night Club Lisbon
Lisboa - SILK CLUB
Aveiro - Restaurante Bacalhau&Afins
Coimbra - Restaurante Taberna Casa Costa (sé Nova)

Figure 17: System's main page

The Add Page. This page allows the addition of new advertisements with the use of *Google Places IDs*. These IDs are received to uniquely identify the Google Places to be fetched (from the Google Places API) and stored in the system. Additionally, a second field was added, the *promotion*. It is a text field with the purpose of adding a promotional campaign associated with the advertisement. In *Figure 18* the designed page can be appreciated.

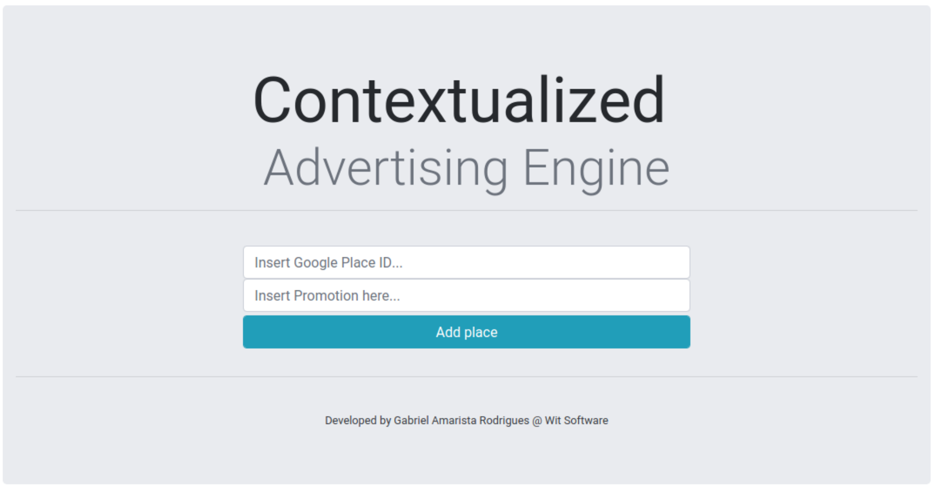
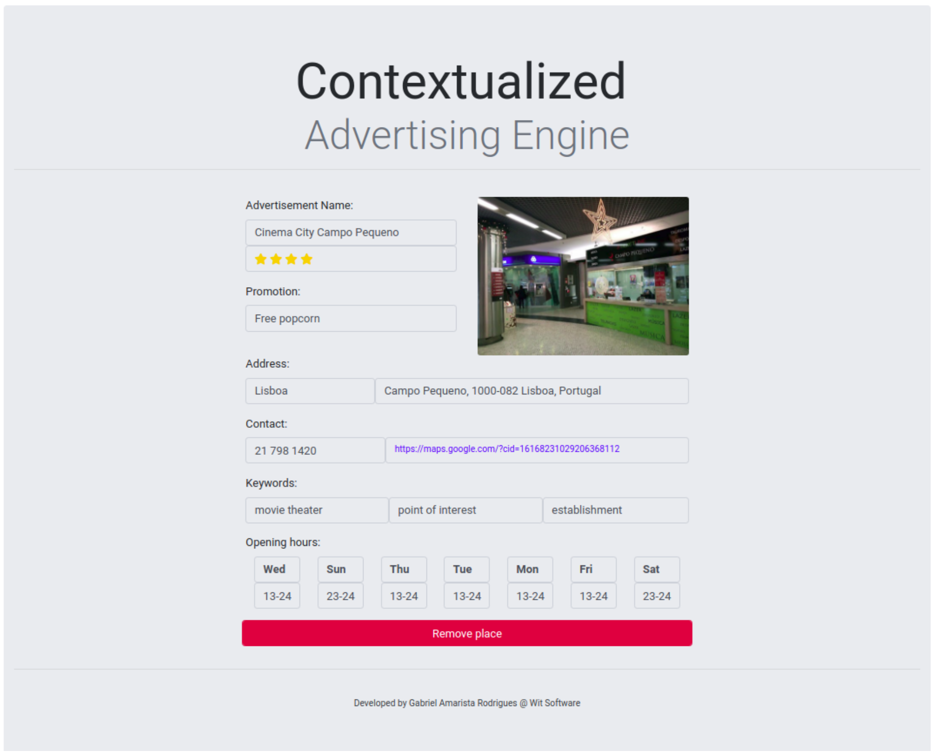

Contextualized
Advertising Engine
Insert Google Place ID...
Insert Promotion here...
Add place
Developed by Gabriel Amarista Rodrigues @ Wit Software

Figure 18: System's add advertisement page

The Inspection Page. This page is used for the visual presentation of advertisements' information. Since all the advertisements information is fetched from the Google Places API, it is important to present the actual information gathered. Additionally, the *remove* was added to facilitate the management of advertisements in any case of error. In *Figure 19* the designed page can be appreciated.


Contextualized
Advertising Engine

Advertisement Name:
Cinema City Campo Pequeno
★★★★

Promotion:
Free popcorn

Address:
Lisboa | Campo Pequeno, 1000-082 Lisboa, Portugal

Contact:
21 798 1420 | https://maps.google.com/?cid=16168231029206368112

Keywords:
movie theater | point of interest | establishment

Opening hours:

Wed	Sun	Thu	Tue	Mon	Fri	Sat
13-24	23-24	13-24	13-24	13-24	13-24	23-24

Developed by Gabriel Amarista Rodrigues @ Wit Software

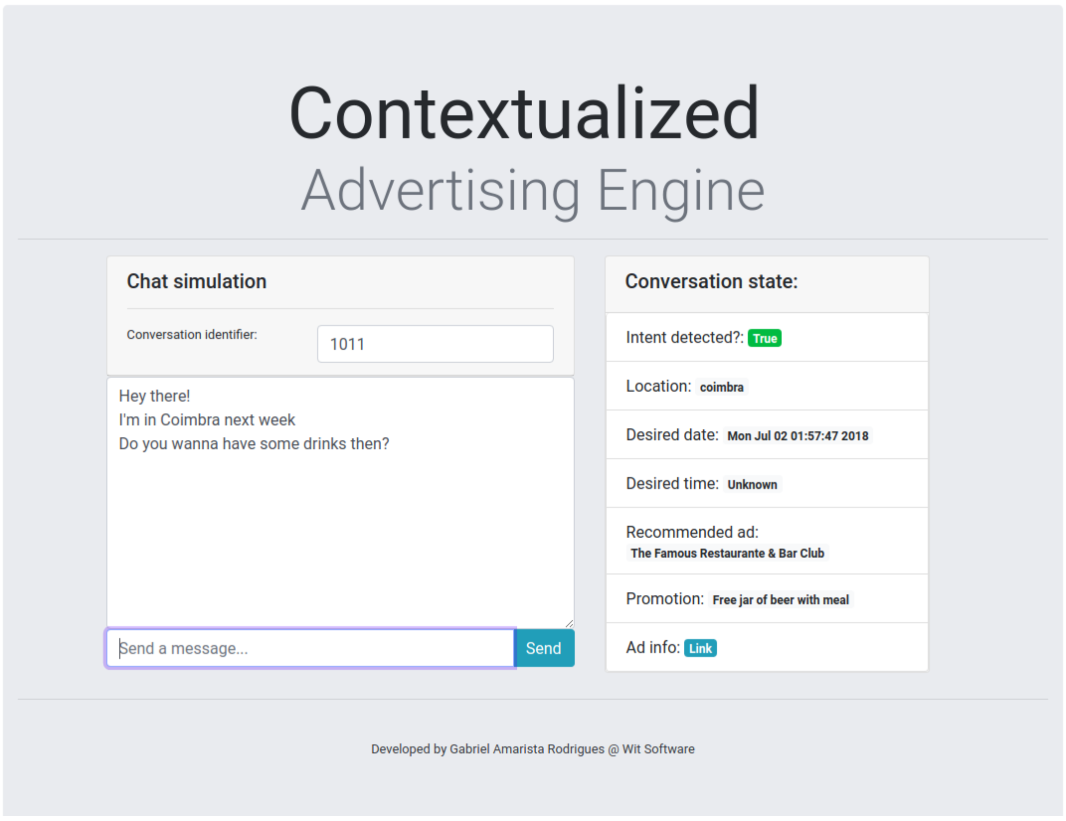
Figure 19: System's advertisement inspection page

The Demonstration Page. The most important page for the project. It includes both a *Chat Simulation* and a *Conversation State* division, each cooperating towards the same goal. The former handles the user’s input while the latter presents the system’s result and current state regarding to the current conversation. The conversation ID can be changed on the *conversation identifier* field.

The *Chat Simulation* allows the user to use a simple one-way IM application. Although only one person is involved in the conversation, it makes no difference towards the detection of intentions, or the demonstration of the system itself. Each message inserted is sent to the *Advertisement Recommendation API* and enters the workflow explained in *Figure 13*. This division also allows the user to switch to different conversations with different histories, facilitating the testing of different scenarios.

The latter division, the *Conversation State*, allows the user to, in real-time, observe the changes in the conversation's state after each message sent. The information included is: whether an intent was detected or not, the context of the conversation (location, day and time) and, finally, the recommended advertisement's information if a recommendation was in order.

In *Figure 20* the designed web page can be appreciated.



Contextualized
Advertising Engine

Chat simulation

Conversation identifier: 1011

Hey there!
I'm in Coimbra next week
Do you wanna have some drinks then?

Conversation state:

Intent detected?: True

Location: coimbra

Desired date: Mon Jul 02 01:57:47 2018

Desired time: Unknown

Recommended ad: The Famous Restaurante & Bar Club

Promotion: Free jar of beer with meal

Ad info: Link

Developed by Gabriel Amarista Rodrigues @ Wit Software

Figure 20: System's demonstration page

Chapter 5: Dataset Research and Construction

This chapter is dedicated to the specification of data needs, the investigation and evaluation of candidates to satisfy those needs and, finally, their adaptation and construction.

As explained in *Section 2.4: Risk Analysis*, the main dataset was built manually and supported by a harvested dataset also built during the project's time frame. The process of their construction is the main focus of this chapter.

Section 5.1: The Twitter dataset

Before attempting to solve any problem, one must first understand the problem itself. Specifically for this project, the understanding of the nature of the text to be analysed was very important and much was studied. Nevertheless, real data was not given and had to be found to further comprehend the challenge imposed.

The data to be found had specific characteristics introduced in *Subsection 2.1.2*. In the research for datasets that fulfilled aforementioned characteristics, several were found but none with the right qualities. In *Appendix D: Dataset Research,* the analysis of the encountered datasets is presented. Ultimately, useful datasets were not found and, conclusively, the need to harvest one was imperative. The data was to be harvested from Twitter, a decision that is also explained in the aforementioned appendix as a conclusion for the dataset research.

The first step towards the construction of the Twitter dataset was the creation of a specific word base to extract domain-specific tweets. The word base contains words related to the use cases, e.g. “restaurant”, “cafeteria”, “coffee”, “movies”, etc. A total of 115 words were added.

Once the word base was prepared, the harvesting strategy inspired by (Hu et al., 2018) was implemented. It consisted of the following steps:

1. Build an initial population of users
    a. Use the domain-specific word base to find tweets
    b. Extract the users from the tweets
2. Iterate over each user in the population
    a. Extract the 5 more recent tweets
    b. Extract 2 random followers
3. Substitute the original population by the extracted followers
4. Repeat the process from step 2

This strategy retrieves both domain specific and diverse data. The idea behind the substitution of the populations of users is the avoidance of bias introduced by the users' specific behaviour on Twitter.

The result of the harvest was separated on 3 different datasets, one containing, exclusively, domain-specific tweets, another containing *reply* tweets and the third one with all the tweets harvested.

The implementation of the harvester was done using the **Twython** API as an abstraction of the Twitter's API.

During the development and analysis of the harvest, some negative characteristics, in tweets and Twitter users, were detected and removed by filtering steps. Also, as a reaction to the amount of spam found, a quick research was done regarding the detection of fake Twitter users (Nimmo & Defense at the Atlantic Council’s Digital Forensic Research Lab, 2017). Those characteristics were also added to the filtering mechanism.

Table 14 presents all filtered situations and the reason for their disposal. The filtering steps presented were possible due to the metadata received from Twitter associated with both tweets and Twitter users.

Entity affected	**Negative characteristic**	**Reason of disposal**
User	Private users	These users cannot be queried
User	Deleted users	These users cannot be queried
User	Few followers (< 50)	High probability of being spam accounts
User	Has its account with a language different than English	The dataset pursued is to be in the English language
User	Repeated users	Avoidance of bias introduction by users' specific behaviour
User	Few friends (< 80)	High probability of being spam accounts
User	Few tweets (< 100)	There is a probability of being either spam or dead accounts
User	Few favourites (<20)	High probability of being spam accounts
User	Absence of a profile banner	High probability of being spam accounts
Tweet	Is written is a different language than English	The dataset pursued is to be in the English language
Tweet	Is a retweet	Avoid redundant tweets
Tweet	Repeated tweet	Avoid redundant tweets
Tweet	Is similar to stored tweets	Avoid redundant tweets

Table 14: Filtered negative characteristics found in Twitter's data

The harvesting velocity achieved is of **110,15** tweets per hour. The small amount is associated to the Twitter API's rate limiter that restrains the number of requests an account can send per minute, the values variate depending on the type of request (Twitter, n.d.).

The statistics of the results obtained by the harvester are presented in *Table 15*.

Statistic	**Value**
Total users analysed	49857
Total tweets stored	91217
Total domain tweets stored	4980
Total reply tweets stored	27465
Total domain words	115

Table 15: Twitter harvester results

Section 5.2: The Intention Detection Dataset

The dataset to be presented in this section is the source of training and testing for the *Intention Detector* classifier, hence, its name.

The data pursued needed to reflect the absence and presence of users' intentions regarding the specific use cases of the project. The formatting of the dataset is fairly simple, 2 columns, one for the text and one for the class. The classes were two (binary), *"0"* for the absence of an intention and *"1"* for the presence of it.

Posteriorly to the first successful harvesting iterations of the *Twitter dataset*, those tweets were analysed to extract entries for the *Intention Detection* dataset.

Although negative cases (class = 0) were common, it was difficult to find perfect tweets containing intentions. Many tweets containing unrelated information were found, which led nowhere. The reason might be under the fact that Twitter is, also, a social network where people interact with strangers. Strangers who one would never invite to "go for a drink". Furthermore, many enterprises doing advertising of their products and services are often found.

Consequently, a different strategy was used. Instead of reading all tweets harvested, read only the *domain specific* and *reply* tweets. This strategy highly improved the encountering of intentions, which is rather obvious since *reply* tweets resemble the most to IM conversations and the *domain-specific* tweets carry essential keywords to detect them.

However, most positive entries in the dataset were not *entirely* found. Mostly, the tweets contained words or expressions that could be leveraged to write different phrases containing intentions. Turning the process completely almost manual.

The evolution of the dataset is presented in *Figure 21*. The current state of the dataset has a total of **1018** entries with its classes perfectly balanced (50-50).

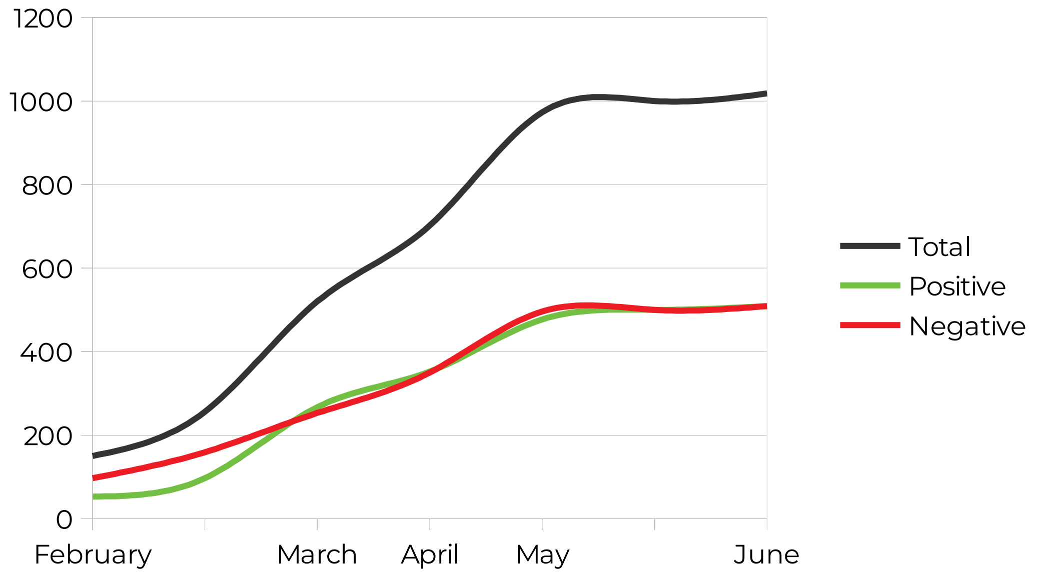


Figure 21: Intention Detection Dataset's evolution

Section 5.3: Other datasets

In addition to the main datasets introduced previously in the actual chapter, other datasets were also researched for multiple purposes. The introduction of those dataset is the purpose of this section.

The Google places dataset. It is a small collection, 27 entries, of Google Places IDs that were manually chosen using the Google Places ID Finder (Google Maps Platform, 2018). These identifiers were collected to be used both in the evaluation and demonstration of the system.

The Reddit Comments May 2015 dataset. It consists, as it name establishes, in a collection of comments data gathered from Reddit. The whole dataset contains data from October 2007 to May 2015, approximately 1.7 billion comments and over 1 Terabyte uncompressed. However, the dataset actually used in this project was a small subset of 170 Megabytes, 1.9 million comments.

This specific data carries informal writing patterns, patterns often found in social media and IM communications. Hence, this dataset was used to train the word embedding algorithms. The Twitter dataset was initially used but had shown to be rather small for the task.

The time gazetteer. It is a very small collection, manually built, of daytime related keywords inspired by the discussion in (Mairs, 2012). Its usage is mainly to support the detection of time context within a text conversation. In *Table 16* the gazetteer is presented.

Time	**Minimum hour**	**Representative hour**	**Maximum hour**	**Keywords (comma separated)**
Morning	5	8	12	Morning, Breakfast, Early
Afternoon	12	14	17	Afternoon, Brunch, Lunch, Tea
Evening	17	18	20	Evening
Night	20	22	5	Night, Tonight, Dinner, Party, Club

Table 16: The Time Gazetteer

The number of keywords is scarce. Nevertheless, it is easily extendable and covers most of the situations encountered.

The cities gazetteer. This gazetteer consists of the collection of 346 cities chosen as the most relevant cities in the world. The raw data was taken from *MaxMind* cities' database (MaxMind, 2008) to be later simplified and filtered.

This gazetteer is used to support the detection of location context within a conversation. A pattern matching approach was preferred over the Named Entity Recognition (NER) APIs that often failed to successfully detect cities.

Chapter 6: Project’s Success Analysis

This chapter is dedicated to the comparison of the built system against the threshold of success defined in *Section 2.2: Requirements and Thresholds of Success*. The threshold of success statements were the following:

1) All *must have* requirements must be fulfilled
2) The system must approach all proposed use cases
3) The *Intention Detection* algorithm must reach an *F1-score* superior to 75%
4) After a call to the *Advertisement Recommendation* API, a response must be given in less than 1.5 seconds
5) The prototype must have a graphical demonstration of the system and idea behind it

The *thresholds of success* number **3** and **5** were both mentioned previously. Their fulfilment can be seen in *Appendix B: The Intention Detection Experiments* and *4.4.3: Web Client* respectively. The *Intention Detector*, achieved an *F1-score* of **85.62%**, surpassing by more than 10% the threshold established.

Regarding the *thresholds of success* number **2** and **4**, specific experiments had to be designed in order to verify them. The design of the experiments and their results can be seen in *Appendix E: The Working System Experiment* and *Appendix F: The API Response Time Experiment* respectively. The response time of the *Advertisement Recommendation API* measured was of **0.268 seconds**, fulfilling successfully the threshold established. Relatively to the use cases approached, the working system experiment shows the evaluation of 15 different conversations that represent all the use cases defined.

Lastly, the *threshold of success* number **1** is the broader of them all, covering situations mentioned on the other thresholds and adding its own. In *Table 17* the analysis of the achievement of this threshold is presented.

Although not all the requirements were fulfilled, all *must have* requirements were, indicating the fulfilment of the *threshold of success* **1**. The requirements left undone were not considered essential and demanded resources that were not available, namely, time.

In conclusion, due to the completion of all project’s thresholds of success, it can be said that the current project was successfully executed by achieving the guidelines imposed for it.

Nº	Requirement	Was fulfilled?
1	The system **MUST** be able to work with English conversations	Yes. The datasets and resources used were specifically build for the English language
2	The system **MUST** extract relevant features from the conversations	Yes. Both phrase vectors and text features were extracted, evaluated and used
3	The system **MUST** detect and discriminate the user's intentions/needs/desires	Yes. This is the purpose of the *Intention Detector* module
4	The system **MUST** detect, extract and manage the context of the conversations	Yes. This is the purpose of the *Context Manager* module
5	The system **MUST** recommend advertisements based on the conversation's most recent intention, text features and context	Yes. This is the purpose of the *Advertisement Recommender* module
6	All information **MUST** be persisted in a database	Yes. Both the *Context* and *Advertisements* databases were designed with this purpose
7	The system **MUST** be available for consumption via an API	Yes. This is the purpose of the *Advertisement Recommendation API*
8	The system's API **MUST** be capable to identify conversations within calls	Yes. This is the purpose of the *conversation ID* gathered on the request
9	The system's API **MUST** always return a response	Yes. The alternative workflow is shown in system's architecture
10	The system's API response **MUST** be within the user's conversational context	Yes. Related to the previously mentioned *threshold of success* **4)**

Nº	Requirement	Was fulfilled?
11	The system **MUST** have a visual interface to manage advertisements	Yes. This is the purpose of the *Web client* module
12	The prototype **MUST** have a visual interface to demonstrate the system in action	Yes. Related to the previously mentioned *threshold of success* **5)**
13	Datasets **MUST** be developed to train and test the system	Yes. As introduced in Chapter 5: Dataset Research and Construction
14	All Artificial Intelligence algorithms **MUST** be evaluated	Yes. *Intention Detector* was evaluated and the AI APIs used had their benchmarks checked
15	The system's performance **MUST** be evaluated	Yes. Related to the previously mentioned *thresholds of success* **2)** and **3)**
16	Alternative algorithms **SHOULD** be evaluated	Yes. This is the purpose of the *Intention Detection Experiments*
17	Different tools and APIs **SHOULD** be evaluated	Yes. Mostly based on benchmark results
18	The system **SHOULD** detect and persist impossible recommendations	Yes. This is one of the responsibilities of the *Advertisements Database*
19	The system's performance **COULD** be monitored	Not fulfilled

Table 17: Success analysis: requirements verification

Chapter 7: Future Work and Conclusions

This chapter is dedicated to the reflection of the author regarding the evolution and final status of the system. Possible approaches that could be taken in order to improve its performance are also introduced.

As the process advanced, from the design to the implementation of the system, the complexity within each of the modules was made more evident. During the whole process, several flaws and potential fixes were found along the way. Nevertheless, due to the amount of work already necessary to achieve the thresholds of success, much was left to be optimized in the future.

As the famous proverb says: *"A chain is no stronger than its weaker link"*.

The immediate future effort should be allocated to the interpretation of implicit context and the enrichment, with more metadata, of the advertisements available. These two had shown to be the weakest links in the chain.

The enrichment of the advertisements might be a relatively easy task. Using the APIs of services such as *Trivago* or *TripAdvisor*, or even waiting for the Google Places API to make available more information about the places, will be enough to guarantee an improved performance. However, the case of the context flaw is much more complex.

The IM conversations have a channel where implicit information is omnipresent. Therefore, expecting to always encounter the context information the system needs, is to be unreasonably optimistic. Getting support from information already linked to the user would be highly useful and would facilitate this process. Following the path of monitoring and storing users' preferences, would also improve the recommendations since there would be more information to further filter the advertisements collected.

Yet, these might not be the only weak links in the system. The fact that the *Intention Detector* was trained with a dataset made by a single annotator, introduces uncertainty about the quality of the dataset. Besides it being extremely small for the use case in question, it may possess biases related to the author’s writing style and his perspective towards the kind of examples that should be included in the dataset.

More annotators should be included in the evolution of the dataset. A good alternative would be to leverage services such as *CrowdFlower* to both expand the dataset and test the overall system. Techniques of data augmentation, based on synonyms and categories of words, can also be used to expand the dataset.

The *Intention Detector* has yet multiples approaches to be explored. New techniques challenging the state-of-the-art of word and phrase embeddings have emerged since the beginning of this project and will continue to do so. Moreover, several different binary classifiers are yet to be tested. However, due to its good performance, this might be the module that requires less attention.

One relevant mistake made at the start of the implementation process was the selection of Python 2.7 as the version of Python to develop the system. For future references, Python 2.7 has a huge problem handling U*nicode* text since U*nicode* is not the default text representation, but *str* (a sequence of bytes). This specific detail raises several issues when handling text coming from different encodings and would have been seamless if Python 3 had been chosen.

During this project, not only its complexity was made evident, but also the complexity of language. It is incredible how we are able to dominate such topic yet we fail to effectively teach it to computers. Moreover, it is truly surprising how we can communicate effectively regardless of the quantity of implicit information and ambiguous statements (which can also be implicit).

Natural language understanding seems to be an extremely hard problem. However, we might be approaching it wrong. As Percy Liang said:

“Language understanding is not about the words, but about the world. It is interactive, communication demands interaction and consensus.”

Perhaps, in order to effectively teach it to computers, we must first allow them to interact with the world as we do.

Bibliography

Aggarwal, C. C. (2016). An Introduction to Recommender Systems. In *Recommender Systems, The Textbook* (pp. 1–29). https://doi.org/10.1007/978-3-319-29659-3

Baumgartner, J. (2015). 1.7 billion Reddit comments [Dataset]. Retrieved 18 November 2017, from https://archive.org/details/redditcomments

Beutel, A., Covington, P., Jain, S., Xu, C., Li, J., Gatto, V., & Chi, E. H. (2018). Latent Cross: Making Use of Context in Recurrent Recommender Systems. *WSDM 2018: The Eleventh ACM International Conference on Web Search and Data Mining*, 46–54. https://doi.org/10.1145/3159652.3159727

Bois, D., W, J., Chafe, W. L., Meyer, C., Thompson, S. A., Englebretson, R., & Martey, N. (2005). Santa Barbara corpus of spoken American English, Parts 1-4 [Dataset]. Retrieved 7 November 2017, from http://www.linguistics.ucsb.edu/research/santa-barbara-corpus#Citing

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016a). Download pre-trained word vectors [Dataset]. Retrieved 19 March 2018, from https://github.com/facebookresearch/fastText/blob/master/docs/english-vectors.md

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016b). Enriching Word Vectors with Subword Information. https://doi.org/1511.09249v1

Boundless Psychology. (n.d.). Introduction to language. Retrieved 15 June 2018, from https://courses.lumenlearning.com/boundless-psychology/chapter/introduction-to-language/

Caelen, O. (2017). A Bayesian interpretation of the confusion matrix. *Annals of Mathematics and Artificial Intelligence*, *81*(3–4), 429–450. https://doi.org/10.1007/s10472-017-9564-8

Carr, M., Konda, S., Monarch, I., Ulrich, F., & Walker, C. (1993). Taxonomy-based risk identification. *Software Engineering Institute*, (June), 1–24. https://doi.org/CMU/SEI-93-TR-006

Carty, F. (2017). The Evolution of the Ad. Retrieved 24 February 2018, from https://blog.prototypr.io/the-evolution-of-the-ad-5e5687d5fd53

Chen, T., & Kan, M.-Y. (2013). Creating a Live, Public Short Message Service Corpus: The NUS SMS Corpus. *Language Resources and Evaluation*, 299–355. Retrieved from http://link.springer.com/article/10.1007%2Fs10579-012-9197-9

Chodorow, K., & Dirolf, M. (2010). *MongoDB: The Definitive Guide* (1st ed.). O'Reilly Media, Inc.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., & Bordes, A. (2017). Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. *Empirical Methods in Natural Language Processing (EMNLP)*. Retrieved from https://arxiv.org/pdf/1705.02364.pdf

Constine, J. (2015). Facebook Is Adding A Personal Assistant Called 'M' To Your Messenger App. *TechCrunch*. Retrieved from https://beta.techcrunch.com/2015/08/26/facebook-is-adding-a-personal-assistant-called-m-to-your-messenger-app/?_ga=2.228168093.1831809675.1529789123-411323904.1529789123

Covington, P., Adams, J., & Sargin, E. (2016). Deep Neural Networks for YouTube Recommendations. *Proceedings of the 10th ACM Conference on Recommender Systems - RecSys '16*, 191–198. https://doi.org/10.1145/2959100.2959190

Craddock, A., Fazackerley, B., Messenger, S., Roberts, B., & Stapleton, J. (2008). MoSCoW Prioritisation. In *DSDM Atern: The Handbook* (2nd ed., p. 201). DSDM Consortium. Retrieved from https://www.agilebusiness.org/content/moscow-prioritisation-0

Danescu-Niculescu-Mizil, C., & Lee, L. (2011). Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs [Dataset]. *ACL*. Retrieved from http://www.cs.cornell.edu/~cristian//Cornell_Movie-Dialogs_Corpus.html

Dillet, R. (2018). Facebook is shutting down its standalone personal assistant 'M'. *TechCrunch*. Retrieved from https://techcrunch.com/2018/01/08/facebook-is-shutting-down-its-standalone-personal-assistant-m/

Drift, Audience, SalesForce, & myclever. (2018). The 2018 State of Chatbots Report. Retrieved from https://blog.drift.com/wp-content/uploads/2018/01/2018-state-of-chatbots-report.pdf

Dsquared Media. (n.d.). Contextualized Ad Design. Retrieved 24 February 2018, from https://www.dsquaredmedia.net/creative-design-services/contextualized-ad-design/

Elavarasi, S. A., Akilandeswari, J., & Menaga, K. (2014). A Survey on Semantic Similarity Measure. *International Journal of Research in Advent Technology*, *2*(3), 389–398.

Englishpage.com. (n.d.). Subjunctive. Retrieved 19 October 2017, from https://www.englishpage.com/minitutorials/subjunctive.html

Evgeniy, G. (2002). The WordSimilarity-353 Test Collection [Dataset]. Retrieved 9 April 2018, from http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/

Forsyth, E., Lin, J., & Martell, C. (2010). NPS Internet Chatroom Conversations, Release 1.0 [Dataset]. Retrieved 7 November 2017, from https://catalog.ldc.upenn.edu/LDC2010T05

Gallegos, J. A. (2016). The History and Evolution of Advertising. Retrieved 11 October 2017, from https://www.tintup.com/blog/history-evolution-advertising-marketing/

Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. *AISTATS '11: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, *15*, 315–323. https://doi.org/10.1.1.208.6449

Google Code. (2013). word2vec [Dataset]. Retrieved 18 March 2018, from https://code.google.com/archive/p/word2vec/

Google Maps Platform. (2018). Find the ID of a particular place. Retrieved 18 May 2018, from https://developers.google.com/places/place-id#find-id

Hoy, G. (2014). Why Contextual Advertising Is Still Hard. Retrieved 24 February 2018, from https://www.targetmarketingmag.com/post/why-contextual-advertising-is-still-hard/all/

Hu, Y., Zheng, L., Yang, Y., & Huang, Y. (2018). Twitter100k: A Real-World Dataset for Weakly Supervised Cross-Media Retrieval. *IEEE Transactions on Multimedia*, *20*(4), 927–938. https://doi.org/10.1109/TMM.2017.2760101

Hutto, C. J., & Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Eighth International AAAI Conference on Weblogs and …*, 216–225. Retrieved from http://www.aaai.org/ocs/index.php/ICWSM/ICWSM14/paper/view/8109%5Cn http://comp.social.gatech.edu/papers/icwsm14.vader.hutto.pdf

Khalaf, S. (2017). On Their Tenth Anniversary, Mobile Apps Start Eating Their Own. Retrieved 24 February 2018, from http://flurrymobile.tumblr.com/post/155761509355/on-their-tenth-anniversary-mobile-apps-start

Kohavi, R. (1995). A Study of Cross-Validatin and Bootstrap for Accuracy Estimation and Model Selection. *International Joint Conference on Artificial Intelligence*. Retrieved from http://web.cs.iastate.edu/~jtian/cs573/Papers/Kohavi-IJCAI-95.pdf

Lafferty, J., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning*, *8*(June), 282–289. https://doi.org/10.1038/nprot.2006.61

Le, Q. V., & Mikolov, T. (2014). Distributed Representations of Sentences and Documents. *CoRR*, *32*. https://doi.org/10.1145/2740908.2742760

Leskovec, J., & Krevl, A. (2014). SNAP Datasets: Stanford Large Network Dataset Collection. Stanford University. Retrieved from https://snap.stanford.edu/data/twitter7.html

Li, Y., Bandar, Z., & A. McLean, D. (2003). An Approach for Measuring Semantic Similarity between Words Using Multiple Information Sources. *IEEE Transactions on Knowledge and Data Engineering*, *15*.

Liang, P. (2017). *Natural Language Understanding: Foundations and State-of-the-Art [Video file]*. Simons Institute. Retrieved from https://l.messenger.com/l.php?u=https%3A%2F%2Fwww.youtube.com%2Fwatch%3Fv%3DmhHfnhh-pB4&h=AT3WfDCt7We3E9ZDpn9aeFwaYDYk3ky67xlTQ4Egm3xaxDzcdcAhPOtkTMwn4Higa8H7BSxDzYrMFMxvwTyqYW_HNoMvC7DhaSEQWkplKq935v1G0yeznWJFrV8N2w

Lin, D. (1998). An Information-Theoretic Definition of Similarity. *Proceedings of ICML*, 296–304. https://doi.org/10.1.1.55.1832

Liu, H., & Singh, P. (2004). ConceptNet - a practical commonsense reasoning tool-kit. *BT Technology Journal*, *22*(4), 211–226. https://doi.org/10.1023/B:BTTJ.0000047600.45421.6d

Ma, X., & Hovy, E. (2016). End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, *1*, 1064–1074. https://doi.org/10.18653/v1/P16-1101

Mairs, J. (2012). Parts of the Day: Early morning, late morning, etc. Retrieved 18 April 2018, from http://www.learnersdictionary.com/qa/parts-of-the-day-early-morning-late-morning-etc

Marketing Magazine. (2016). The Future of Messaging Apps. Retrieved 13 October 2017, from http://www.marketing-interactive.com/future-messaging-apps/

MaxMind. (2008). Free World Cities Database [Dataset]. Retrieved 18 April 2018, from https://www.maxmind.com/en/free-world-cities-database

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2017). Advances in Pre-Training Distributed Word Representations. *CoRR*. Retrieved from https://arxiv.org/pdf/1712.09405.pdf

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. *NIPS'13 Proceedings of the 26th International Conference on Neural Information Processing Systems, 2,* 3111–3119. Retrieved from https://arxiv.org/pdf/1310.4546.pdf

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, *38*(11), 39–41. https://doi.org/10.1145/219717.219748

Miller, G. A., & Charles, W. G. (1991). Contextual correlates of semantic similarity. *Language and Cognitive Processes*, *6*(1), 1–28. https://doi.org/10.1080/01690969108406936

Moltmann, F. (2016). Natural Language Ontology, 1–27. Retrieved from http://friederike-moltmann.com/uploads/Natural Language Ontology-2016(3).pdf

Nimmo, B., & Defense at the Atlantic Council's Digital Forensic Research Lab. (2017). #BotSpot: Twelve Ways to Spot a Bot. Retrieved 22 February 2018, from https://medium.com/dfrlab/botspot-twelve-ways-to-spot-a-bot-aedc7d9c110c

PageFair. (2017). *2017 Adblock Report [Graph]*. Dublin. Retrieved from https://pagefair.com/blog/2017/adblockreport/

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, É. (2012). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. https://doi.org/10.1007/s13398-014-0173-7.2

Pennington, J., Socher, R., & Manning, C. D. (2014a). GloVe : Global Vectors for Word Representation. *Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543. Retrieved from https://nlp.stanford.edu/pubs/glove.pdf

Pennington, J., Socher, R., & Manning, C. D. (2014b). GloVe: Global Vectors for Word Representation [Dataset]. Retrieved 18 March 2018, from https://nlp.stanford.edu/projects/glove/

Perez, S. (2018). A Google R&D team wants to bring Smart Reply to all your chat apps. *TechCrunch*. Retrieved from https://techcrunch.com/2018/02/13/a-google-rd-team-wants-to-bring-smart-reply-to-all-your-chat-apps/?guccounter=1

Plummer, L. (2017). The best bots you can used on Facebook Messenger. Retrieved 16 June 2018, from http://www.wired.co.uk/article/chatbot-list-2017

Radev, D. (2016). *Lecture 3 — Funny Sentences - Natural Language Processing | University of Michigan [Video file]*. University of Michigan. Retrieved from https://www.youtube.com/watch?v=UeiUiCRchiU&index=3&list=PLLssT5z_DsK8BdawOVCCaTCO99Ya58ryR

Rehurek, R., & Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 45–50. https://doi.org/10.13140/2.1.2393.1847

Ribeiro, F. N., Araújo, M., Gonçalves, P., André Gonçalves, M., & Benevenuto, F. (2016). SentiBench - a benchmark comparison of state-of-the-practice sentiment analysis methods. *EPJ Data Science*, *5*(1). https://doi.org/10.1140/epjds/s13688-016-0085-1

Richter, F. (2017). Netflix Turns 20 [Graph]. Retrieved 11 October 2017, from https://www.statista.com/chart/3153/netflix-subscribers/

Roffo, G., Giorgetta, C., Ferrario, R., & Cristani, M. (2014). Just the way you chat: Linking personality, style and recognizability in chats. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8749*, 30–41. https://doi.org/10.1007/978-3-319-11839-0_3

Rollins, J. B. (2015). Foundational Methodology for Data Science, 6. Retrieved from https://www.slideshare.net/JohnBRollinsPhD/foundational-methodology-for-data-science

Rubenstein, H., & Goodenough, J. B. (1965). Contextual correlates of synonymy. *Communications of the ACM*, *8*(10), 627–633. https://doi.org/10.1145/365628.365657

Rudrapal, D., Jamatia, A., & Chakma, J. (2015). Sentence Boundary Detection for Social Media Text. *ICON*. Retrieved from http://cdn.iiit.ac.in/cdn/ltrc.iiit.ac.in/icon2015/icon2015_proceedings/PDF/13_rp.pdf

Russell, J. (2018). Google changes its messaging strategy again: Goodbye to Allo, double down on RCS. *TechCrunch*. Retrieved from https://techcrunch.com/2018/04/19/google-changes-its-messaging-strategy-again-goodbye-to-allo-double-down-on-rcs/

Santorini, B. (1995). Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision, 2nd printing). *University of Pennsylvania*. Retrieved from ftp://ftp.cis.upenn.edu/pub/treebank/doc/tagguide.ps.gz

Schwaber, K., & Sutherland, J. (2013). The Scrum Guide. *Scrum.Org and ScrumInc*, (July), 17. https://doi.org/10.1053/j.jrn.2009.08.012

Slimani, T. (2013). Description and Evaluation of Semantic Similarity Measures Approaches. *International Journal of Computer Applications*, *80*(10), 25–33. https://doi.org/10.5120/13897-1851

Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., … Dolan, B. (2015). A Neural Network Approach to Context-Sensitive Generation of Conversational Responses. In *Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies (NAACL-HLT 2015)*. Retrieved from https://www.microsoft.com/en-us/research/project/data-driven-conversation/?from=http%3A%2F%2Fresearch.microsoft.com%2Fen-us %2Fprojects%2Fconvo

spaCy. (2016a). Annotation Specifications. Retrieved 25 March 2018, from https://spacy.io/api/annotation#named-entities

spaCy. (2016b). Facts & Figures. Retrieved 25 March 2018, from https://spacy.io/ usage/facts-figures

Speer, R., Chin, J., & Havasi, C. (2017). ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. *Proceedings of 31St AAAI Conference on Artificial Intelligence*. Retrieved from https://arxiv.org/pdf/1612.03975.pdf

Speer, R., & Lowry-duda, J. (2017). ConceptNet at SemEval-2017 Task 2: Extending Word Embeddings with Multilingual Relational Knowledge. *Conference: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 85–89. Retrieved from https://arxiv.org/pdf/1704.03560.pdf

Stanford NLP Group. (n.d.). The Stanford Natural Language Inference (SNLI) Corpus. Retrieved 7 March 2018, from https://nlp.stanford.edu/projects/snli/

Statista. (2018). Number of paying Spotify subscribers worldwide from July 2010 to January 2018 (in millions) [Graph]. Retrieved 22 January 2018, from https://www.statista.com/statistics/244995/number-of-paying-spotify-subscribers/

Suchanek, F. M., Kasneci, G., & Weikum, G. (2007). YAGO: a core of semantic knowledge. *Proceedings of the 16th International Conference on World Wide Web*, 697–706. https://doi.org/10.1145/1242572.1242667

Tagg, C. (2009). *A CORPUS LINGUISTICS STUDY OF SMS TEXT MESSAGING*. University of Birmingham. Retrieved from http://etheses.bham.ac.uk/253/1/Tagg09PhD.pdf

Team Snapchat. (2018). Introducing Story Ads - a new way to showcase your brand or product in Discover! Retrieved 3 June 2018, from https://forbusiness.snapchat.com/blog/introducing-story-ads-a-new-way-to-showcase-your-brand-or-product-in-discover/

Twitter. (n.d.). Rate limits. Retrieved 21 November 2017, from https://developer.twitter.com/en/docs/basics/rate-limits

Twitter. (2017). Developer Policy. Retrieved 21 November 2017, from https://developer.twitter.com/en/developer-terms/policy.html

Twitter Sentiment Analysis. (n.d.). Retrieved 25 November 2017, from https://www.kaggle.com/c/twitter-sentiment-analysis2/data

Universal Dependencies. (2017). Universal POS tags. Retrieved 13 March 2018, from http://universaldependencies.org/u/pos/all.html

University of California. (2016). SMART Goals : A How to Guide. California: University of California. Retrieved from https://www.ucop.edu/local-human-resources/_files/performance-appraisal/How to write SMART Goals v2.pdf

University of Cambridge. (2017). SimLex-999. Retrieved 7 June 2018, from http://www.cl.cam.ac.uk/~fh295/simlex.html

Wikipedia Community. (2002a). AIML. Retrieved 16 June 2018, from https://en.wikipedia.org/wiki/AIML

Wikipedia Community. (2002b). Interactive voice response. Retrieved 17 June 2018, from https://en.wikipedia.org/wiki/Interactive_voice_response

Wikipedia Community. (2002c). Singlish. Retrieved 7 November 2017, from https://en.wikipedia.org/wiki/Singlish

Wikipedia Community. (2003). Levenshtein distance. Retrieved 19 February 2018, from https://en.wikipedia.org/wiki/Levenshtein_distance

Wikipedia Community. (2007a). Bag-of-words model. Retrieved 19 June 2018, from https://en.wikipedia.org/wiki/Bag-of-words_model#N-gram_model

Wikipedia Community. (2007b). Cosine similarity. Retrieved 12 October 2017, from https://en.wikipedia.org/wiki/Cosine_similarity

Zhang, X., Zhao, J., & Lecun, Y. (2015). Character-level Convolutional Networks for Text. *NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems*, *1*, 649–657. Retrieved from https://arxiv.org/pdf/1509.01626.pdf

Zhou, Z., Wang, Y., & Gu, J. (2008). New model of semantic similarity measuring in wordnet. *Proceedings of 2008 3rd International Conference on Intelligent System and Knowledge Engineering*, 256–261. https://doi.org/10.1109/ISKE.2008.4730937

Zhu, G., & Iglesias, C. A. (2015). Sematch: Semantic entity search from knowledge graph. *CEUR Workshop Proceedings*, *1556*.

Zhu, G., & Iglesias, C. A. (2017). Computing Semantic Similarity of Concepts in Knowledge Graphs. *IEEE Transactions on Knowledge and Data Engineering*, *29*(1), 72–85. Retrieved from https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7572993

Appendixes

Appendix A: Descriptive Work Plan

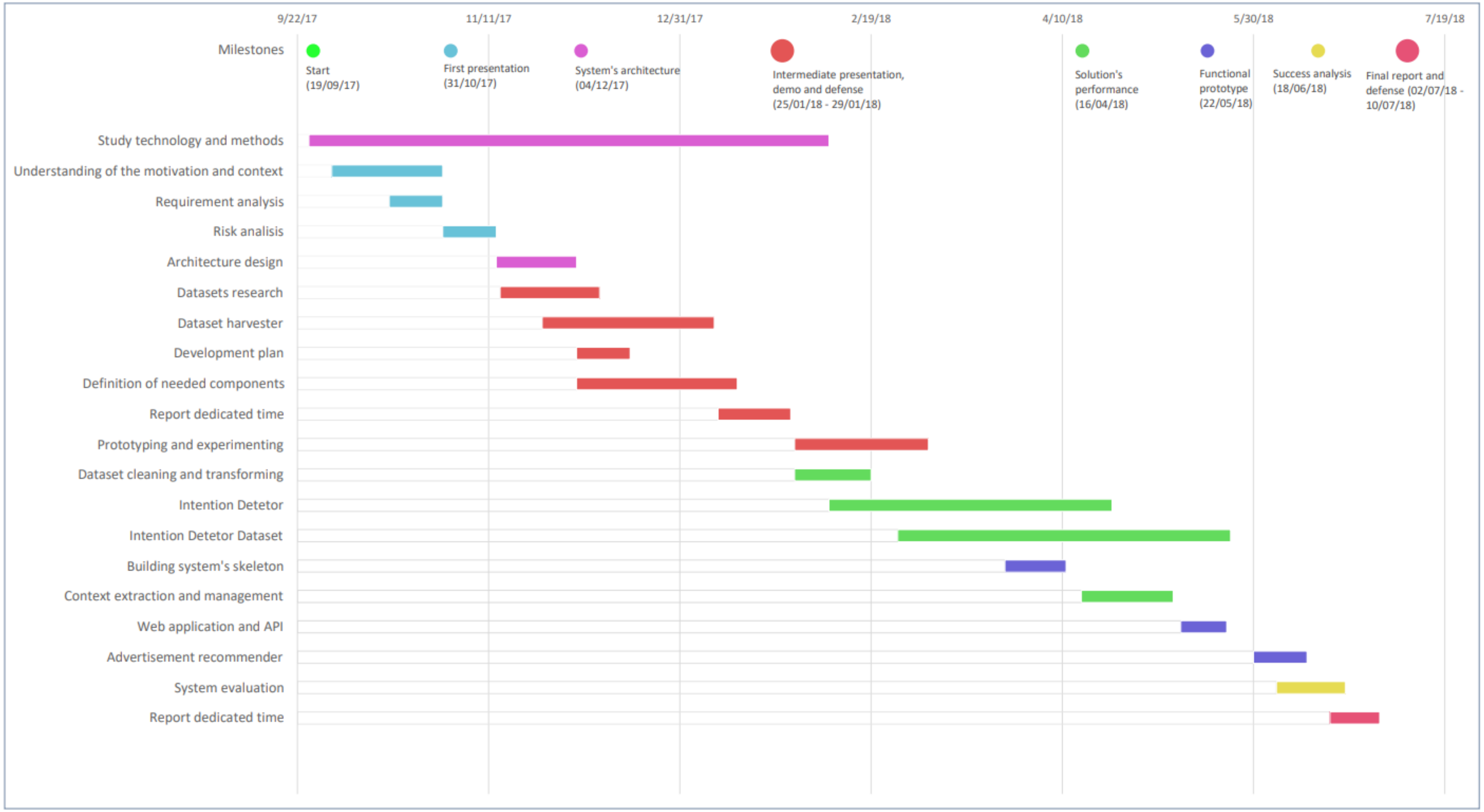


Figure 22: Descriptive work plan. The colours associate tasks to the milestones they are tightly associated with

Appendix B: The Intention Detection Experiments

The followings experiments were done to better understand the efficiency of different algorithms and features found during research.

The first of the experiments focus\ on finding the best Word/Phrase Embeddings, together with the ML algorithm that better performs on the *Intention Detection Dataset*.

The second, and last, experiment focuses on the assessment of the addition of text features as input for the training of the ML algorithms. Identifying if they bring discriminative power or noise.

Appendix B1: The phrase embedding experiment

The present experiment was designed to obtain information regarding the performance of different combinations of WPEAs and MLAs.

The evaluation consisted of the use of different WPEAs either trained by scratch with the *Reddit Corpus* or using their pre-trained available version. Once these were prepared, they would be used as translators for the MLAs, transforming text into vectors. The *Intention Detection Dataset* was then used to train and test the different MLAs, supported by the WPEAs. The testing step was done using the *Stratified Cross-Validation* with 10 folds, and the *F1-score* as the evaluating metric.

Since the text to be transformed consisted of sentences rather than words, the WE algorithms would use the average of the WEs in a phrase to calculate their PEs.

The variables of the experiment are shown in *Table 18*

Variable	Values	Library
WPEA	Trained and Pre-trained Word2Vec	(Rehurek & Sojka, 2010)
	Trained Doc2Vec	
	Trained and Pre-trained FastText	
	Trained and Pre-trained GloVe	(Pennington et al., 2014a)
	Pre-trained InferSent	(Conneau et al., 2017)
MLA	Gaussian Naive Bayes	(Pedregosa et al., 2012)
	Logistic Regression	
	Linear Support Vector Machine	
	Polynomial Support Vector Machine	

Table 18: Intention Detection Experiment 1: Variables

Regarding the training of the WPEAs, standard parameters were used. The pre-trained versions had already those parameters defined, however, for those trained from scratch the parameters were the following:

- Number of dimensions: 300
- Window size: 10
- Number of iterations over the dataset: 13
- Learning rate (*alpha*): 0.05

The missing parameters were left unchanged with their default values.

The training of the MLAs was done by optimising their hyper-parameters. Nevertheless, for simplicity, the optimal parameters were omitted. The results of the experiment can be seen in *Tables 19, 20, 21 and 22.*

On a general basis, the *Polynomial SVM* managed to outperform any other MLA regardless of the WPEA used. This superiority strongly indicates that the remaining MLAs can be discarded concerning the *Intent Detector Dataset*. Also, the pre-trained versions of the algorithms outperformed the trained versions. However, the trained versions had a fairly good performance and used fewer resources to be trained.

In respect to the question of the experiment, the *InferSent + Polynomial SVM* performed the better. Although the remaining WPEAs, with the exception of *Doc2vec*, were close behind by a distance of *2%* on average. Furthermore, the *Doc2vec* WPEA had a notorious underperformance, greater than *10%*, which indicates its inadequacy to approach this kind of problems when compared with other candidate solutions.

In conclusion, *InferSent + Polynomial SVM* was the best performing combination, and, therefore, selected as default models for the *Intention Detector*.

GloVe	MLA	Accuracy	Precision	Recall	F1 Score	F1 Score Std. deviation
Pre-trained with “Twitter” corpus” Available at: (Pennington, Socher, & Manning, 2014b)	Gaussian Naive Bayes	72.40%	70.24%	79.39%	74.38%	6.90%
	Linear SVM	77.30%	76.39%	79.37%	77.77%	5.00%
	Polynomial SVM	**82.50%**	**84.02%**	**80.56%**	**82.20%**	4.49%
	Logistic Regression	76.90%	76.44%	78.39%	77.34%	5.33%
Pre-trained with “Wikipedia 2014 + Gigaword 5” corpus. Available at: (Pennington et al., 2014b)	Gaussian Naive Bayes	68.30%	65.92%	77.61%	71.16%	4.35%
	Linear SVM	76.60%	75.74%	78.40%	76.98%	3.81%
	Polynomial SVM	**81.60%**	**79.22%**	**86.06%**	**82.43%**	**3.69%**
	Logistic Regression	75.25%	74.29%	77.42%	75.79%	4.78%
Trained with the *Reddit corpus*	Gaussian Naive Bayes	66.42%	64.65%	73.47%	68.62%	5.77%
	Linear SVM	73.68%	73.46%	74.85%	74.00%	3.78%
	Polynomial SVM	**80.27%**	**80.95%**	**79.96%**	**80.23%**	**3.32%**
	Logistic Regression	73.18%	73.18%	73.67%	73.33%	**3.56%**

Table 19: Intention Detection Experiment 1: GloVe results. The three best values for each metric are highlighted

Word2Vec	MLA	Accuracy	Precision	Recall	F1 Score	F1 Score Std. deviation
Pre-trained with “Google News” corpus. Available at: *(Google Code, 2013)*	Gaussian Naive Bayes	68.39%	65.39%	79.58%	71.69%	4.82%
	Linear SVM	77.12%	76.66%	78.40%	77.46%	3.93%
	Polynomial SVM	**83.40%**	**82.18%**	**85.47%**	**83.74%**	3.97%
	Logistic Regression	76.43%	76.12%	77.42%	76.68%	3.80%
Trained with the Reddit corpus	Gaussian Naive Bayes	71.83%	69.48%	78.80%	73.75%	6.70%
	Linear SVM	76.91%	75.76%	79.36%	77.50%	**3.20%**
	Polynomial SVM	**80.85%**	**79.51%**	**83.69%**	**81.44%**	**3.54%**
	Logistic Regression	**78.10%**	**77.27%**	**79.96%**	**78.53%**	**2.41%**

Table 20: Intention Detection Experiment 1: Word2vec results. The three best values for each metric are highlighted

FastText	MLA	Accuracy	Precision	Recall	F1 Score	F1 Score Std. deviation
Pre-trained with "Common Crawl" corpus. Available at: (Bojanowski, Grave, Joulin, & Mikolov, 2016a)	Gaussian Naive Bayes	68.39%	64.89%	**81.16%**	72.06%	6.46%
	Linear SVM	**78.79%**	**78.07%**	80.56%	**79.20%**	**2.95%**
	Polynomial SVM	**82.91%**	**80.55%**	**86.84%**	**83.54%**	**2.09%**
	Logistic Regression	**78.30%**	**77.85%**	79.77%	78.70%	3.98%
Trained with the Reddit corpus	Gaussian Naive Bayes	68.77%	66.40%	76.82%	71.08%	4.69%
	Linear SVM	77.00%	75.27%	80.53%	77.77%	3.89%
	Polynomial SVM	77.80%	74.62%	**84.47%**	**79.16%**	**3.31%**
	Logistic Regression	77.00%	75.73%	79.74%	77.62%	4.36%

Table 21: Intention Detection Experiment 1: FastText results. The three best values for each metric are highlighted

WPEA	**MLA**	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**F1 Score Std. deviation**
Doc2vec *Trained with the Reddit corpus*	Gaussian Naive Bayes	63.67%	65.94%	56.62%	60.64%	6.78%
	Linear SVM	70.91%	73.04%	66.40%	69.46%	**5.26%**
	Polynomial SVM	72.79%	76.26%	66.22%	70.74%	6.31%
	Logistic Regression	71.02%	73.11%	66.42%	69.36%	6.65%
InferSent *Trained with the pre-trained Glove vectors using the "Common Crawl 840B" corpus* *Available at: (Pennington et al., 2014b)*	Gaussian Naive Bayes	76.34%	74.69%	**80.17%**	77.24%	5.54%
	Linear SVM	**81.93%**	**80.45%**	**84.49%**	**82.34%**	**3.28%**
	Polynomial SVM	**85.08%**	**84.82%**	**85.85%**	**85.26%**	**3.87%**
	Logistic Regression	**77.52%**	**76.45%**	79.98%	**78.03%**	5.54%

Table 22: Intention Detection Experiment 1: Doc2vec and InferSent results. The three best values for each metric are highlighted

Appendix B2: The text features experiment

The present experiment was designed to obtain information regarding the performance of text features considering the *Intention Detection Dataset*.

This experiment consisted of the evaluation of the best performing MLA, found in the previous experiment, trained with different input features. Analogously to the previous experiment, the evaluation was done using the *Intention Detection Dataset* through the *Stratified Cross-Validation* with 10 folds, and the *F1-score* metric.

The text features extracted were empirically divided into four categories:

- Sentiment analysis (Vader (Hutto & Gilbert, 2014)
- Named entities (full list in (spaCy, 2016a))
- Part-of-speech tags (Universal POS tags (Universal Dependencies, 2017))
- Verb tags (extracted from the fine-grained POS tags (Santorini, 1995))

The combinations experimented and their results are shown in *Table 23.*

Surprisingly, the *Sentiment Analysis* feature that seemed to hold potential was not relevant at all. This can be seen in the performance of the MLA with *Phrase Embeddings* when compared to its performance with *Phrase Embeddings + Sentiment*.

Additionally, the *Part-of-speech* set of features, although it had the best performance by itself when compared to the other text features, it was inferior when merged with the *Phrase Embeddings*. This might be explained by an overlap of discriminative power which would indicate that having both sets of features is redundant and, ultimately, negative for the MLA.

Finally, the best text feature found was the *Verb tags* set. However, the improvement it adds to the lone *Phrase Embeddings* is of *0.36%*, which is rather marginal. A further division of features must be done to unveil and select the real valuable text features from this set, since, as shown in the performance of *All features*, having redundant or non-discriminative features can create noise and affect the overall performance of the MLA.

In conclusion, the *verb tags* text features were added to the *Phrase embeddings* to very slightly improve the best model's performance.

Input features	Accuracy	Precision	Recall	F1 Score	F1 Score Std. deviation
Phrase Embeddings	**85.08%**	**84.82%**	**85.85%**	**85.26%**	3.87%
All text features	54.41%	53.09%	76.61%	62.60%	4.68%
Sentiment	48.43%	46.44%	30.82%	34.58%	11.80%
Entities	52.26%	69.67%	8.25%	14.47%	5.85%
Part-of-speech	57.38%	55.60%	82.27%	65.84%	3.87%
Verb tags	58.15%	67.95%	30.44%	41.86%	7.58%
Phrase embeddings + Verb tags	**85.47%**	**85.36%**	**86.05%**	**85.62%**	**3.21%**
Phrase embeddings + Part-of-speech	84.09%	84.72%	83.88%	84.13%	4.19%
Phrase embeddings + Entities	84.88%	84.56%	85.66%	85.05%	**3.32%**
Phrase embeddings + Sentiment	**85.08%**	**84.82%**	**85.85%**	**85.26%**	3.87%
Phrase embeddings + Verb tags + Sentiment	**85.47%**	**85.36%**	**86.05%**	**85.62%**	**3.21%**
All features	83.70%	83.84%	84.27%	83.89%	4.54%

Table 23: Intention Detection Experiment 2: Input features results. The four best values for each metric are highlighted

Appendix C: The Similarity Measures Experiment

In the interest of selecting the best performing semantic similarity measure to be used on the *Advertisement Recommender*, series of tests had to be executed.

The experimental setup consisted of the evaluation of semantic similarity measures with different datasets designed for the assessment of the word similarity task. Using the evaluation extension of the *Sematch* library, it was possible to test datasets and to compare the measures' performances against human's.

For a given measure, *Sematch* calculates all similarities between words within the dataset, saving each result in a vector. Once the vector, whose size must be equal to the number of word pairs in the dataset, is filled, it is compared to another vector created from human's performance. This comparison is done using the *Spearman Correlation* metric, which allows the testing of a measure against human's performance. In conclusion, the more it correlates with human's performance, the better the measure.

The variables of the experiment are shown in *Table 24*.

Variable	Values	Reference
Dataset	SimLex-999 (666 word pairs, exclusively nouns)	(University of Cambridge, 2017)
	WordSimilarity-353 (353 word pairs)	(Evgeniy, 2002)
	Rubenstein & Goodenough, RG65 (65 word pairs)	(Rubenstein & Goodenough, 1965)
	Miller & Charles 1991 (30 word pairs)	(Miller & Charles, 1991)
Semantic measure	Shortest Path	(Slimani, 2013)
	Wu & Palmer	
	Li	
	Jiang & Conrath	
	Lin	(Lin, 1998)
	Zhou (k=0.5)	(Zhou et al., 2008)
	Wpath	(Zhu & Iglesias, 2017)
	Wpath-graph	(Zhu & Iglesias, 2017)
	Project’s best word embedding candidate	–
Ontology	WordNet	(Miller, 1995)
	YAGO	(Suchanek et al., 2007)

Table 24: Similarity measures experiment's variables

The results of the experiments are shown in *Figure 23*. It can be appreciated the performance of each pair ontology-measure against the four datasets. The last column indicates the final evaluation. It is a weighted average of the performances in each dataset. The weights were set considering the number of samples in each dataset, the more samples, the higher its weight. See *Table 25*.

Dataset	**Amount of samples**	**Weight**
SimLex-999	666	0.5978456014
WordSimilarity-353	353	0.3168761221
Rubenstein & Goodenough (RG-65)	65	0.05834829443
Miller & Charles 1991	30	0.02692998205
Total	**1114**	**1.0**

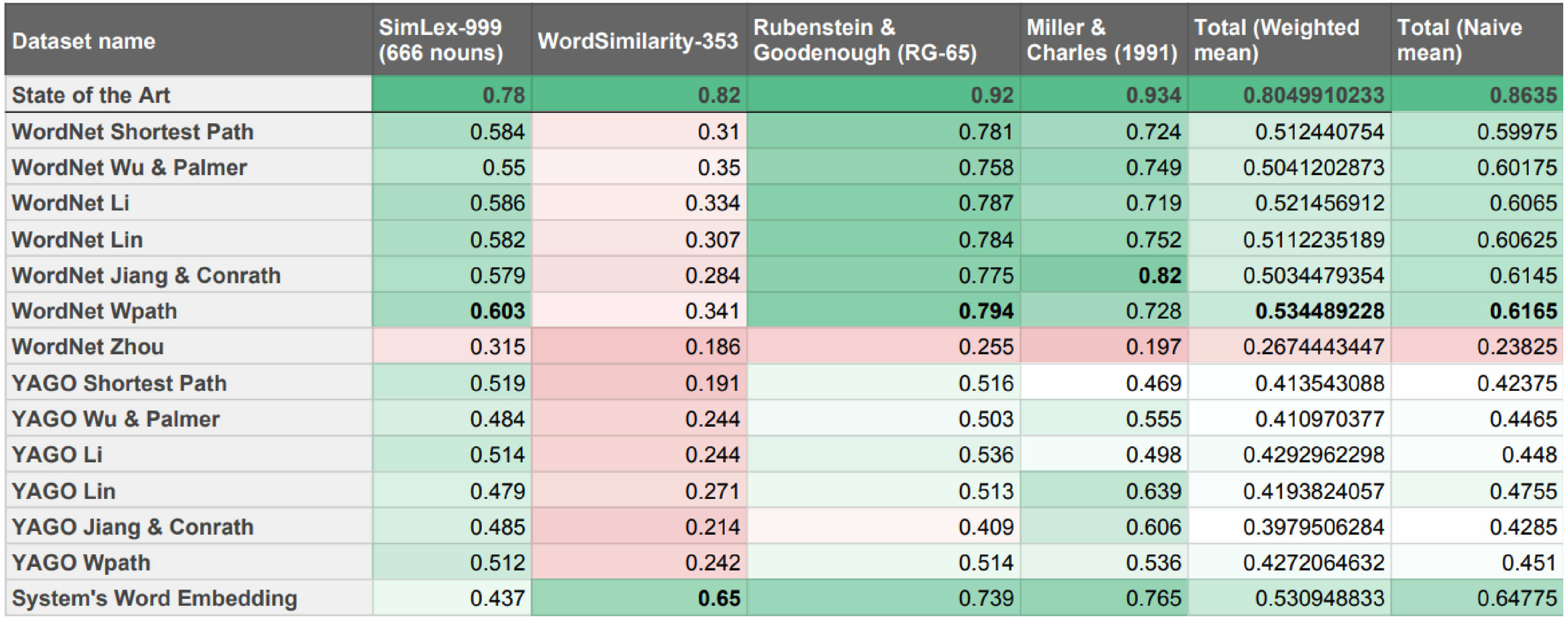
Table 25: Datasets weights for the similarity measures experiment

It is noticeable that the semantic similarity measure *Wpath* together with the *WordNet* ontology was the best combination in two out of the four datasets, also obtaining the highest result in the weighted evaluation closely followed by the *System's Word Embedding*.

Surprisingly, the *Zhou* measure underperformed heavily compared with the benchmarks seen in previous works (Zhou et al., 2008). The reason may be associated with the omitted optimization of the *k* parameter. Moreover, these results might highlight the importance of this optimization. However, more evidence is needed to be certain.

Understanding whether YAGO and WordNet were complementary rather than alternatives, was also an intended purpose of this experiment. No conclusion could be achieved regarding this matter, yet, the performances between those two ontologies had shown to be significantly different. Since no decision could be taken with higher certainty, both ontologies were included in the *Advertisements Recommender*.

Conclusively, both *WordNet* with *Wpath* and *YAGO* with *Li* were taken to be part of the implementation of the *Advertisements Recommender*.



Dataset name	SimLex-999 (666 nouns)	WordSimilarity-353	Rubenstein & Goodenough (RG-65)	Miller & Charles (1991)	Total (Weighted mean)	Total (Naive mean)
State of the Art	0.78	0.82	0.92	0.934	0.8049910233	0.8635
WordNet Shortest Path	0.584	0.31	0.781	0.724	0.512440754	0.59975
WordNet Wu & Palmer	0.55	0.35	0.758	0.749	0.5041202873	0.60175
WordNet Li	0.586	0.334	0.787	0.719	0.521456912	0.6065
WordNet Lin	0.582	0.307	0.784	0.752	0.5112235189	0.60625
WordNet Jiang & Conrath	0.579	0.284	0.775	**0.82**	0.5034479354	0.6145
WordNet Wpath	**0.603**	0.341	**0.794**	0.728	**0.534489228**	**0.6165**
WordNet Zhou	0.315	0.186	0.255	0.197	0.2674443447	0.23825
YAGO Shortest Path	0.519	0.191	0.516	0.469	0.413543088	0.42375
YAGO Wu & Palmer	0.484	0.244	0.503	0.555	0.410970377	0.4465
YAGO Li	0.514	0.244	0.536	0.498	0.4292962298	0.448
YAGO Lin	0.479	0.271	0.513	0.639	0.4193824057	0.4755
YAGO Jiang & Conrath	0.485	0.214	0.409	0.606	0.3979506284	0.4285
YAGO Wpath	0.512	0.242	0.514	0.536	0.4272064632	0.451
System's Word Embedding	0.437	**0.65**	0.739	0.765	0.530948833	0.64775

Figure 23: Similarity measures experiment's results. The greener the cells, the higher the correlation between the measure and human's performance, the values in bold indicate the highest correlation. On top, the State-of-the-Art performance is given as reference

Appendix D: Dataset Research

The appropriate data had to be found both for understanding better the problem at hand, and to serve as training for the AI algorithms. Due to these needs, finding an IM conversational dataset was mandatory.

During the research, very interesting datasets were found, but all of them had troublesome issues.

Singapore SMS Corpus (Chen & Kan, 2013), seemed to be a perfect choice. However, by examining its data it was noticed that the language spoken could not be considered as typical English. Several expressions couldn't even be interpreted from the author's perspective. Further investigation about this fact was done unveiling that no traditional English could be detected since the language was not English but *Singlish*, which is a variation of English influenced by other Asian languages and popular culture (Wikipedia Community, 2002c). In conclusion, the dataset was discarded.

The NPS Internet Chatrooms Conversations (Forsyth, Lin, & Martell, 2010) is a misleading dataset that at first glance seemed to have potential but it represents a whole different type of conversations, chatroom conversations. These communications are usually anonymous and tend to be full of discussions. No context or user's information is given whatsoever. Conclusively, this dataset was discarded.

The Santa Barbara Corpus (Bois et al., 2005), consists of 14 transcriptions of phone conversations. It has interesting conversations that could give insights about users' intentions. However, it cannot be directly used. Only 14 transcripts are recorded, the transcriptions show no characteristics that resemble IM conversations and it demands some cleaning due to interferences detected along the conversation. This dataset was discarded.

The CorTxt Dataset is a huge corpus of English language text messages used in this study (Tagg, 2009). It was the perfect candidate. Still, it was nowhere to be found and its only reference is the included study.

The Movie Dialogs Corpus (Danescu-Niculescu-Mizil & Lee, 2011), it has a huge amount of conversations transcribed from films. Yet, similarly to the *Santa Barbara* Corpus, its text consists purely of transcriptions of conversations which do not resemble text in IM conversations. This dataset was discarded.

The Reddit Corpus (Baumgartner, 2015), similarly to the *NPS* dataset, Reddit is a forum made for anonymous discussions. It is a very good source for informal writing communication, but, this anonymity inhibits the presence of user intentions. Phrases like "Let's go out tonight" are rarely found in anonymous environments.

After the failure in finding the perfect dataset, a second-best solution was sought instead, social media conversations. Posterior to some investigation it was clear that other researchers were mainly using Twitter's data when they needed to approach this informal writing style predominant in social media platforms. Consequently, the focus of the research shifted to Twitter datasets. Unfortunately, a different problem raised. Twitter's new policy about its own data, turning effective the 11th of November of 2017 (Twitter, 2017). This policy highly limited the sharing of Twitter datasets to third parties.

As a consequence, datasets like **SNAP** *(Leskovec & Krevl, 2014)* and **Twitter100k** (Hu et al., 2018), were available no more. A Twitter dataset was made available for a Kaggle's competition which purpose was, specifically, sentiment analysis ('Twitter Sentiment Analysis', n.d.). However, after a shallow look at the data it was clear that it did not have the heterogeneity necessary, the text found was not representative of the conversations in IM applications, it was specifically tailored for approaching the sentiment analysis problem.

Finally, the **Microsoft: Social Media Conversation Corpus** was found (Sordoni et al., 2015). It consists of several sets of three tweetIDs representing a three-step conversation through Twitter. The date of the tweets is around 2012, however, some tweets were missing and the Twitter API was required to do the hydration of the tweetIDs. Yet, this was the best candidate found.

The creation of a dataset wanted to be avoided since it would translate into a delay in the project. However, since the use of the Twitter's API was already required, a step further was done in order to harvest a specific data. In conclusion, the Twitter API was to be used to harvest tweets associated with the use cases of the project, resulting in a domain-specific dataset which size would grow over time.

Appendix E: The Working System Experiment

As established by one of the *thresholds of success* defined in *Section 2.2: Requirements and Thresholds of Success*, all project's use cases must be approached. Furthermore, one of the must-have requirement demands for the whole system to be tested.

In the interest of fulfilling these requisites, this experiment was designed. It consists of the monitoring of 15 different conversations by the *Advertisement Recommendation API*. The evaluation was based on the differences between the system's and human's recommendations.

The variables of this experiment are *the conversations* and *advertisements*. The conversations were manually manufactured, while the advertisements were collected through the Google Places API. Both were specifically selected to include information regarding the uses cases of the project.

The human recommendation is represented by the author's recommendations. In some cases, the recommendation made by the system might differ, but this does not necessarily mean it is a bad recommendation. Due to the existence of advertisements that overlap in the services/products they provide, more than one recommendation can be done effectively.

In *Table 26* and *Table 27* the *advertisements* and *conversations,* respectively, are presented.

Name	Location	Working hours	Promotion
Café Santa Cruz	Coimbra	Everyday from 8h to 0h	Live music
Murphy's Irish Pub	Coimbra	Closed on Mondays, opens from 18h to 4h the remaining days	Biggest beers in city
Pizza Hut Coimbra	Coimbra	Everyday from 12h to 23h	Buy 2 pizzas and get free delivery
NB Club Coimbra	Coimbra	Closed on Sundays, Mondays and Wednesdays, opens from 23h to 6h the remaining days	Girls night
Cinema NOS Forum Coimbra	Coimbra	Everyday from 13h to 3h	Popcorn 50% off
McDonald's Coimbra Fernão Magalhães	Coimbra	Everyday, at any time	Pay 1 McFlurry get 2
Brunn's Coffee Diner Coimbra	Coimbra	Everyday from 12h to 2	Girls pay half
Cinemas NOS	Lisboa	Sunday opens from 10h to 3h, opens from 12h to 3h the remaining days	2 tickets per 1

Table 26: The Working System Experiment: advertisement variable's values

Nº	Conversation
1	- Hi Bob, how are you? - I just got to Coimbra. Do you want to grab a coffee?
2	- Hey! - Do you know any good restaurant in Coimbra?
3	- Are you free tonight? - Let’s watch a movie together :)
4	- I will be back to Coimbra next week - Are you free then? - We can lunch together
5	- What are you doing tomorrow? - I’m in Coimbra and was thinking on going out at night, wanna tag along?
6	- Are you in Coimbra already? - Let’s go out tonight, drinks on me!
7	- Are you on Lisboa? - Let’s hang out tonight! - Are there any good movies airing?
8	- I will visit coimbra next weekend - We can have dinner if you are free ;)
9	- Hey! - Wanna go for a drink?
10	- Hey I’m in Coimbra! - I will call for some food before going out at night
11	- I loved living in coimbra… - Btw, do you wanna have dinner later?
12	- I am in coimbra - but I ain’t in the mood for movies tonight
13	- Yes, Im in coimbra - but I will have dinner home

Nº	Conversation
14	- did you hear? - he asked her out and she said no!
15	- I have to deliver my thesis next monday in coimbra - do you have any plans for that night?

Table 27: The Working System Experiment: conversation variable’s values

The results of the experiments are shown in *Table 28*

Conversation	System’s recommendation	Human’s recommendation	System’s performance
1	None	Café Santa Cruz	Negative
2	None	Brunn's Coffee Diner Coimbra	Negative
3	None	None, not enough context	Positive
4	Pizza Hut Coimbra	Pizza Hut Coimbra	Positive
5	Murphy's Irish Pub	NB Club Coimbra	Acceptable
6	Pizza Hut Coimbra	Murphy's Irish Pub	Negative
7	Cinemas NOS	Cinemas NOS	Positive
8	Pizza Hut Coimbra	Brunn's Coffee Diner Coimbra	Acceptable
9	None	None. Not enough context	Positive
10	Pizza Hut Coimbra	Pizza Hut Coimbra	Positive

11	None	None. Not enough context	Positive
12	Cinema NOS Forum Coimbra	None	Negative
13	None. No intention detected	None	Positive
14	None. No intention detected	None	Positive
15	Pizza Hut Coimbra	Brunn's Coffee Diner Coimbra	Acceptable

Table 28: The Working System Experiment: results

A total of **8 out of 15** cases were handled equally by the human and system. In the remaining cases, **3 out of 9** were good recommendations while the rest (4) were incorrect.

As it can be appreciated in the results of this experiment, much is yet to be done in the optimization of the recommendation system. It can be seen how the system successfully avoids recommending when no enough context is given. Nevertheless, it prevents itself from recommending when the context is missing but is not necessary as in *conversation 1* and *2*.

Additionally, more specific information about every advertisement is necessary to effectively discriminate between them. The constant recommendation of *“Pizza Hut Coimbra”* might be a result of this lack of specific information, ranking food establishments at a very similar level.

It is to be noticed how the context must be expressly received to be detected by the recommender. If no context is expressed, the system would not have enough information to recommend an advertisement.

Appendix F: The API Response Time Experiment

As established by one of the *thresholds of success* defined in *Section 2.2: Requirements and Thresholds of Success*, each individual request, regarding the advertisement recommendation, must be responded in **less than 1.5 seconds**.

With the intention of verifying whether this requisite is satisfied or not, this experiment was designed. It consists of the usage of all sentences in the *Intention Detection Dataset* to send 1018 sequential requests to the *Advertisement Recommendation API*. Finally, measuring the average response time.

After the execution of the experiment, the average response time obtained was: **0.268 seconds**. Since the result was inferior to **1.5 seconds**, it safe to conclude the threshold of success in question was achieved.

It is to be noticed that this result only represents the performance of the *Advertisement Recommendation API* in isolation. This cannot be extrapolated to a production environment since there are more variables to consider.

University of Coimbra Contextual Advertisement Engine Gabriel Angel Amarista Rodrigues