Escuela Politécnica Superior

Analysis, discovery and exploitation of Open Data for the creation of question-answering systems

Degree in Computer Engineering

Final Degree Project

Author:
Gonzalo Molina Gallego

Tutor/s:
Elena Lloret Pastor

Jose Norberto Mazón Lopez

Universitat d'Alacant
Universidad de Alicante

September 2018

Analysis, discovery and exploitation of Open Data for the creation of question-answering systems

A new approach for a Conversational Agent

Autor
Gonzalo Molina Gallego

Directores
Elena Lloret Pastor
Department of Languages and Computer Systems
Jose Norberto Mazon Lopez
Department of Languages and Computer Systems

Computing engineering

Escuela Politécnica Superior

ALICANTE, 7 de septiembre de 2018

Abstract

Text-based Dialogue Systems have become popular in recent years. More and more companies are using them because they can help in different processes of customer service. These conversational agents can be Goal Oriented (GO) systems or Open Domain systems.

The GO systems offer very specific information, while Open Domain systems can deal with and talk about multiple topics of conversation but without offering information in detail.

In this project, a hybrid architecture is proposed to define specific intents for oriented conversations and perform searches on data portals when the conversation is open domain. The proposed architecture combines Natural Language Processing (NLP) techniques and metadata extraction, in which the Conversational Agent (CA) can give detailed information as well as answer open domain questions exploiting open data.

Resumen

Los Sistemas de Diálogo Basado en Texto se han popularizado en los últimos años, cada vez son más empresas las que los utilizan porque pueden agilizar diferentes procesos de atención al cliente. Estos agentes conversacionales pueden ser dirigidos o de dominio abierto.

Los sistemas dirigidos ofrecen información muy específica, mientras que los de dominio abierto pueden dialogar sobre multiples temas de conversación pero sin ofrecer información en detalle.

En este proyecto, se propone una arquitectura híbrida con el fin de poder crear múltiples convresaciones orientadas y búsquedas en portales de datos cuando es de dominio abierto. La aqrquitectura propuesta une técnicas de Procesamiento de Lenguaje Natural y extracción de metadatos, en la que el agente conversacional puede dar información detallada además de responder preguntas de dominio abierto explotando datos abiertos.

Acknowledgements

Firstly, it was in my Erasmus scholarship when I experimented with NLP techniques, and since I discovered it, it was very clear to me that I wanted to project myself in that field. A few months later, in the subject Explotación de la Información, I knew my tutor, Elena Lloret, and she was a reference in this area.

This project would not have been possible without Elena and José Norberto Mazón, who has contributed with different approaches and great ideas. The project started with a small idea of a question-answer system, and together we have made it grow. I am very grateful to them, I have learn a lot with them.

I also want to thank my friends and family members who have accompanied me during these four years.

A mis abuelos y abuelas, y en especial a mi tío Antonio,
muchas gracias por todo

Those who can imagine anything,
can create the impossible

Alan Turing.

Index

Index of figures

Index of tables

Index of Lists

1 Introduction

In recent years, **Artificial Intelligence (AI)** has appeared to change the way of understanding the computers and their capabilities to help people in many and different situations, from solving difficult equations until recommending you a movie based on your interest [Krasadakis, 2018].

AI has many fields of investigation. This research will focus on Natural Language Processing (NLP) and Natural Language Generation (NLG), both terms related with the natural language of the humans and its understanding.

Question-answering (QA) is a computer science discipline within the fields of information retrieval and natural language processing (NLP), which is concerned with building systems that automatically answer questions posed by humans in a natural language. If this concept of answer question is extended to create a dialogue, a Conversational Agent (CA) emerges. The CA is capable of interacting with the user as well as answering his questions.

Within the field of AI, the contribution of this project is to create a CA capable of recognizing the conversation topic, search information about that topic and offer to the user the possibility of see that new information. This technique is based on a education model called Socratic Method (SM).

This project will take advantage of the advanced state of NLP techniques to be able to identify data sets within **Open Data** and offer it to a user in a simple way. Currently, the process of searching for information of a user is to enter a question in a search engine, make a request, and filter different websites trying to find the answer of its question. For example, if a user wants to know *How many libraries are in New York?*, he would go to Google to look for that information and explore within the obtained result to find it. Instead of that, using the CA proposed in this project, it would respond directly with the information previously analyzed and filtered.

The CA, popularly known as chatbot, will be capable of identifying the important words of a sentence, know the intent of the user and answer its doubts generating and obtaining information from Open Data Portals or other data sources.

Once the previous terms have been introduced, the next step is to answer the next questions, what are NLP and NLG? and what are their differences?

NLP: Area of AI with the main goal of how computers can be used to understand and manipulate human language. With NLP, developers use natural language text or speech to create tools and techniques to make computers interact with humans. It includes several fields of studies, such as machine translation, natural language text processing and summarization, user interfaces, speech recognition and more [Chowdhury, 2007].

NLG: Is a subfield of NLP, it generates natural language from a source of data, knowledge base or logical form. In order to perform this task, templates are the most common way to create sentences, but there exist different approaches using Neural Networks (NNs), as it can be seen in this article [Freitag and Roy, 2018].

Reading both meanings, its differences can be appreciated, while NLP focus on analyze the language to extract relevant data, NLG is used to combine the data extracted with contextualized narratives.

At this point, the tools to understand and generate language have been presented, but, if the chatbot does not have knowledge, it would be useless. To give the power of information to the chatbot, it will be feed of Open Data from many different datasets.

Data are commonly understood to be the raw material produced by abstracting the world into categories, measures and other representational forms. It can be extracted through observations, computations and experiments [Kitchin, 2014]. If all this data are published to everyone, the concept of Open Data appears, is the idea that some data should be freely available to everyone to use and republish as they wish, without restrictions [Aparicio García et al., 2016].

But, why Open Data? The availability of open data has grown significantly. Everyday there are more public organizations releasing their raw data. The motivations are that open access to publicly generated data provides greater returns from the public investment, users can analyze large quantity of datasets to solve complex problems [Janssen et al., 2012]. The chatbot proposed in this project will be able to connect to Open Data portal, request the information that user needs, analyze it and extract conclusions from these output data.

1.1. Motivation

Almost all people are connected to Internet through a mobile, computer or other device, either for buying a new par of shoes in Amazon [1] or watch a movie on Netflix [2]. Every move they do or operation they perform, is saved as data. All these data is stored, and with the idea of Open Data, publicly available to every person to use it, publish again or just analyze it. Governments join this practice creating several data portals

[1] https://www.amazon.es/
[2] https://www.netflix.com/

with many datasets, for example, the crime rate in New York or the government public finances are accessible on Data.Gov [3] to citizens, but is impossible for a human to read every single dataset in every platform.

The motivation of this project is to make more accessible the data published by different governments in its Open Data portals to people, just interacting with an agent, giving them the power of know everything related with their city or country in just a few seconds.

The agent will create a bridge between AI and Open Data. With NLP, the Dialogue System will understand what users want, and then it will search among a lot of Open Data datasets, trying to find the most accurate. Furthermore, the CA is based on a education concept called SM (explained on 5.4.1), this method consist on create a conversation only using new questions about the previous concepts. So the proposed project is a *Socratic Chatbot* and it will understand a user and it will identify concepts to give them to it, having rich dialogues.

1.2. Problem Setup

Nowadays, there are many possibilities of finding freely data, but it is hard to navigate through every dataset related with the desired information and extract conclusions from it. In this report an alternative way to do this task thanks to a CA is proposed.

The CA has been designed to answer questions that need a lot of information processing before providing the right output and then give more information related with the question of the user. If a user wants to find the answer without using the agent, the process is complicated and inefficient as illustrated in section 1. Using the intelligent agent, the user will ask the same question, but it will obtain the information without entering in many websites, also, the output information is given by a reliable data source. Once the user receives the answer, the CA will ask the user for more information related with its first question.

Figure 1.1 shows the different tasks that a user has to perform when he wants to search for information in a traditional Search Engine, where after receiving the result of the search, the user has to manually filter all the information to find the one he needs (left diagram of 1.1. On the other hand, it shows the task that the user does when interacting with the CA, the user only has to introduce what he wants to look for and the chatbot will perform the search and filtering actions. Also the chatbot will give to the user extra information about its first query.

The first challenge is how to extract important words of the sentence and recognize its meanings. After that, a dataset will be chosen between many others in a Open Data portal. Finally, a request will be sent to the dataset selected to receive the result of the

[3] https://www.data.gov/

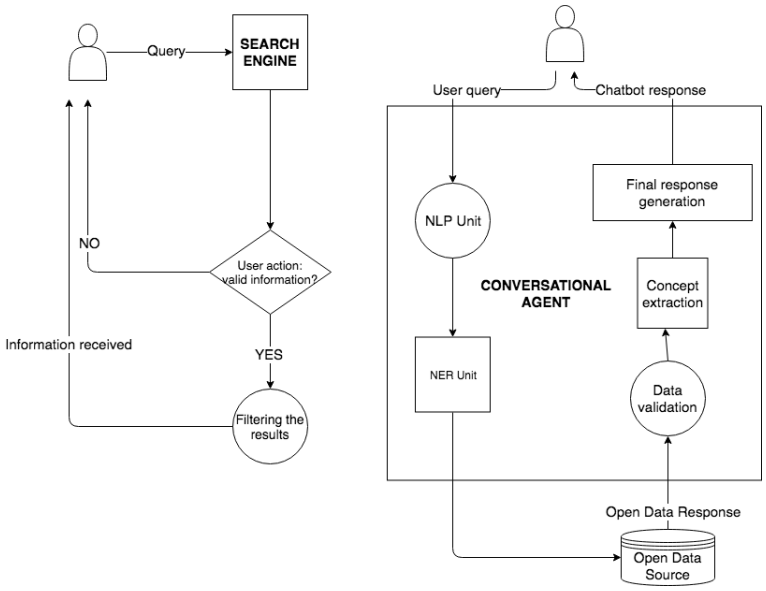

Figure 1.1: Traditional Search Engine vs. Conversational Agent.

query, and then, a small text will be generated with more information related to that dataset.

1.3. Objectives

This report aims to create an intelligent agent capable of interacting with humans and providing them new knowledge from public open data sources.

The detailed analysis of each objective will be discussed below, but here there is a summary of most important goals:

- Understand the user input: With a sentence of less than 120 characters, the chatbot will realize several operations on it to extract what user wants to know.
- Search and analyze Open Data: With the intent of the user input, it will select between thousands of datasets the correct one to request the data, then, it has to be analyzed to take the relevant information.
- Generate the response and give the user more information: Once the data is taked, a final output will be generated with the answer of user input and a new way to obtain more related data.

2 Theoretical Framework

In this chapter I will discuss the latest studies about NLP and NLG from different perspectives, understand and process the natural language on one hand and these techniques combined with Open Data on the other hand. Within these perspectives, I will analyze how can they be related with Open Data. Also, the chapter will explain the fundamentals of a chatbot and its State Of The Art (SOTA).

2.1. NLP

NLP is a section of AI, which takes care of understanding and process the human language. Inside this section there are two big blocks, the understanding of language and its generation. Nowadays, there are many applications using these techniques because it allows user to communicate with the computer in natural language [Diksha Khurana, 2017].

There are many subsections in NLP. As it is shown next2.1, five action areas appear: phonology that refers to sound, morphology to word formation, syntax to sentence structure, semantics syntax and pragmatics which refers to understanding.

Each area of study inside NLP corresponds to a level, and there are 7 different.

1. Phonology: Focus on the words in a sentence without taking care of the structure of the sentence
2. Morphology: The different parts of the word represent the smallest units of meaning known as Morphemes.
3. Lexical: Meaning of individual units, being a part-of-speech tag to each word
4. Syntactic: Grammatical structure of the sentence
5. Semantic: Determines the possible meanings of a sentence by pivoting on the interactions among word-level meanings in the sentence
6. Discourse: Properties of the text as a whole that convey meaning by making connections between component sentences
7. Pragmatic: The use of language in many different situations, with several meanings.

In this last decade, with the growth of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM), it has become popular the creation of machine translation

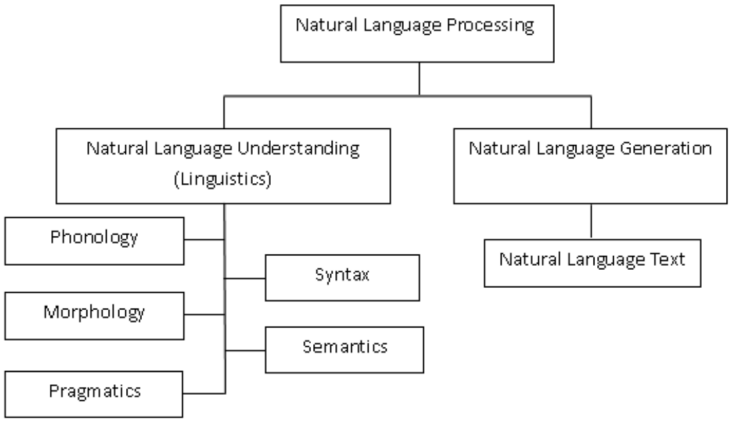

Figure 2.1: Classification of NLP [Diksha Khurana, 2017]

systems, for example, with Encoder-Decoder, capable of learning the translation probability of an English phrase to a corresponding French phrase [Kyunghyun Cho, 2014], also system for automatic summarization or Word2Vec models using Deep Learning (DP).

2.2. NLG

NLG is the process of producing phrases, sentences and paragraphs with a meaning and structure. It is a section of NLP, opposite with the understanding of a text. To generate natural language, it is needed relevant data to transform into a representation of human language.

Figure 2.2 shows the components of generating natural language:

1. Speaker and Generator: The generated text needs to be included in a intent to be related with the content of the situation.

2. Components and Levels of Representation: Going deeply into the generation of language, there exists many tasks being part of the generation core.

   a) Content selection.

   b) Textual Organization.

   c) Linguistic Resources.

   d) Realization.

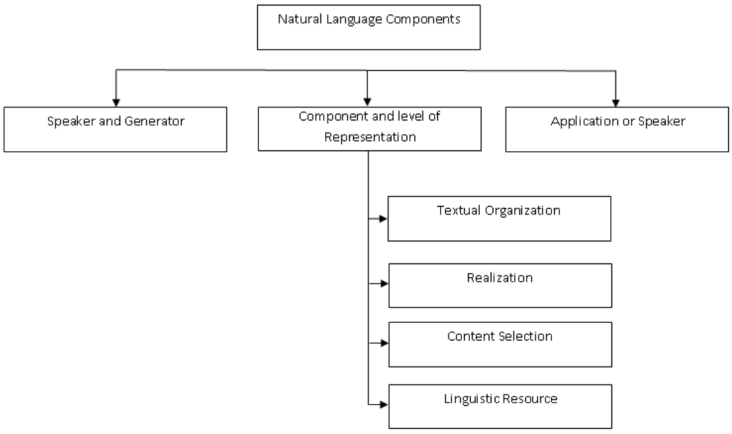

Figure 2.2: Components of NLG [Diksha Khurana, 2017]

3. Application: initiates the process of conversation but does not participate in the generation of text, it saves relevant information about the content and the topic of the conversation [Diksha Khurana, 2017].

The applications of NLG have changed with the growth of Generative Adversarial Networks (GAN), that helps humans in a range of task from posting a comment until write a poem. GAN models solves the problem of the low variety of output sentences generated by an application [Heng Wang, 2017].

2.3. Dialogue Systems

Chatbots have been become more popular from the last two decades, actually, many companies use these systems to have feedback about their products or services or launch a new publicity campaign. To better understand the potential of a chatbot, it is needed to know their origin.

A Dialogue System or CA aims to create comprehensive systems that can hold a real conversation, understanding the topics, giving reasonable answers, reasoning power, sentiment detection etc. Depending on the type of input-output of the system, there exist two types of CA [Ilievski, 2018]:

- Spoken Dialogue Systems (SDS): Models are designed for environments without chat interfaces and keyboards, the only way to communicate with the system is through a microphone and speakers. These systems are composed in several modules depending on its complexity.

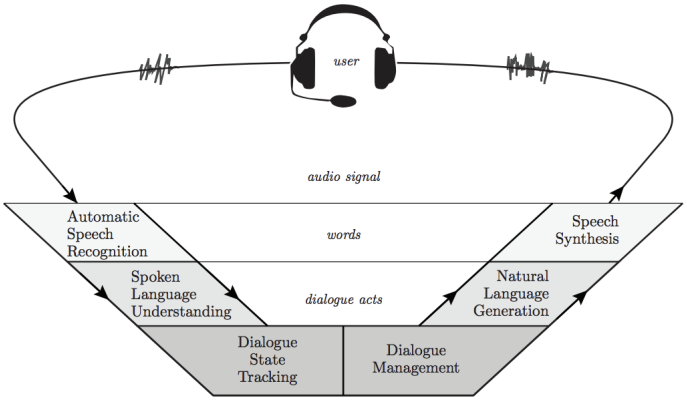

Figure 2.3: Modules of SDS from [Henderson, 2015]

In figure 2.3, the different modules of a SDS are shown. At the beginning of the process, it is necessary to convert from speech-to-text to predict the words of the sentence. Once the input audio is converted to text, the CA performs the functions of understanding and processing the natural language, and finally, generate a output speech from a generated text.

One of the first systems created of this type was JUPITER [Zue et al., 2000], a telephone-based conversational interface for weather information. The behavior of this systems follows the 2.3 schema. First, the Dialogue System understands the user input, and then the Dialogue System performs a SQL query to obtain the weather data and generate a response.

- Text-Based Dialogue Systems (Chatbots): As [Ilievski, 2018] explains, Text-Based Dialogue Systems are based on a chat interface, the user will use it to interact with the application. Depending on the conversation, there are 3 types of Text-Based Dialogue Systems:

  - Closed-domain or Goal Oriented (GO): The system has prepared previous knowledge about many intents that will be covered in the conversation and the chatbot will be trained to understand these topics. When the chatbot does not identify the concept of the conversation, it returns a predefined sentence.

  - Open-Domain: Here the system does not have trained topics, it has to understand and generate responses from a live searched dataset.

    [Higashinaka et al., 2014] shows an alternative architecture to understand the Open-Domain conversation and generate responses from many sources.

Figure 2.4: Logo of *data.gov*

2.4. Open Data

Open Data is the concept of the freely use of information, available to everyone to use and republish as they wish, without restrictions from copyright, patents or other mechanisms of control. In order to access to this data, most of the times you need to navigate through open data portals trying to find the appropriate set of data.

This section will give an overview of how Governments are using Open Data to create new strategies and initiatives.

Actually, many Governments have been publishing their data to make it available to public, everybody can access these data on-line. Government agencies are responsible of publishing and uploading the public datasets, then, citizens or businesses can download it or re-use it to create new innovation products.

The strategies of every Government related with the availability of the data seems similar, they publish all datasets in a centralized web portal. From this portal, citizens can access without any log in or restriction and check the datasets and use it. For example, one of the most famous portals is *data.gov* (this is its logo 2.4), which includes the open data published in the Unites States of America.

Actually, a user can search along more than 250000 datasets and uses several applications that contain these datasets as its knowledge base. Also, in the website there are many ways to access the data as a developer and create a simple application.

According to [Chan, 2013], the Governments have two main motivations making data available:

- Ethos of democracy: One of the features of the democracy is the freedom of information. Governments make its data public to become more open and transparent as opposed to being hide behind a veil of secrecy.

- Economical: Many governments are pursuing Open Data policies to drive economic growth and business innovation. By making these data for everyone, small businesses are able to utilize the data to create value-added products and services either for commercial gain or as a public good. Also, Governments publish their data to

have citizen engagement, because they become more transparency and they offer more public services related with these data as [Stott, 2014].

3 Objectives

This chapter will analyze the different challenges and goals of the project. The objectives are divided in three groups (three main goals) depending on the related solution of a problem, not with it nature (in many objectives it will need techniques of NLP and Open Data joined together). In each group, there will be many objectives of functions, and if these objectives are achieved, the CA will success.

Before starting with the core of the chapter, first it is needed an explanation of the three groups mentioned previously.

The first main goal is to answer the question of *How to build a chatbot?*, in that section there will appear goals related with the architecture of the system, the breadth of conversation that it will cover or the nature of answer.

The second group of objectives will be related with the data request to Open data, from the identification of the dataset until the creation of the response.

Finally, the last group is about the knowledge that the system predicts that will interest to user, creating questions with more information of a topic of the dialogue.

3.1. Designing a functional Dialogue System

Without a system capable of understanding the needs of a user, the rest of the project would be useless, so the base is to create a GO Dialogue System that identifies the intent of the conversation and interacts with the user. To achieve this goal, two sub-objectives arise.

- Intent detection: As it was previously mentioned, the architecture of the CA will be a hybrid between GO Dialogue Systems and Open Domain Systems, so it has to identify when it needs to scrape information from the Open Data portal or not. When it does not have to query the Open Data portal, there will be many trained intents and the CA will have answers related with every topic declared. In this table 3.1 there is an example.

- Answer generation: Each intent will have a set of predefined answers, but when the system accesses to the Open Data portal, it has to generate a response through a template related with the type of the question.

User input	Intent detected	Bot response
Good afternoon!	Welcome-CORPUS	Hello, how can I help you?
How are you?	How-are-you-CORPUS	I'm fine! Try to ask something.
How many libraries are in New York?	OPEN-DATA-REQUEST	TEMPLATE-RESPONSE
Who is Elon Musk?	WIKI-DATA-REQUEST	TEMPLATE-RESPONSE

Table 3.1: Intent detection example.

3.2. Providing knowledge to the CA

- Dataset selection: With the most important terms of the user sentence when it makes a question and the CA queries the Open Data portal, before that request, the system selects a specific dataset.
- Query the selected dataset and response: Once the system has selected the target dataset, it will obtain the necessary data making a API request to the Open Data portal.

3.3. Giving knowledge to the user

- Topic recognition: The CA will be able to identify the topic in which the user is interest, and it will give him the option of obtain more data related with the conversation.
- Search for information: Once it has the user interest, it will search in the web and make a summary of the information it has found.

4 Technology

This chapter briefly mentions and explains the technologies used to create the CA, from the language that has been used, to the most complex tools. The main technologies are divided into two groups depending on their purpose, one of these groups is NLP and the other the extraction of Open Data.

4.1. General

In this part a general overviw of the technologies used to develop the project will be given as the programming language chosen to do it and the Integrated Development Environment (IDE) on which it has been implemented.

4.1.1. Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for being used as a scripting or glue language to connect existing components together. Python is simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

4.1.2. Spyder IDE

Scientific PYthon Development EnviRonment is a a powerful interactive development environment for the Python language with advanced editing, interactive testing, debugging and introspection features and a numerical computing environment.

4.2. NLP

This section focuses on the libraries used for the human language processing.

4.2.1. Gensim

Gensim (see its logo on figure 4.1) is a free Python library designed to automatically extract semantic topics from documents, as efficiently (computer-wise) and painlessly

Figure 4.1: Gensim logo.

(human-wise) as possible. It is designed to process raw, unstructured digital texts.

The algorithms in Gensim, such as Word2Vec, FastText, Latent Semantic Analysis or Latent Dirichlet Allocation, automatically discover the semantic structure of documents by examining statistical co-occurrence patterns within a corpus of training documents. These algorithms are unsupervised, which means no human input is necessary – you only need a corpus of plain text documents.

Once these statistical patterns are found, any plain text documents can be succinctly expressed in the new, semantic representation and queried for topical similarity against other documents.

4.2.2. Natural Language Toolkit (NLTK)

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, or semantic reasoning.

4.3. Querying data

Finally, the technologies used to perform the operations related to the extraction of data from the United States Open Data portal are named and explained.

4.3.1. Comprehensive Knowledge Archive Network (CKAN)

CKAN (logo on figure 4.2) is a fully-featured, mature, open source data management solution. CKAN provides a streamlined way to make your data discoverable and presentable. Each dataset is given its own page with a rich collection of metadata, making it a valuable and easily searchable resource. CKAN allows you to easily publish and find datasets, store and manage your data, engage with the community, and customize and extend the features because it is all open source.

Figure 4.2: CKAN logo.

4.3.2. PostMan

Postman is a great tool when trying to dissect RESTful APIs made by others or test ones you have made yourself. It offers a sleek user interface with which to make HTML requests, without the hassle of writing a bunch of code just to test an API's functionality.

With this tool, the test of data requests is made to the open data portal of the United States. In this way we can see what url to call and how it returns the data.

5 Methodology

This chapter will focus on the different levels of implementation of the CA, starting at the base level with the explanation of the detection of the intents of the conversation, then, the query of data on external platforms, and finally, the enrichment of knowledge of the Dialogue-System and the generation of the final answer.

5.1. System Overview

The architecture of the system is divided into three levels or phases. Each level perform a different task. Without the task of the first level, the last one can not be performed. As shown in figure 5.1, the basis of the entire system is the detector of the topic of the conversation. The Intent Detection System will be in charge of managing the flow of the dialogue and launches the actions related to the topics of data search.

At the next level, searches are performed on the open data portal or on another website that can provide the information that is required. Here also an analysis of the phrase will be made in search of the necessary parameters to carry out this query

In the last level, related questions to the conversation topics will be created, offering the user more information about what he is talking about.

In order to unify these three levels and see the work-flow, figure 5.2 shows the internal communication of all the components of this architecture

In the diagram of figure 5.2 there are six columns, each related to a component of the architecture. The CA will start the conversation with a default message welcoming the user, and when he responds, the intent detector system (column 2) returns the topic of the conversation. If the intent has an associated action, the action handler will activate the process of accessing other platforms for data extraction, on the other hand, if it does not have any action, it will generate a random text among a set of responses (column 3).

When the action handler launches a data search process it can be of two types:

- Wikipedia Action: A concept or an entity will be searched in Wikipedia and the information will be returned in two different ways, or a link to the article in which the information was found, or the first paragraph of it.
- Open Data Portal Action: The actions related to the search in the Open Data portal will be operations of type *COUNT*, in which the user will request the amount of a

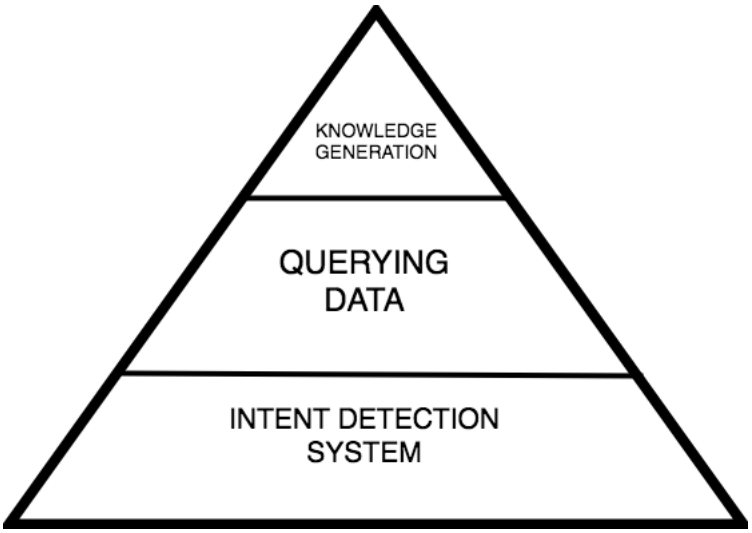

Figure 5.1: Phases of the System.

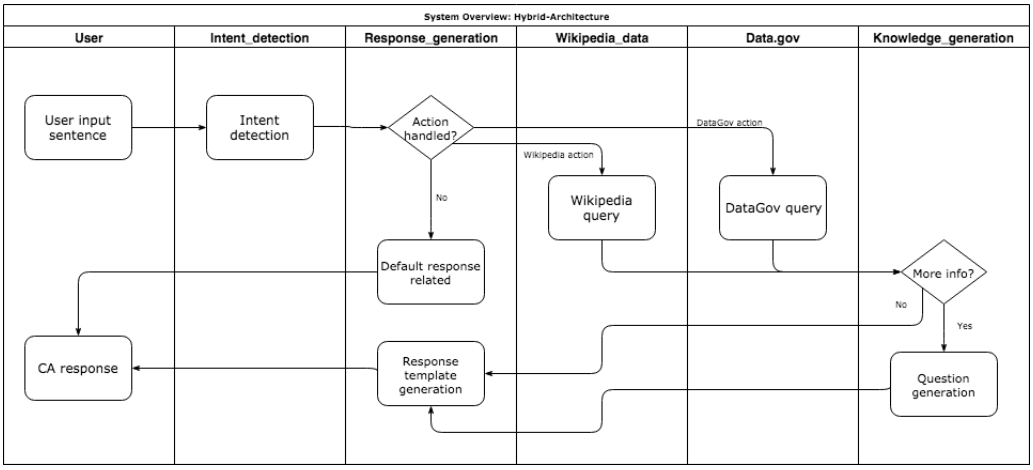

Figure 5.2: System Work-Flow.

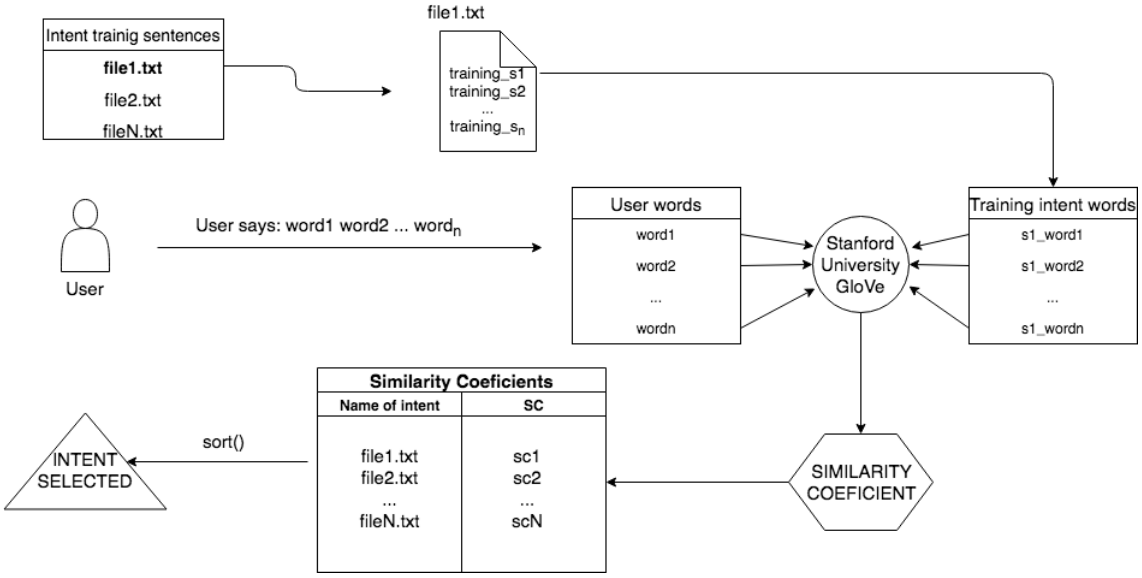

Figure 5.3: Intent detection architecture.

certain entity and will be searched in the portal. If the search does not return any results, the agent will respond with a default answer.

When an action has been launched, the detection of relevant information of the conversation is also activated. Depending on the type of action, some parameters or others are analyzed to give more data to the user. Once the system has all the enriched information available, it will be offered to the user. At the end of the process, if an action has been invoked, a template will be used to generate the response and show it to the user.

Unifying the scheme of figure 5.1 and the diagram of figure 5.2, it can be seen that column 2 correspond to the basic level (Intent Detection), 4 and 5 to intermediate (Querying data) and 6 to the last (Knowledge Generation).

5.2. Intent Detection Architecture

In this section the most important phase of the whole project is explained. It is the base of the CA and if it does not work well it would be impossible to be successful in the project.

The whole process of selecting an intent is outlined in the figure 5.3. Next, each process will be explained in detail.

This Dialogue-System will be composed of a set of predefined intents. An intent will correspond to a specific conversation topic, therefore, each one will have a set of valid answers associated. To create an intent, it is necessary to have some training phrases, which will compose the internal structure of that conversation topic. It is very important how the training phrases are defined and to what intent they are related, if several sentences are confusing, the agent will fail to recognize what the user is talking about.

The CA will have several classes of intent, in this case, there will be three different types:

- Corpus intents: Corpus-type intents are those related to conversations without the need for data search, that is, they have no associated action. When the system identifies an intent of this type, the agent returns a response at the moment, without the need to access Wikipedia or the Open Data portal, neither does the information enrichment process to be offered to the user. These intents are very important because they give the agent a personality.
- Wikipedia intents: When the user asks for a place or a public personality, the output of the intents detection system will be associated with the search in Wikipedia. From this point, a series of processes are activated to identify what information the user needs. Depending on the intent that activates this action, a different template will be used.
- Open Data intents: The search intents of Open Data are those that have to be formed by a well-defined structure, they are the most complex. From the user's phrase and the intents of this class, the system will have to perform a parameterized search in the data portal, choose an appropriate set and generate the associated response.

5.2.1. Creating and defining intents

The first step of the intents detection process is the creation and definition of training phrases. In a specific folder, new documents are added corresponding to each conversation topic, and within those files, a series of sentences are defined. The training phrases are possible interactions of the user with the chatbot, several sentences have to be defined by intent in order to increase the accuracy of the system. In the table 5.1 there are some examples of how define intents and training sentences.

5.2.2. Sentence transformation

Once the intents have been created and defined, the user will be able to interact with the CA. Here is when the process of NLP appears. Before analyzing the similarity of the user input phrase with the intents sentences, it is necessary to make several transformations to the training phrases and the user's input phrase.

Intent: help-intent-CORPUS
Sentence 1: I need help
Sentence 2: I need your help
Sentence 3: Help
Sentence 4: How can you help me?
Sentence 5: You can help me
Sentence 6: I want your help
Sentence 7: Help me please

Table 5.1: Training sentences for help-intent-CORPUS.

Specifically, the first transformation is related to a lexical analysis of the phrase, in which a tokenization process is applied. Once the system obtains the sentence divided into lexical units, it will be filtered by consulting a set of stop-words to eliminate unnecessary information.

This transformation process is applied to the user's input phrase and to each training phrase of each intent. To find the similarity between them, it compares the input phrase with each training phrase of the intent. After applying this, two vectors are obtained, one with the words relevant to the user's sentence and the other with the words of the training phrase of the intent.

The table 5.2 shows an example of how this process works. There are three columns in the table: the first is the start phrase, with punctuation marks and with all the words, the second column is the sentence after being tokenized, and the third is after the elimination of the stop-words.

In the first row, the user's phrase appears, the term *wu* corresponds to the user's word. The second is the intent training phrase and the term *wi* is word intent. The first column has both phrases complete because no process has yet been applied, in the second column each token of the sentences appears in a vector, and in the third column eliminates the unnecessary tokens. In this example, the user's phrase has five words (from wu1 to wu5), but after being filtered, it has only three words (wu2, wu3, wu5), so the terms wu1 and wu4 would be stop-words.

5.2.3. Global Vectors for Word Representation (GloVe)

To find how similar two terms are and obtain a ratio of similarity between two sentences, we use a GloVe model that provides system this information. Before explaining the application of this model to the project, first, GloVe will be explained.

	Complete sentence	After being tokenized	After being filtered
User	wu1 wu2, wu3 wu4 wu5!	['wu1','wu2','wu3','wu4','wu5','!']	['wu2', 'wu3','wu5']
Intent	wi1, wi2 wi3 wi4?	['wi1','wi2','wi3','wi4','?']	['wi1','wi2','wi4']

Table 5.2: Example of transformation process.

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

The statistics of word occurrences in a corpus is the primary source of information available to all unsupervised methods for learning word representations, but GloVe captures these statistics directly by the model ([Pennington et al., 2014]).

This technique has many uses, one of them is compute the similarity between two words. To do this task, the model places each word in a space, and to compute the similarity between two terms, the distance of both is calculated in that space.

Once the transformation process has been carried out in the user's input phrase and the training sentences of each intent, it is necessary to calculate the similarity coefficient between the user's sentence and each training sentence.

Within each intent there are different training phrases, once the coefficient of similarity of the sentence of the user has been calculated with each training phrase, only the highest coefficient is stored, which will be called *representative coefficient* of that intent.

As shown in figure 5.3, when performing these calculations on the set of all intents, a vector with the representative coefficients is obtained, and each position of this vector is directly related to the initial vector of intents. When the highest representative coefficient is ordered and returned, the corresponding intent will be chosen.

To achieve higher accuracy, an algorithm that relates the length of both sentences with the result of entering the terms in the GloVe model has been designed. Formula 5.1 shows how the coefficient of similarity is calculated, relating the length of both sentences to get a higher quality solution.

$$SC = \frac{\sum_{i=0}^{len(U)} max(model(U, I)))}{len(U)} \tag{5.1}$$

When this calculation has been made on all the set of intents, the intent of the conversation will be obtained and a response will be given, depending on whether it has action or not. This is the most important part of the project because it is the basis of the CA functioning.

Parameter	Meaning
U	Vector of filtered words of the user's sentence
I	Vector of filtered words of the intent training sentence

Table 5.3: Formula 5.1 parameters.

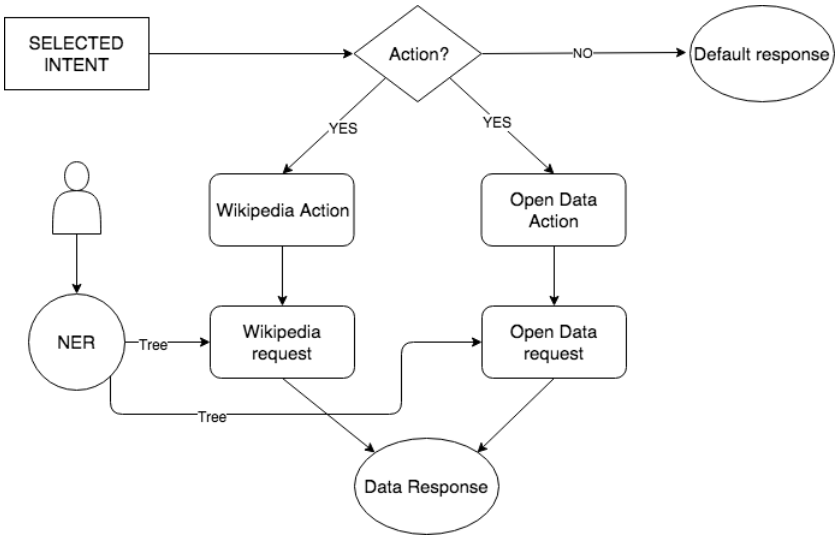

Figure 5.4: Data request architecture.

When the Intent Detection System returns the most appropriate conversation topic, it is checked if it has an associated action. If it does not have any action, it returns a predefined answer. If the intent has action, it goes to the second phase, explained next 5.3.

5.3. Data Query

If the selected intent has an associated action, the action handler is triggered and the second phase of the project begins. The second phase of the CA is related to the search for information that does not appear in the corpus. To obtain this data, the agent will use Wikipedia and Data.Gov to get them. The figure 5.5 shows an overview of the data request flow.

5.3.1. Identifying topics

In order to to consult the Open Data or Wikipedia platform, one or more parameters are needed to make the request. These parameters will be extracted from the input sentence of the user, who will ask for information that the CA does not have stored in the general corpus and has to search it in another data source.

To perform this task, it is needed a Named-Entity Recognition (NER). NER is a basic component of Information Extraction (IE) that aims to locate and classify named entities in text into pre-defined categories. There are different techniques for designing a NER as [Kanya and Ravi, 2012] shows:

- Supervised Learning: Tagging test corpus words when they are annotated as entities in the training corpus.
- Semi-Supervised Learning: It consists of using a small number of examples and the system tries to find other contextual similarities.
- Unsupervised Learning: The typical approach in unsupervised learning is clustering. The system tries to creates groups based on context similarity. There are also other unsupervised methods focusing on lexical resources, lexical patterns and statistics computed on a large unannotated corpus.

Once the relevant terms of the phrase have been extracted, they will be used to make the request to Wikipedia or the Open Data portal.

5.3.2. Open Data Portal

Querying Open Data portal is a difficult task because each dataset has a different structure. In this case, the CA uses the portal of the United States. This portal is based on CKAN, so the system is able to perform this Open Data search in every portal based on this API.

When the parameters of the query are identified, the most relevant data sets are searched by making a request to the CKAN API. This request will return an array of JSON documents with all the metadata of the datasets, for example the description, the name, the different formats, etc.

There are different restrictions of the portals that use CKAN. Through its API, you can only access the metadata of the portal, but not to a specific dataset. To solve this problem, a study of the response JSON model of the portal has been carried out in order to access the data. A variable was found in the response JSON that gave information about the formats that each dataset has (HTML, CSV, JSON, etc.). If one of the available formats is JSON, the system can access the data of that set through a request to an external API with the specific information of that dataset.

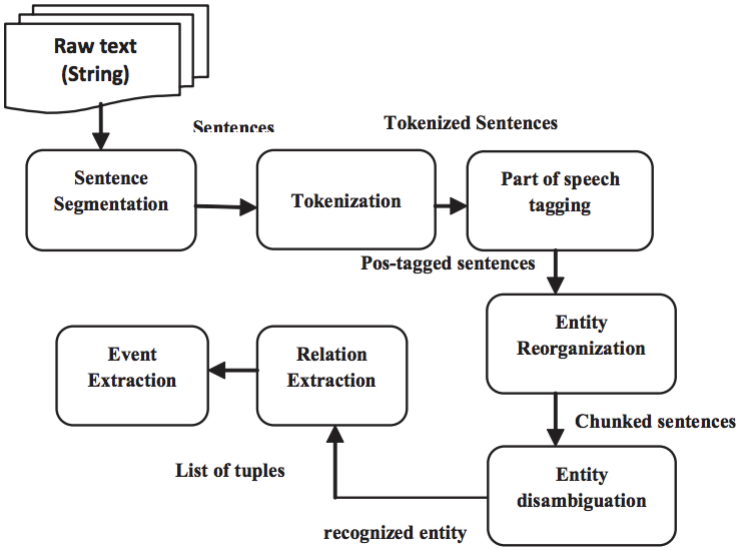

Figure 5.5: IE architecture (extracted from [Kanya and Ravi, 2012]).

These datasets models are not easily or properly mapped, so to simplify their representation, the operations will be divided in three groups:

- **Distributive functions**: which can be defined by structural recursion, for example, an input collection that can be partitioned into subcollections that can be individually aggregated and combined [Cabot et al., 2010].
- **Algebraic functions**: Finite algebraic expressions over distributive functions (average is computed using count and sum).
- **Holistic functions**: Functions that are not distributive nor algebraic.

Due to the high complexity of this process, the operations on these datasets have been restricted to Algebraic operations, , the system will return a quantity associated with the number of fields that the data have.

5.3.3. Wikipedia Data

To make requests to Wikipedia portal, the system uses a Python library to access the data. When the NER system generates the tree with the information, the extracted entities are searched in the portal.

The Wikipedia articles are of a large size, if the chatbot answered with all the content, it would be tedious for the user to read all the information. Therefore to improve

the user experience with the CA, this returns a summary of the content (this summary is just some sentences of the whole article) in one phrase and the link of the article so the user can visit it.

5.4. Getting Knowledge

Traditionally, most CA follow a question-and-answer model. The user sends a message and the chatbot responds. Although nowadays the chatbots offer more types of services to customers thanks to AI (product recommendation, advice, etc.), they follow the conventional model mentioned above.

One of the main objectives of this report is to create a Dialogue System that breaks the traditional models and proposes to the user the possibility of more information without the user requesting it. This model is influenced by the Socratic Conversation Methods.

This section will focus on how the chatbot search new information related with the conversation topic. Also, a review of what is the Socratic Method for the education will be explained.

5.4.1. The Socratic Method (SM)

The SM was created by Socrates, he was a classical Greek philosopher credited as one of the founders of Western philosophy, and as being the first moral philosopher of the Western ethical tradition of thought.

SM is composed mainly by two components, the teacher and the student. There is a dialogue between student and teacher. The role of a teacher is to ask the questions and a student role is to organize their past experiences and their knowledge in answering the questions. This method not only involves an interactive dialogue between teacher and students, it is also inductive. The teacher continuously leads the students to reason incorrectly then uses the counterexample to clarify the problem as [Bećirović, 2016] says.

The proposed Text-Based Dialogue System is based on this educational model. Even though the chatbot does not ask questions, its purpose is to educate the user by offering more data. The CA would be the teacher and the user who uses it, the student. The user begins the conversation with a sentence and the chatbot will show him information related to its question, if the user accepts that the system continues giving information, the dialogue tree evolves as more concepts appear.

5.4.2. Latent Semantic Analysis (LSA)

Once the system has found the information that the user requested, it is necessary to find the concepts related to the new data. When these terms are identified, they will be

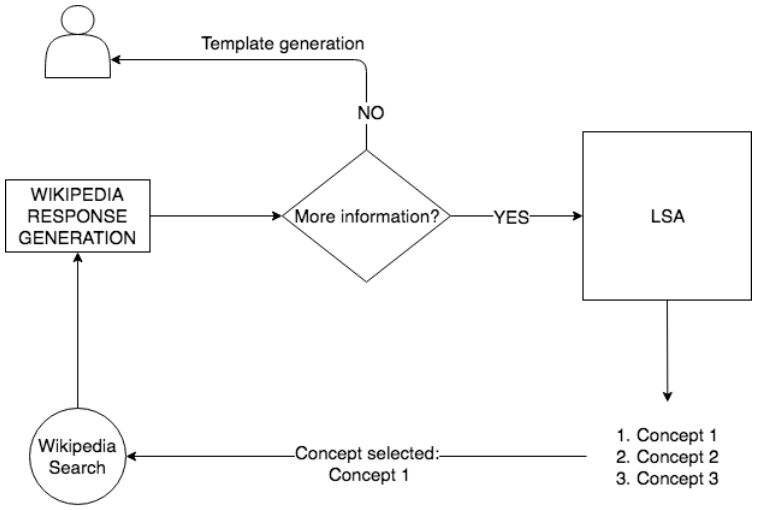

Figure 5.6: LSA architecture.

searched in Wikipedia and the result will be offered to the user. Data sets extracted from Data.Gov are labeled by subject, so the concepts are already identified, the problem is when the action launched is a request to Wikipedia. Figure 5.6 shows the architecture of the LSA in the system.

The system will have to extract the concepts from a short text, to solve this LSA is used. LSA is a technique for creating a vector representation of a document. Having a vector representation of a document gives you a way to compare documents for their similarity by calculating the distance between the vectors.

After executing this algorithm, a list with the most important concepts of the text is obtained , the most valued will be the one chosen to perform the following data query.

5.4.3. Response generation

The last phase of the conversational process of the chatbot is to find a way to generate responses that are not repetitive for the user. When the chosen intent is from the CORPUS, the answers are predefined and are chosen randomly.

On the other hand, if the intent has an associated action, the response of the chatbot must show the information of the searched data. Then, the CA asks the user if he wants to go deeper into the topic of conversation, if the user accepts, the chatbot collects more data and show them to him.

Using the same method to define default answers, several templates will be created and the extra content will be automatically added to the final answer. Each type of action will have many examples of templates to improve the user-experience.

6 Implementation

This chapter explains how the most important parts of the project have been implemented. To make the process clearer, code fragments were added together with explanations. Here, it will see how the GloVe model is declared and initialized, how the Intent Detection System has been implemented based on the formula 5.1, how calls are made to the Open Data portal and how the LSA is performed

6.1. GloVe

In order to use the GloVe model provided by Stanford University, downloading a file that contains the necessary data is needed. Then, we import the Word2Vec model to measure the similarity between two words. List 6.1 shows that the 100-dimensional version is used.

List 6.1: Definition and initialization of GloVe model

```
1 glove_input_file = 'glove.6B.100d.txt'
2 word2vec_output_file = 'glove.6B.100d.txt.word2vec'
3 glove2word2vec(glove_input_file, word2vec_output_file)
4 filename = 'glove.6B.100d.txt.word2vec'
5 model = KeyedVectors.load_word2vec_format(filename, binary=False)
```

To calculate the similarity of two terms, the model internally computes the distance between these terms and return a similarity value. 6.2 is an example of use, comparing the words *boy* and *man*.

List 6.2: Example of GloVe model

```
1 example = model.similarity('boy', 'man')
```

6.2. Intent Detection System

The algorithm that gives a weight to each training phrase of an intent being compared with the input of the user is the most important process in the system. In the code 6.3, the function calculate_average() receives the tokenized sentence of the user and the set of training phrases. The algorithm is in charge of looking for the maximum values of similarity between the user sentence and the set of training phrases.

List 6.3: Implementation of the Intent Detection System algorithm

```
def calculate_average(user_in, bot_in):
    max_coeficient = 0  #total similarity
    avg_coeficient = 0  #similarity between two sentences
    sentence_rep = 0    #higher similarity between the intent a the input
    for bs in bot_in:
        tokenized_bot = tokenizer(bs)  #tokenized bot sentence
        for wu in user_in:
            representative_ub = 0
            for wb in tokenized_bot:
                similarity_btw_words = model.similarity(wb.lower(), wu.lower())
                if similarity_btw_words > representative_ub:
                    representative_ub = similarity_btw_words
            max_coeficient = max_coeficient + representative_ub
        avg_coeficient = float(max_coeficient/len(user_in))
        if avg_coeficient > sentence_rep:
            sentence_rep = avg_coeficient
        max_coeficient = 0
        avg_coeficient = 0
    return sentence_rep
```

6.3. Data search: Open Data Portal

The process of searching, filtering and analyzing open data is very complex. The impossibility of accessing the data through CKAN, makes necessary to look for the external API to the Open Data Portal and make a second request to be able to create a response.

The method open_data_req() in 6.4 search and filter the Open Data Portal response, and if there are an external API, searches the data making a second request and return the information for the user and the tags of that dataset to offer more knowledge.

List 6.4: Open Data Search Request

```
def open_data_req(entity):
    r = requests.get('https://catalog.data.gov/api/3/action/package_search?q=
        entity&sort=score+desc')
    your_json = r.json()
    data = your_json['result']
    datasets = data['results']
    stop = 0
    url = ''
    quantity = 0
    tags = []
    for set in datasets:
        resources = set['resources']
        for r in resources:
            if 'JSON' in r['format']:
                stop = 1
                url = r['url']
                quantity = external_api(url)
                break
        if stop == 1:
            for tag in set['tags']:
                tags.append(tag['display_name'])
            break
    return [quantity, tags]
```

6.4. LSA

The LSA algorithm finds the most relevant terms within a text. This will be used to create the Socratic conversation, so its proper functioning is really important.

List 6.5: LSA implementation

```
def lsa_function(data):
    docs = [data]        # Data = text
    stopset = set(stopwords.words('english'))
    vectorizer = TfidfVectorizer(max_features = 10000, stop_words=stopset, use_idf
        =True, ngram_range=(1, 4))    # Convert a collection of raw documents
    # to a matrix of TF-IDF features.
    X = vectorizer.fit_transform(docs)  # Normalize data
    # MATRIX = U * SIGMA * Vt
    # Truncated SVD take Vt
    lsa = TruncatedSVD(n_components = 100, n_iter = 100)
    lsa.fit(X)
    terms = vectorizer.get_feature_names()
    selected = ''
    for i, comp in enumerate(lsa.components_):
        termsInComp = zip (terms, comp)
        sortedTerms = sorted(termsInComp, key = lambda x: x[1], reverse = True)
            [:10]
        selected = [sortedTerms[0], sortedTerms[1], sortedTerms[2]]
    return selected
```

The code 6.5 shows how this algorithm was made. Certain parameters have been modified to do it correctly on a document only. Now, lets see some important components:

List 6.6: TfidfVectorizer declaration

```

vectorizer = TfidfVectorizer(max_features = 10000, stop_words=stopset, use_idf=
    True, ngram_range=(1, 4))
X = vectorizer.fit_transform(docs)       # Normalize data
```

TfidfVectorizer 6.6 converts a collection of raw documents to a matrix of Term Frequency–Inverse Document Frequency (TF-IDF) features. TF-IDF is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in searches of information retrieval.

7 Experimentation

This chapter will focus on the experimentation and testing of our final project. The results obtained will be analyzed and commented in order to obtain the final conclusions. The intents detection system and the LSA extraction algorithm will be tested.

The Intent Detection System will be tested from four different approaches:

- Total topics recognized: The tests in this section will focus on testing the performance of the CA with different numbers of conversation topics. Basic packages of 5 conversation topics will be created and will be added to a set of user inputs. One of the most common problems of a chatbot is the training of the user, so it will be assumed that the user has already spoken with the Dialogue System.
- Higher verbosity vs. Lower verbosity: The packages of intents previously mentioned, will be tested with low verbosity and high verbosity sentences. Verbosity is the complexity of the sentence, assuming that a sentence with low verbosity has a maximum of 3 words, and one with high, of 4 words and up.
- Performance filtering words: To see the advantages of the NLP, test will be carried out without filtering the sentences of the user and those trained, afterwards, with the same data, a tokenization and a stop-words filter will be made to see if there is any change.
- Changing the dimensions of GloVe: GloVe is the representation of each word on a plane. With the Stanford University model, there are many variations created, one with 60, other with 100, 200 and 300 dimensions. It will test if the dimensions change the accuracy of the system.

The LSA tests will be comparing 10 random terms of Wikipedia by changing the length of the input text to see if the terms obtained with less text as input are similar to a larger amount of text.

7.1. Intent Detection System test

Before starting to test the Intent Detection System, several tables will be created with the different conversation topics and their training phrases. The declaration of training sentences have been based on [Lopez, 2018]. They will be defined in 3 different stages in order to compare the success by adding another set of intents.

There is no difference in complexity between the intents of each stage, it is a totally a random distribution designed to test the CA's ability to identify them.

The first row of each table of the stages is the name of the intent, so each column will correspond to a set of its training sentence. In the implementation, the intents are TXT files, those includes the sentence to train the system.

welcome-CORPUS	farewell-CORPUS	help-CORPUS	chatbot-CORPUS	how-are-you-CORPUS
greetings	bye	help	conversational agent	how are you?
hi	goodbye	I need help	robot	what about you?
welcome	bye bye	I need you	dialogue system	are you okay?
hello		I need something	chat bot	what's up?
hey				how are you doing?

Table 7.1: Intents and training sentences - Stage 1.

joke-CORPUS	music-CORPUS	movies-CORPUS	love-CORPUS	insult-CORPUS
joke	favorite band	movies	I love you	Idiot
some joke	do you like music	i love movies	Do you love me	Motherfucker
tell a joke	listen to music	favorite movie	Love	Jackass
	favorite song	film	Girlfriend	Jerk
	song		Boyfriend	fuck you

Table 7.2: Intents and training sentences - Stage 2.

turn-off-CORPUS	football-CORPUS	where-are-you-CORPUS	hobbies-CORPUS	creator-CORPUS
exit	football	Where do you live	my favorite hobbies	who creates you
turn off	I like football	Your city	I like to swim	you are a human
		Where are you	play football is fun	human
			I like to cook	creator
			read books	

Table 7.3: Intents and training sentences - Stage 3.

7.1.1. Filtering testing

The first test that will be carried out will be to observe if applying a tokenization and a stop-words filtering improves the results. This test will be perform using 100 dimensions of the GloVe.

ID	USER INPUT	EXPECTED INTENT	INTENT OUTPUT	SCORE
s1_1	Hello!	welcome-CORPUS	welcome-CORPUS	1.00
s1_2	Hi! My name is James	welcome-CORPUS	how-are-you-CORPUS	0.57
s1_3	Hey my friend	welcome-CORPUS	help-CORPUS	0.68
s1_4	See you later	farewell-CORPUS	how-are-you-CORPUS	0.81
s1_5	Bye bye!	farewell-CORPUS	farewell-CORPUS	1.00
s1_6	Have a good day	farewell-CORPUS	how-are-you-CORPUS	0.73
s1_7	I have a doubt	help-CORPUS	help-CORPUS	0.78
s1_8	Help me please	help-CORPUS	help-CORPUS	0.81
s1_9	I need your help	help-CORPUS	help-CORPUS	0.92
s1_10	Can you help me?	help-CORPUS	help-CORPUS	0.90
s1_11	Are you a chat bot?	chatbot-CORPUS	how-are-you-CORPUS	0.63
s1_12	You are a robot	chatbot-CORPUS	how-are-you-CORPUS	0.73
s1_13	What is going on?	how-are-you-CORPUS	how-are-you-CORPUS	0.82
s1_14	How are you?	how-are-you-CORPUS	how-are-you-CORPUS	1.00
s1_15	Everything okay?	how-are-you-CORPUS	how-are-you-CORPUS	0.91

Table 7.4: Results of the Intent Detection System without tokenization and stop-word filter process.

The Table 7.4 shows the results of experimenting with the user input set in the CA without performing the tokenization and the stop-words filtering.

There are 15 test sentences related to 5 different intent classes. When introducing the input phrases, the system has only hit **8 intents**, it has had a success of **53.3 %**. If we look the Figure 7.1 and 7.2, the most obtained output has *been how-are-you-CORPUS*. To understand why the system has tended to give that class as an exit, the verbosity of the training phrases of that intent and the user's sentences (s1_2, s1_4, s1_6, s1_11, s1_12) must be analyzed. These phrases have the higher verbosity, and although they have not important words, the system gives the same importance to all because the sentences have not been filtered, so if the user's phrase has a high verbosity, there is a high probability the output would be *been how-are-you-CORPUS*.

Now, input user's sentences and training sentences will be tokenized and filtered by a stop-words list. The purpose of that process is delete non-important words to avoid noise in our data.

The Figure 7.5 contains the results of the experimentation with the processed sentences. The critical sentences (s1_2, s1_4, s1_6, s1_11 and s1_12) have obtained a new score, so before compare the new output distribution, the new score will be analyzed in 7.3. The sentences s1_11 and s1_12 are interesting, in this experiment, both input have generated the correct output, the score of s1_11 has changed from 0.63 to 0.75 but s1_12 score has decrease. Thats why the distribution of the relevance of some declared intents

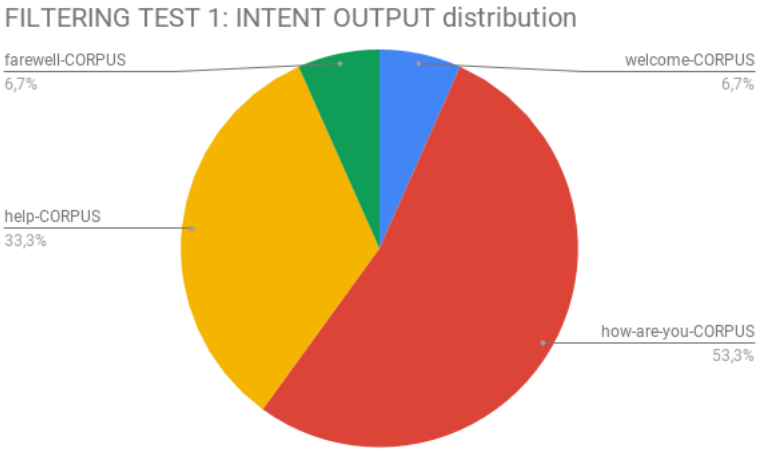

Figure 7.1: Filtering Experiments: Output distribution.

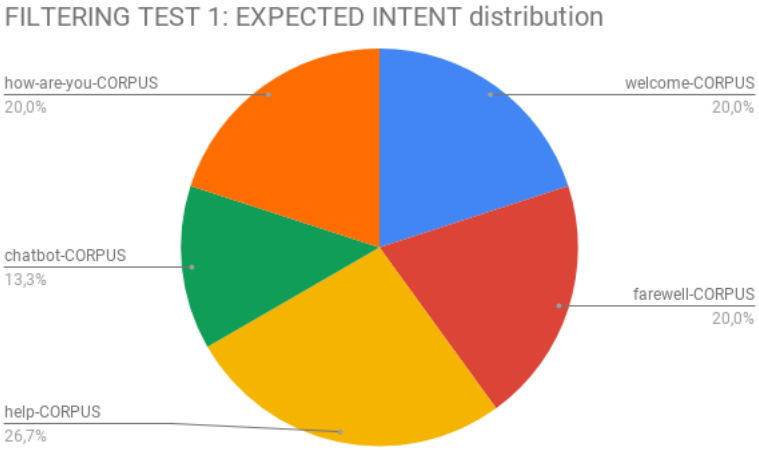

Figure 7.2: Filtering Experiments: Expected distribution.

ID	USER INPUT	EXPECTED INTENT	INTENT OUTPUT	SCORE
s1_1	Hello!	welcome-CORPUS	welcome-CORPUS	1.00
s1_2	Hi! My name is James	welcome-CORPUS	help-CORPUS	0.51
s1_3	Hey my friend	welcome-CORPUS	welcome-CORPUS	0.71
s1_4	See you later	farewell-CORPUS	how-are-you-CORPUS	0.71
s1_5	Bye bye!	farewell-CORPUS	farewell-CORPUS	1.00
s1_6	Have a good day	farewell-CORPUS	help-CORPUS	0.73
s1_7	I have a doubt	help-CORPUS	help-CORPUS	0.85
s1_8	Help me please	help-CORPUS	help-CORPUS	0.78
s1_9	I need your help	help-CORPUS	help-CORPUS	1.00
s1_10	Can you help me?	help-CORPUS	help-CORPUS	0.91
s1_11	Are you a chat bot?	chatbot-CORPUS	chatbot-CORPUS	0.75
s1_12	You are a robot	chatbot-CORPUS	chatbot-CORPUS	0.63
s1_13	What is going on?	how-are-you-CORPUS	how-are-you-CORPUS	0.89
s1_14	How are you	how-are-you-CORPUS	how-are-you-CORPUS	1.00
s1_15	Everything okay?	how-are-you-CORPUS	how-are-you-CORPUS	0.84

Table 7.5: Results of the Intent Detection System **with** tokenization and stop-word filter processes.

have been modified after filtered their training sentences, now these sentences have fewer words, so the output of the algorithm of matching intents is different.

With the same number of inputs, the system hits **12 intents**, with a new accuracy of **80 %**.

Figure 7.4 shows the new distribution of the accuracy by intent, more similar to 7.2 than the 7.1 in the first experiment.

In conclusion, this experiment shows that by performing the processes of tokenization and stop-words, irrelevant information is removed from the sentences and better results are obtained because the words that are compared are the most important of the sentence.

7.1.2. Adding more Intents

From now on, the experiments will be performed with the processes mentioned above. Now, experiments with the amount of intents that could be introduced in the model and that the results are still optimal are needed. With the previous experiments, the results of the stage 1 are already available, so Stage 2 will be added first, the same tests will be done to see the scores obtained and the distributions of the intents, and finally, the process will be repeated with the Stage 3 intents.

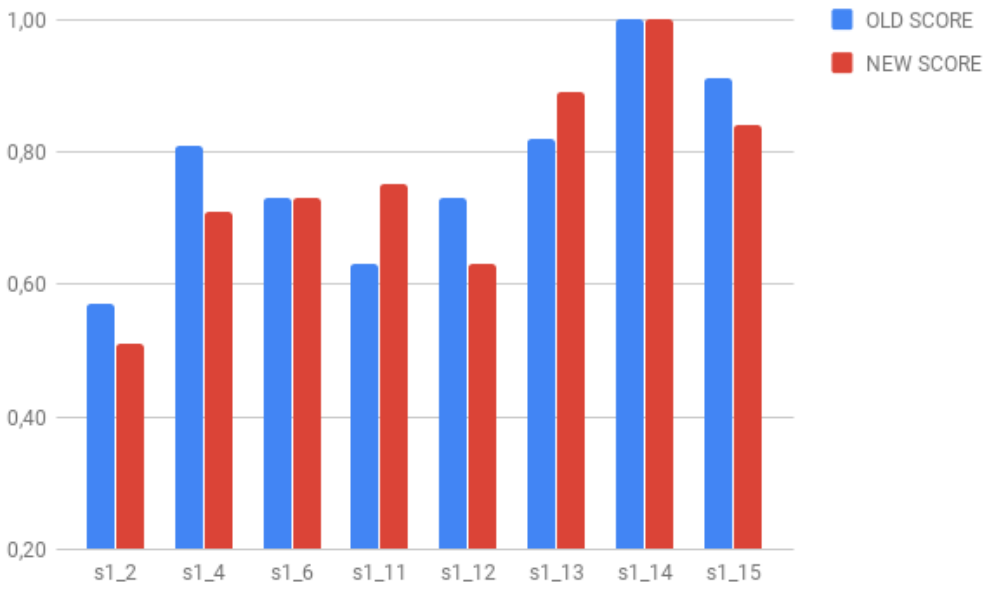

Figure 7.3: Final Score comparison.

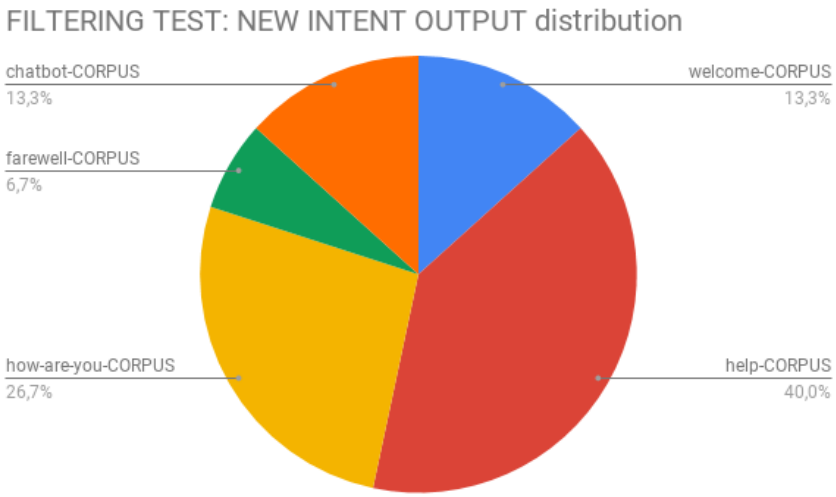

Figure 7.4: New Output Distribution.

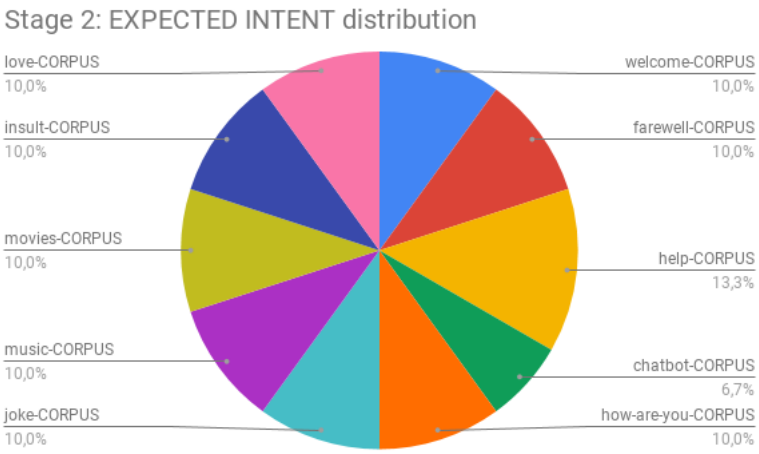

Figure 7.5: Stage 2: Expected distribution.

Table 7.6 shows the results of the experiment after add the Stage 2 to the general Corpus of intents. The global analysis will be done after studying each distribution in every stage.

When running this experiment, a problem derived from using a Wor2Vec or GloVe model is observed. Two totally different intents have been added, one is *love-CORPUS*, and the other is *insult-CORPUS*, in sentence s2_10, the System predicts that the topic of conversation is *love-CORPUS*, but the user is telling it *I hate you*. The words of GloVe models are placed in a plane depending on their appearance in structures in different texts, if you compare where are located the word *love* and the word *hate* in those dimensions, they would be very close. That is why the agent detects the intent of love and not the insult.

Figures 7.5 and 7.6 prove that the success of the detection of intents is still optimal, the distributions of both figures are very similar, every intent still appear.

Finally, the intents of **Stage 3** are added, thus completing the system. The experiments of this stage are the most important because they will reveal if the detection algorithm is good or not.

The table 7.7 shows the results after inserting the new intents and executing the Intent Detection System again.

As more topics are added, the CA has more problems trying to recognize the intents. That could happen if the training sentences are not good defined (using similar words

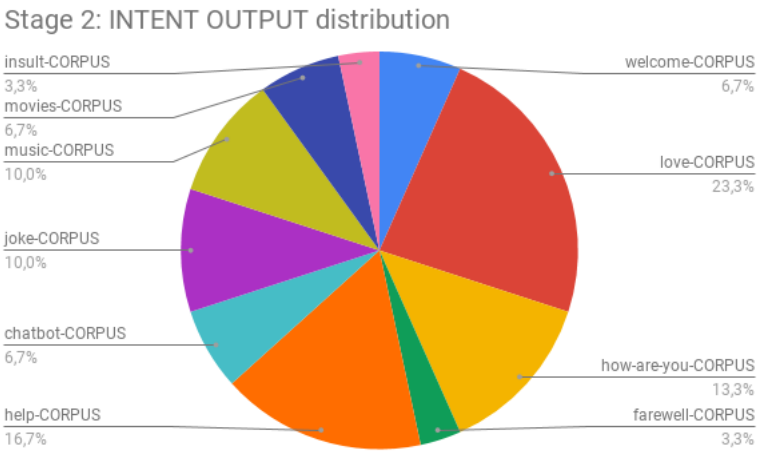

Figure 7.6: Stage 2: Output distribution.

between two sentences of two different intents for example). As a general rule, in a company dedicated to the creation of these systems of dialogue, they have several employees focused on the linguistic theme. Figures 7.7 and 7.8 show how some intents are overlapping the rest, making the system have a tendency to detect those intents more often.

In the results of **Stage 3**, most interactions fail, but global results are still promising because these failures have an explanation:

- **Low Quality Training Sentences**: The declared training sentences are not previously studied, the system has 15 different topics and in each topic are 5 sentences as maximum. With a good study of the final user and choosing the most promising verbosity and sentence structure, the accurate of the System would increase.
- **Confused Intents**: The figure shows the problem and the solution, the *football-CORPUS* and *hobbies-CORPUS* are concepts related and the CA could be confuse. To avoid this problem, the most recommendable is join both intents in just one, in that way, the problems with the confusion would be solved.

The conclusions are that as the number of intents in the system increases, one must be careful when putting them in a context and make a study of the training phrases, so that they conform to the main concepts. They are good results, the system manages to hit 62 % (Table 7.8) of the time without any previous study of the user. The Stages 1 and 2 are a success, but the Stage 3 presents problems. Looking at the results 7.7 and the scoring column, the values are in a range between 1 and 0.57, but it shows that the system can fail even though it gives a very high weight to that intent, and it can

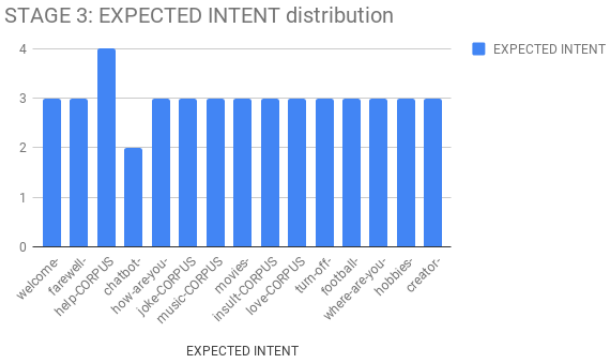

Figure 7.7: Stage 3: Expected distribution.

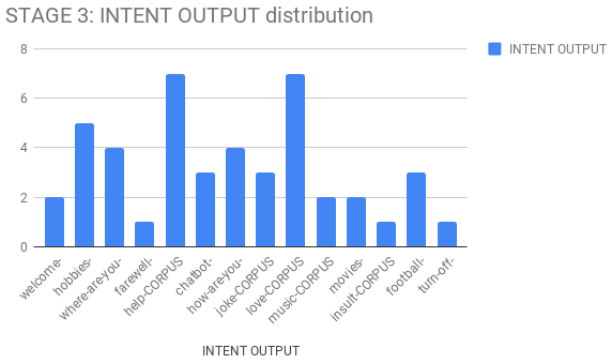

Figure 7.8: Stage 3: Output distribution.

be right even if it has a very low weight (as it is the case of s2_15 with 0.57 score). To conclude, the score is not as relevant as it could be at the beginning, so in the following experiments that column will be ignored.

7.1.3. Modifying GloVe Dimensions

In 5, the GloVe model was introduced and explained. In these experiments, the main goal is to study the behavior of the model changing the dimensions where the words are represented and compare the results with the other models.

To perform these experiments, they do not analyze deeply the intent detection, they will focus only in the accuracy of each stage to see if there are similar or not.

In the table 7.9, the success percentages are shown with each modification of the dimensions of the model. Analyzing this table, you can quickly see that the **50-dimensional model** is the one that gets the worst results, the interesting thing is in the models of 100, 200 and 300 dimensions.

The **100-dimensional model** improves in stages 1 and 2 comparing it with the 50-dimensional model, but in Stage 3 the percentage drops, obtaining an overall percentage of **62.22 %**. From the experiments with the GloVe of 100 dimensions, Stage 1 reaches its maximum percentage of success, the key in selection of the model will be in stages 2 and 3.

The **200-dimensional model** is able to improve in Stage 3 the rest of models, obtaining a percentage of 46.66 %. The global average of this model is **66.66 %**, being **the**

most efficient model.

Last experiment was performed with the **300-dimensional model**. It has not managed to overcome the 200-dimensional model, it worsens its results in scenario 2 and 3, having an overall percentage of 62.22 %. In conclusion, using 200 dimensions the maximum percentage of accuracy of the intents detection system is reached

7.2. LSA test

This CA is based on Socratic educational models, is capable of extracting concepts from a text and offering them to the user in a simple way. Internally, this process is carried out by means of an algorithm called LSA.

In this last part of the experimentation, texts from several entities extracted from Wikipedia will be used, and the LSA algorithm will be applied to these documents, varying the text size to see which one obtains the best concepts. A concept will be classified as good when it is a noun that can be re-searched in Wikipedia and extract more concepts as a result of it.

To begin this section of the documentation, 5 entities are chosen randomly to perform the algorithm on their texts extracted from Wikipedia. These are the chosen entities:

- Alicante.
- Barack Obama.
- Cristiano Ronaldo.
- Tesla Motors.
- Samsung.

This experiment is made up of two different phases. The first will have a Wikipedia text of 11 sentences, and the second will have the total content of the article. The objective of this experiment is to know if the concepts extracted from the whole article (better scenario because it has a longer text and the algorithm behaves better) are similar to those extracted from a summary of less lines.

Figures 7.9 and 7.10 show the result of applying LSA on those terms. Before discussing the selected terms, the graphs will be analyzed first.

These terms are only 10 concepts with the highest score. It can be seen that both show results between 0.1 and 0.25, so the difference in importance of the terms is minimal. The resulting terms show a pattern, the graph 7.10 shows how there are 3 concepts that stand out above the others with more than 0.175 points, then there are two terms with 0.15 and the rest between 0.10 and 0.12.

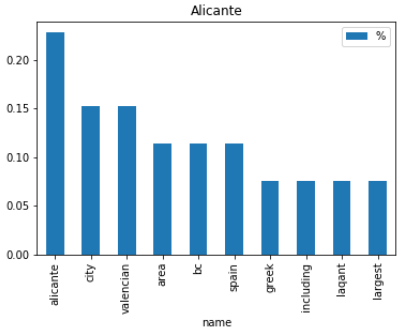

Figure 7.9: LSA Applied on *Alicante* (short text).

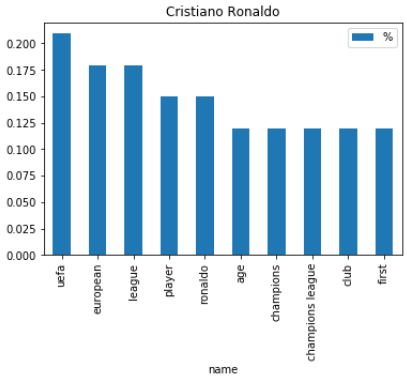

Figure 7.10: LSA Applied on *Cristiano Ronaldo* (short text).

The table 7.10 contains the first ten terms of each entity after applying the algorithm to the short text. We see that the quality of the results is not bad, in the *Cristiano Ronaldo* entity, concepts have been obtained that can then be analyzed again. In contrast, the *Samsung* entity has not given good results because the terms with more value would not serve to seek more information related to this entity.

Now the same procedure will be performed with the same entities, but using the entire document as the input of the algorithm.

The figures show the new results applying the LSA algorithm on the complete Wikipedia article of those terms. Now the graphics no longer have the shape they had before, the distribution now remains linear but more continuous and not staggered. The values have also been affected, the first concept has much more value than the rest.

Table 7.11 has the new terms extracted after running the algorithm again. It is observed that there have been changes in the concepts, they will be analyzed and compared with the previous ones to see what size of input is more suitable.

The most important quality change has been that of *Samsung*, now they can relate their most valued concepts to the entity. The quality in the rest is not very different, but the results of now are better less in *Tesla* because *2017* and *2016* appear as main concepts, being less valued terms such as *electric* or *car*.

Analyzing both sets of responses, it can be concluded that the quality difference between

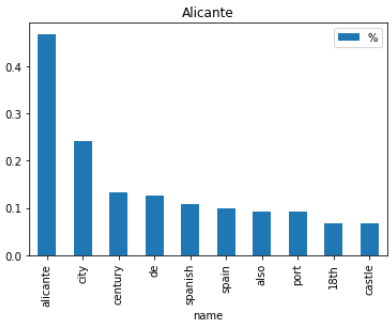

Figure 7.11: LSA Applied on *Alicante* (full text).

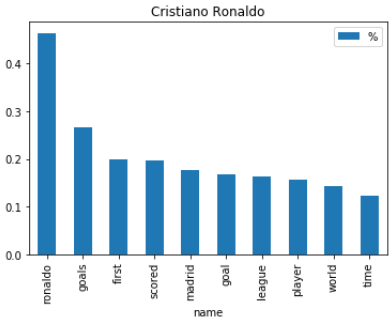

Figure 7.12: LSA Applied on *Cristiano Ronaldo* (full text).

performing the LSA with a short text of 11 lines as used in the first experiment, is not much compared to the result applying full texts. In addition, when sending short texts as data, it is more efficient temporarily. So a short text of 11 sentences is enough to extract important concepts.

ID	USER INPUT	EXPECTED INTENT	INTENT OUTPUT	SCORE
s1_1	Hello!	welcome-CORPUS	welcome-CORPUS	1.00
s1_2	Hi! My name is James	welcome-CORPUS	love-CORPUS	0.53
s1_3	Hey my friend	welcome-CORPUS	welcome-CORPUS	0.77
s1_4	See you later	farewell-CORPUS	how-are-you-CORPUS	0.71
s1_5	Bye bye!	farewell-CORPUS	farewell-CORPUS	1.00
s1_6	Have a good day	farewell-CORPUS	help-CORPUS	0.73
s1_7	I have a doubt	help-CORPUS	help-CORPUS	0.85
s1_8	Help me please	help-CORPUS	help-CORPUS	0.78
s1_9	I need your help	help-CORPUS	help-CORPUS	1.00
s1_10	Can you help me?	help-CORPUS	help-CORPUS	0.91
s1_11	Are you a chat bot?	chatbot-CORPUS	chatbot-CORPUS	0.75
s1_12	You are a robot	chatbot-CORPUS	chatbot-CORPUS	0.63
s1_13	What is going on?	how-are-you-CORPUS	how-are-you-CORPUS	0.89
s1_14	How are you	how-are-you-CORPUS	how-are-you-CORPUS	1.00
s1_15	Everything okay?	how-are-you-CORPUS	how-are-you-CORPUS	0.74
s2_1	Tell me a joke	joke-CORPUS	joke-CORPUS	1
s2_2	a joke please	joke-CORPUS	joke-CORPUS	0.85
s2_3	Do you know jokes?	joke-CORPUS	joke-CORPUS	0.84
s2_4	I love music	music-CORPUS	love-CORPUS	0.86
s2_5	Play music	music-CORPUS	music-CORPUS	0.81
s2_6	What is your favorite song	music-CORPUS	music-CORPUS	0.8
s2_7	Do you like movies?	movies-CORPUS	love-CORPUS	0.84
s2_8	Tell me a movie	movies-CORPUS	movies-CORPUS	0.77
s2_9	Watch a movie on television	movies-CORPUS	movies-CORPUS	0.73
s2_10	I hate you	insult-CORPUS	love-CORPUS	0.78
s2_11	You are so idiot	insult-CORPUS	insult-CORPUS	0.7
s2_12	You are a dick head	insult-CORPUS	music-CORPUS	0.57
s2_13	I think I love you	love-CORPUS	love-CORPUS	0.97
s2_14	Do you want to be my girlfriend	love-CORPUS	love-CORPUS	0.82
s2_15	Marry me!	love-CORPUS	love-CORPUS	0.57

Table 7.6: Results of the Intent Detection System in the **Stage 2**.

ID	USER INPUT	EXPECTED INTENT	INTENT OUTPUT	SCORE
s1_1	Hello!	welcome-CORPUS	welcome-CORPUS	1.00
s1_2	Hi! My name is James	welcome-CORPUS	hobbies-CORPUS	0.55
s1_3	Hey my friend	welcome-CORPUS	welcome-CORPUS	0.71
s1_4	See you later	farewell-CORPUS	where-are-you-CORPUS	0.72
s1_5	Bye bye!	farewell-CORPUS	farewell-CORPUS	1.00
s1_6	Have a good day	farewell-CORPUS	help-CORPUS	0.73
s1_7	I have a doubt	help-CORPUS	help-CORPUS	0.85
s1_8	Help me please	help-CORPUS	help-CORPUS	0.78
s1_9	I need your help	help-CORPUS	help-CORPUS	1.00
s1_10	Can you help me?	help-CORPUS	help-CORPUS	0.91
s1_11	Are you a chat bot?	chatbot-CORPUS	chatbot-CORPUS	0.75
s1_12	You are a robot	chatbot-CORPUS	chatbot-CORPUS	0.63
s1_13	What is going on?	how-are-you-CORPUS	how-are-you-CORPUS	0.89
s1_14	How are you	how-are-you-CORPUS	how-are-you-CORPUS	1.00
s1_15	Everything okay?	how-are-you-CORPUS	how-are-you-CORPUS	0.74
s2_1	Tell me a joke	joke-CORPUS	joke-CORPUS	1
s2_2	a joke please	joke-CORPUS	joke-CORPUS	0.85
s2_3	Do you know jokes?	joke-CORPUS	joke-CORPUS	0.84
s2_4	I love music	music-CORPUS	love-CORPUS	0.86
s2_5	Play music	music-CORPUS	music-CORPUS	0.81
s2_6	What is your favorite song	music-CORPUS	music-CORPUS	0.8
s2_7	Do you like movies?	movies-CORPUS	hobbies-CORPUS	0.81
s2_8	Tell me a movie	movies-CORPUS	movies-CORPUS	0.77
s2_9	Watch a movie on television	movies-CORPUS	movies-CORPUS	0.73
s2_10	I hate you	insult-CORPUS	love-CORPUS	0.78
s2_11	You are so idiot	insult-CORPUS	insult-CORPUS	0.7
s2_12	You are a dick head	insult-CORPUS	hobbies-CORPUS	0.6
s2_13	I think I love you	love-CORPUS	love-CORPUS	0.97
s2_14	Do you want to be my girlfriend	love-CORPUS	love-CORPUS	0.82
s2_15	Marry me!	love-CORPUS	love-CORPUS	0.57
s3_1	Go out	turn-off-CORPUS	love-CORPUS	0.86
s3_2	Close	turn-off-CORPUS	where-are-you-CORPUS	0.64
s3_3	Turn off the agent	turn-off-CORPUS	chatbot-CORPUS	0.67
s3_4	Do you play football?	football-CORPUS	hobbies-CORPUS	0.88
s3_5	Put football on television	football-CORPUS	football-CORPUS	0.73
s3_6	I like football	football-CORPUS	football-CORPUS	1
s3_7	Where are you right now?	where-are-you-CORPUS	where-are-you-CORPUS	0.79
s3_8	Where do you live	where-are-you-CORPUS	where-are-you-CORPUS	1
s3_9	from which country do you come from	where-are-you-CORPUS	turn-off-CORPUS	0.74
s3_10	Do you have hobbies?	hobbies-CORPUS	love-CORPUS	0.62
s3_11	I play basketball	hobbies-CORPUS	football-CORPUS	0.84
s3_12	I like to swim	hobbies-CORPUS	hobbies-CORPUS	1
s3_13	How is your creator?	creator-CORPUS	how-are-you-CORPUS	0.66
s3_14	who made you?	creator-CORPUS	help-CORPUS	0.71
s3_15	I create you	creator-CORPUS	help-CORPUS	0.87

Table 7.7: Results of the Intent Detection System in the **Stage 3**.

	Stage 1	Stage 2	Stage 3	Total
Hits	12	11	5	28
Errors	3	4	10	17
Average	80 %	73.33 %	33.33 %	62.22 %

Table 7.8: Global Results.

	50 dimensions	100 dimensions	200 dimensions	300 dimensions
Stage 1	73.33 %	80 %	80 %	80 %
Stage 2	60 %	73.33 %	73.3 %	66.66 %
Stage 3	40 %	33.33 %	46.66 %	40 %
Total	57.77 %	62.22 %	66.66 %	62.22 %

Table 7.9: GloVe Dimensions Results.

Alicante		Tesla Motors		Cristiano Ronaldo		Samsung		Barack Obama	
Term	Score	Term	Score	Term	Score	Term	Score	Term	Score
Alicante	0.228	Tesla	0.324	Uefa	0.209	Samsung	0.479	Law	0.159
City	0.152	Company	0.259	European	0.179	Largest	0.169	2004	0.119
Valencian	0.152	Model	0.226	League	0.179	Group	0.141	President	0.119
Area	0.114	Electric	0.097	Player	0.149	South	0.141	Senate	0.119
bc	0.114	Solar	0.097	Ronaldo	0.149	World	0.141	1961	0.079
Spain	0.114	Vehicles	0.097	Age	0.119	Electronics	0.112	20	0.079
Greek	0.076	000	0.064	Champions	0.119	Company	0.112	2008	0.079
Including	0.076	2003	0.064	Champions League	0.119	Insurance	0.084	American	0.079
laquant	0.076	2015	0.064	Club	0.119	Korea	0.084	Born	0.079
largest	0.076	2017	0.064	First	0.119	Koean	0.084	Campaign	0.079

Table 7.10: LSA on short text.

Alicante		Tesla Motors		Cristiano Ronaldo		Samsung		Barack Obama	
Term	Score	Term	Score	Term	Score	Term	Score	Term	Score
Alicante	0.468	Tesla	0.702	Ronaldo	0.463	Samsung	0.798	Obama	0.707
City	0.242	Model	0.306	Goals	0.265	Korea	0.137	President	0.159
Century	0.133	2017	0.151	First	0.199	Company	0.127	First	0.122
De	0.125	2016	0.136	Scored	0.197	Group	0.115	2009	0.115
Spanish	0.108	company	0.136	Madrid	0.177	Exchange	0.102	act	0.115
Spain	0.100	musk	0.136	Goal	0.167	Electronics	0.087	United	0.108
Also	0.091	000	0.129	League	0.163	Venture	0.082	States	0.106
Port	0.091	battery	0.102	Player	0.157	Insurance	0.075	2010	0.104
18th	0.066	electric	0.102	World	0.143	Joint	0.072	United States	0.090
Castle	0.066	car	0.090	Time	0.122	Largest	0.072	Law	0.078

Table 7.11: LSA on full text.

8 Conclusions

This chapter will focus on the evaluation of results and the review of the entire project, analyzing if the proposed objectives at the beginning of the work have been achieved. There will be a reflection on the future work that could be added as an improvement too.

8.1. Project Summary

This project has been a challenge for me because it is the first time that I have proposed to create a CA and make it able to be connected with different data sources to try to create a feeling of a Open Domain Dialogue System. There are not many studies on the connection of a chatbot with Open Data, and that is what I liked most about the project, which was not common. If we take this previous idea and we added a extra task of giving to the user new information, we obtain an ambitious project.

In this project, a CA is presented, capable of recognizing multiple conversation topics, searching for information that it does not know and offering extra knowledge to the user. To create the intent recognition system, I used GloVe created by Stanford University, using an architecture of intent detection, and I created a formula to balance the related percentages of each conversation intent, also I readjusted the training phrases to achieve an increase of accuracy.

Depending on the topic of the conversation, I made a action handler that identified if the intent had associated a data search action in Wikipedia or Data.Gov, and if it did, special templates to offer the answer clearly to the user.

Once I had done the intents detection system and the action handler, I looked for ways to extract topics related to the conversation and be able to offer the user more relevant information of the conversation.

8.2. Evaluation of results

The results obtained in the Experimental section 7 are promising. In the tests of the Intent Detection System, that is the most important part because is the base of the system. It shows with the architecture implemented, the system can recognize many conversation topics if they have been previously well defined.

Once the base of the system is robust and well structured, experiments are focusing

now on the LSA algorithm to see if it is able to extract the concepts properly in texts of different sizes.

Both tests have been successful, so the proper functioning of the program is assured.

The designed architecture also has several restrictions that make the chatbot unable to show its full potential. The most important restriction is if the intents are not well defined, the detection of the topic of dialogue will not be optimal, thus making the agent not work correctly.

Other problem is directly related to the GloVe model. Although two words are antonyms, if their position on the plane is similar, they will have the same value in the phrase. If the word has a misspelling, the model does not recognize it and does not take it into account to evaluate the phrase globally, so the user should speak correctly and use terms that are present in the GloVe.

The search of data in the Open Data portal is very tedious and complex, if the system does not find a set of data that fits the user's request and the characteristics that it must present in order to extract information, the agent will not return data. In the Wikipedia search, if the term to search is not clear and there are several documents that can be chosen, the agent does not return the information requested by the user.

If it works on how to deal with these problems and resolve them, the CA would show its full potential and could be trained to solve different specific objectives (GO chatbot).

8.3. Further work

In this section, an analysis of what functionalities can be added to the agent to improve its quality will be performed. There are many approaches to improve the CA, from a designing view until change the architecture of the whole Intent Detection System.

1. **Create a chat view**: This CA only has academic purposes, its designed to make this report and study the SOTA of chatbots. It is a good idea to design a user-friendly chat view to make public the project and let real users test the System.

2. **Publish the CA in a server**: With the same goal than the previous improvement, publishing the chatbot to people would help the definition of every intent and the creation of the training sentences.

3. **RNN addition**: Change the architecture of the GloVe by a trained RNN, continuously learning from the user inputs. Recent neural models of dialogue generation offer great promise for generating responses for CA. In example, LSTM sequence-to-sequence model is one type of neural generation model that maximizes the probability of generating a response given the previous dialogue turn.

List of Acronyms

AI: Artificial Intelligence
NLP: Natural Language Processing
NLG: Natural Language Generation
QA: Question-answering
NN: Neural Network
SOTA: State Of The Art
RNN: Recurrent Neural Network
LSTM: Long Short-Term Memory
DP: Deep Learning
GAN: Generative Adversarial Networks
CA: Conversational Agent
SDS: Spoken Dialogue System
GO: Goal Oriented
NER: Named-Entity Recognition
TF-IDF: Term Frequency–Inverse Document Frequency
IDE: Integrated Development Environment
NLTK: Natural Language Toolkit
CKAN: Comprehensive Knowledge Archive Network
API: Application Programming Interface
GloVe: Global Vectors for Word Representation
LSA: Latent Semantic Analysis
IE: Information Extraction
SM: Socratic Method

Bibliography

[Aparicio García et al., 2016] Aparicio García, J. M., Fuster-Guilló, A., Garrigós Fernández, I., Maciá Pérez, F., Mazón López, J. N., Vaquer Gregori, L., and Zubcoff, J. (2016). Ecosistema de datos abiertos de la universidad de alicante.

[Bećirović, 2016] Bećirović, S. (2016). Socratic method as an approach to teaching. 111:511–517.

[Cabot et al., 2010] Cabot, J., Mazón, J.-N., Pardillo, J., and Trujillo, J. (2010). Specifying aggregation functions in multidimensional models with ocl. pages 419–432.

[Chan, 2013] Chan, C. M. L. (2013). From open data to open innovation strategies: Creating e-services using open government data. In *2013 46th Hawaii International Conference on System Sciences*, pages 1890–1899.

[Chowdhury, 2007] Chowdhury, G. G. (2007). Natural language processing.

[Diksha Khurana, 2017] Diksha Khurana, Aditya Koli, K. K. S. S. (2017). Natural language processing: State of the art, current trends and challenges.

[Freitag and Roy, 2018] Freitag, M. and Roy, S. (2018). Unsupervised natural language generation with denoising autoencoders.

[Henderson, 2015] Henderson, M. S. (2015). Discriminative methods for statistical spoken dialogue systems (doctoral thesis).

[Heng Wang, 2017] Heng Wang, Zengchang Qin, T. W. (2017). Text generation based on generative adversarial nets with latent variable.

[Higashinaka et al., 2014] Higashinaka, R., Imamura, K., Meguro, T., Miyazaki, C., Kobayashi, N., Sugiyama, H., Hirano, T., Makino, T., and Matsuo, Y. (2014). Towards an open-domain conversational system fully based on natural language processing. In *COLING*.

[Ilievski, 2018] Ilievski, V. (2018). Ecole polytechnique fédérale de lausanne. Master's thesis, The school of the thesis.

[Janssen et al., 2012] Janssen, M., Charalabidis, Y., and Zuiderwijk, A. (2012). Benefits, adoption barriers and myths of open data and open government. *Information Systems Management*, 29(4):258–268.

[Kanya and Ravi, 2012] Kanya, N. and Ravi, T. (2012). Modelings and techniques in named entity recognition-an information extraction task. In *IET Chennai 3rd International on Sustainable Energy and Intelligent Systems (SEISCON 2012)*, pages 1–5.

[Kitchin, 2014] Kitchin, R. (2014). *The Data Revolution*. SAGE.

[Krasadakis, 2018] Krasadakis, G. (2018). How ai is changing the world. [Online; posted 12-January-2018].

[Kyunghyun Cho, 2014] Kyunghyun Cho, Bart van Merrienboer, C. G. D. B. F. B. H. S. Y. B. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation.

[Lopez, 2018] Lopez, C. (2018). Las interacciones de lenguaje natural más comunes con un chatbot. [Online; posted 1-March-2018].

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

[Stott, 2014] Stott, A. (2014). Open data for economic growth.

[Zue et al., 2000] Zue, V., Seneff, S., Glass, J. R., Polifroni, J., Pao, C., Hazen, T. J., and Hetherington, L. (2000). Juplter: a telephone-based conversational interface for weather information. *IEEE Transactions on Speech and Audio Processing*, 8(1):85–96.