KINO: AN APPROACH FOR RULE-BASED CHATBOT DEVELOPMENT, MONITORING AND EVALUATION

THIAGO CARVALHO D'ÁVILA

KINO: AN APPROACH FOR RULE-BASED CHATBOT DEVELOPMENT, MONITORING AND EVALUATION

Dissertação apresentada ao Programa de Pós-Graduação em Computer Science do Instituto de Ciências Exatas da Federal University of Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Computer Science.

Orientador: Pedro Olmo Stancioli Vaz de Melo

Belo Horizonte
Agosto de 2018

THIAGO CARVALHO D'ÁVILA

KINO: AN APPROACH FOR RULE-BASED CHATBOT DEVELOPMENT, MONITORING AND EVALUATION

Dissertation presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

Advisor: Pedro Olmo Stancioli Vaz de Melo

Belo Horizonte
August 2018

Ficha catalográfica elaborada pela Biblioteca do ICEx - UFMG

D’Ávila, Thiago Carvalho.

D255k Kino: an approach for rule-based chatbot development, monitoring and evaluation / Thiago Carvalho D’Ávila — Belo Horizonte, 2018.
xxvi, 65 f. il.; 29 cm.

Dissertação (mestrado) - Universidade Federal de Minas Gerais – Departamento de Ciência da Computação.

Orientador: Pedro Olmo Stancioli Vaz de Melo

1. Computação – Teses. 2. Inteligência artificial. 3. Chatbot. 4. Agente da conversação. I. Orientador. II. Título.

CDU 519.6*82(043)

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Kino: an approach for rule-based chatbot development, monitoring and evaluation

THIAGO CARVALHO D'AVILA

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

[illegible]
PROF. PEDRO OLMO STANCIOLI VAZ DE MELO - Orientador
Departamento de Ciência da Computação - UFMG

[illegible]
PROFA. MARIA DAS GRAÇAS VOLPE NUNES
Departamento de Ciência da Computação - USP

[illegible]
PROF. FLAVIO VINICIUS DINIZ DE FIGUEIREDO
Departamento Ciência da Computação - UFMG

[illegible]
PROF. LUIZ CHAIMOWICZ
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 24 de agosto de 2018.

For all those who work like robots, with the hope that they find humaneness.

Acknowledgments

First and foremost, I would like to thank Celio Faria, my friend, and colleague for the great enthusiasm and help with the chatbots theme.

Next, I would like to thank Raquel, my girlfriend, for her love and patience with me during these years. A heartfelt thank you to my parents, Luis and Carmen, for the sacrifices they have made so as to give me all the support and opportunities I had. And to my brother and partner of adventures, Diogo, for his inspiring curiosity.

I also want to thank all my teachers and professors, especially my advisor Dr. Pedro for always questioning all my answers. And, Dr. Chaimo, who presented me the subject and was also my advisor in graduation. And also, Dr. Graça and Dr. Flávio for the kind contributions.

I extend my gratitude to Marcelo for helping me with experiment analysis and Emilio Suyama with statistics. Special thanks to Helena Portilho for countless hours of English reviews. And I am also grateful to Fernanda Portilho and Lulu for their artwork and help with social media disclosure.

Thanks to my colleagues at Serpro for the massive participation in the experiments. Big thanks to Gabriel, João Paulo, Kaabah, Beth, Bel, Mauro, Mateus, Valeska and Wellington for spending most of their days with me. And thanks to my friends at WiseDados especially Carolina, Janaina, Marcos, Paulo and Túlio who shared moments of crashing smartphones during my tests.

Special thanks to Alice, Aline, Bernardo, Davi, David, Eduardo, Fernanda Karla, Luana, Luiza, Luã, Michelle, Miguel, Odon, Rodrigo Otávio, Samir and Victor who helped me train KINO for the experiment. And to the over 400 people who chatted with KINO and shared the posts about it on social media.

Lastly, I would like to thank the AMs, TNs and all my friends for their presence and for understanding my absence during difficult moments. You are the ones I chose for my life and represent who I am.

"The price of reliability is the pursuit of the utmost simplicity."

(Charles Antony Richard Hoare)

Resumo

Chatbots são programas de computador que atuam como agentes conversacionais inventados na década de 1960 respondendo em linguagem humana. Ainda há muitos desafios e oportunidades a serem explorados, apesar de mais de cinquenta anos se passarem desde a criação do primeiro chatbot. O objetivo desta pesquisa é desenvolver a implementação, avaliação e monitoramento de um chatbot baseado em regras. Tendo isso em mente, KINO, um bot que responde a perguntas sobre filmes, foi desenvolvido. A arquitetura do KINO é apresentada, juntamente com desafios e soluções propostas descritas como uma referência para o desenvolvimento de outros chatbots. Um experimento foi realizado no Facebook com mais de 300 usuários gerando métricas automatizadas de assertividade e feedback para avaliação e monitoramento. Essas métricas foram validadas usando diferentes métodos de normalização de texto em português do Brasil: RiveNorm, Enelvo e UGCNormal. Como resultado, a viabilidade da avaliação e monitoramento propostos foi demonstrada, gerando várias análises. Além disso, as análises das métricas coletadas mostraram problemas que permitem manutenção futura e possíveis melhorias do KINO.

Palavras-chave: Chatbot, Sistemas de Diálogo, Agente Conversacional.

Abstract

Chatbots are software conversational agents invented in the 1960's that answer in human language. There are still many challenges and opportunities to be explored, even though more than fifty years have gone by since the first chatbot was created. The aim of this research is to develop the implementation, evaluation, and monitoring for a rule-based chatbot. Having this in mind, KINO, a bot that answers questions about movies, was developed. KINO's architecture is presented, along with challenges and proposed solutions described as a reference for the development of other chatbots. An experiment was performed on Facebook with more than 300 users generating automated metrics of assertiveness and feedback for further evaluation and monitoring. These metrics were validated using different Brazilian Portuguese text normalization methods: RiveNorm, Enelvo, and UGCNormal. As a result, the feasibility of evaluation and monitoring proposed has been demonstrated, yielding various analyses. Furthermore, the analyses of the collected metrics have shown issues that allow future maintenance and possible improvements of KINO.

Keywords: Chatbot, Dialogue Systems, Conversational Agent.

List of Figures

List of Tables

Acronym List

AIML	Artificial Intelligence Markup Language
ALICE	Artificial Linguistic Internet Computer Entity
ANN	Artificial Neural Network
AI	Artificial Intelligence
API	Application Programming Interface
DL	Deep Learning
DNN	Deep Neural Network
HTTPS	Hypertext Transfer Protocol Secure
LSTM	Long short-term memory
MIT	Massachusetts Institute of Technology
MUD	MultiUser Dungeon
NLP	Natural-Language Processing
NLU	Natural Language Understanding
OIG	Original Imitation Game Test
OWL	Web Ontology Language
POS	Part Of Speech
QA	Question Answering
RDF	Resource Description Framework
RL	Reinforcement Learning

RSS	Residual Sum of Square
RTM	Real Time Messaging
SEQ2SEQ	Sequence-to-sequence
SL	Supervised Learning
STC	Short-Text Conversation
STT	Standard Turing Test
TT	Turing Test
UGC	User-Generated Content

Contents

Chapter 1

Introduction

Chatbots are computer programs used to interact in human language (eg. English, Spanish, Portuguese), also known as natural language. The term chatbot was derived from CHATTERBOT, the name of a software robot that used to talk to players in an online computer game through text inputs (chat) [Mauldin, 1994]. Other terms also used to refer to these are bots, dialogue systems, conversational agents and chatterbots.

1.1 Motivation

The chatbots market has shown significant growth with over 180 companies in the industry attracting 6 billion dollars in recent years [Cifuentes, 2018]. Predictions for 2018 show that most of the developed chatbots will still be considered useless, but by the end of the year, there will be bots able to show their transformative potential. At the current pace, in 2020 more bots will be built and launched than mobile applications [Plummer et al., 2017].

According to Gao et al. [2018] dialogue systems are expected to solve the following problems:

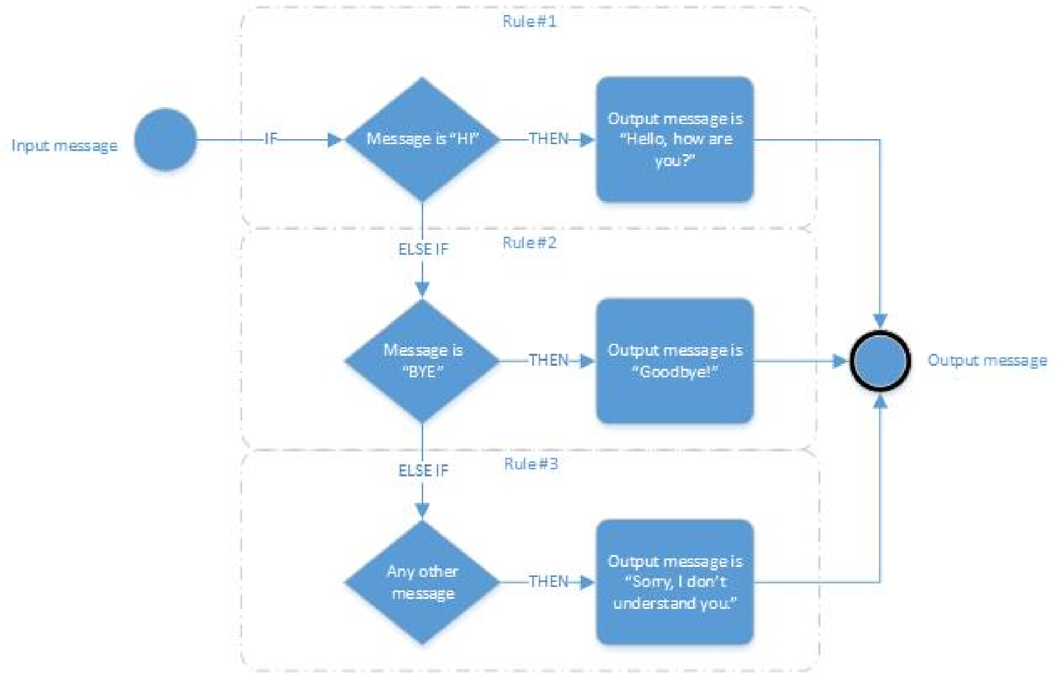
- **Question answering (QA)**: provide concise, direct answers to user queries using data sources;
- **Task completion**: accomplish some task such as making a restaurant reservation or making a business trip planning;
- **Social chat**: converse seamlessly like a human and provide useful recommendations.

The development of the chatbots is a research area in which the problem can be divided into open or closed domain. The open domain configuration is that the user can talk freely without a well-defined direction on any subject. In the closed domain, there is a defined subject or goal, where the conversation should be limited to a specific goal [Majumder et al., 2018]. Open domain bots are more related to social chat whereas the closed domain ones to question answering and task completion.

Several applications are possible even with a closed domain. Deryugina [2010] shows examples in the area of business with virtual assistants and customer service and also in the educational area as tutors in e-learning. Abdul-Kader and Woods [2015] also presented applications in other areas such as health where chatbots can be used to give medical assistance.

A well-known approach for building such bots is rule-based. A rule set is created as their knowledge base to instruct how to answer an input message from the user. A very simple example is to chain IF-THEN-ELSE statements with predefined answers as seen in Figure 1.1.

Figure 1.1. Rule-based example.



This kind of approach has its challenges such as structuring this chain of rules and dealing with multiple ways of saying the same thing. Some of them can also be unknown prior to the development such as how the introduction or update of a new rule changing the answer to previously known answers. Few studies discuss details of chatbot challenges, their impact on the quality of answers and possible solutions

[Berglund, 2017], especially concerning evaluation and monitoring.

Regarding chatbot evaluation, some studies rely on user questionnaires [Maeda and Moraes, 2017] or manual tests [Oliveira, 2017], with their associated costs, making evaluation unfeasible for smaller studies. Although dozens of quality attributes are proposed in the literature, there are no standard metrics, and proposed analytic processes have high complexity [Radziwill and Benton, 2017]. Simpler metrics could be proposed in an attempt to reduce chatbot evaluation costs, enabling broader use in studies and their analysis over time.

Likewise, according to the Scopus platform, between 2002 and 2018, the number of studies in the area has been increasing, with Brazil appearing in the fifteenth place in production with 4 of 273 documents, ahead of countries like France, Finland, and Hong Kong [Sônego et al., 2018]. Although there are studies for the Portuguese language, many difficulties arise because few frameworks and natural language processing algorithms are not available in this particular language. Unfortunately, even when available, they usually have errors or do not work as well as in other languages, such as in English [Maeda and Moraes, 2017].

1.2 Objectives

The aim of this work is to develop an implementation, evaluation, and monitoring for a rule-based chatbot. These three tasks can be defined as follows:

a) Implementation: the process of realization, including the development of the architecture and knowledge base creation;

b) Evaluation: the process of assessing the quality of a chatbot, considering the use of benchmarks for comparison;

c) Monitoring: the process of continuous observation of the operation to maintain and/or evolve the solution.

The following objectives have been set to fulfill the main goal:

- To develop a proof of concept chatbot in Brazilian Portuguese that retrieves information about movies. There is no other bot with these features to the best of our knowledge;
- To identify and enumerate major and minor challenges in the development of a rule-based chatbot, detailing a proposed solution for each one of them;

- To propose and implement a set of simple metrics used for evaluating and monitoring chatbots;
- To perform an experiment to validate the use of the chatbot, the evaluation metrics and monitoring.

1.3 Contributions

The main contribution of this study is a validated development, evaluation and monitoring process of a rule-based chatbot. KINO chatbot was developed and used in an experiment to chat with approximately 300 users on Facebook Messenger. During its development challenges and proposed solutions were registered. Furthermore, evaluation metrics were collected and indicators monitored throughout this experiment.

1.4 Organization

The remainder of this dissertation is structured as follows: Chapter 2 presents an overview and concepts about chatbots, as well as the discussion on artificial intelligence-related issues. It gives more details on the background information of chatbots history and related work; Chapter 3 describes the proposed approach for the architecture, evaluation, and monitoring used in KINO chatbot; Chapter 4 introduces an experimental validation of KINO and the proposed method for evaluating and monitoring rule-based chatbots; Chapter 5 presents the conclusions of this thesis and some possibilities for future work.

Chapter 2

Background and Related Work

2.1 Turing Test

"Can machines think?" This questioning by Alan Turing has led to a discussion about the creation of artificial intelligence. The answer to this question requires the definition of "machine" and "thinking". These terms have a potential for ambiguous understanding and are hard to be broadly defined in a static and absolute manner [Alan Turing, 1950].

Instead of defining it, Turing proposed an "Imitation Game", played by three players. One of the players would be a male, another a female and the third one the interrogator, whose gender would not matter. The goal of the game is for the interrogator to tell which one is the male and the female after talking to each of them for a limited time (e.g. five minutes each). However, one of them would try to mislead the interrogator by answering as the other gender presumably would. The other player should be consistent with his/her gender. Features, such as voice tones, handwriting and appearance, should be isolated in this communication, therefore, the challenge would be increased by using a typewritten channel in which players remain in separate rooms.

Finally, Turing asked what would happen if a machine were to take the place of one of the players in the game and whether the interrogator would make mistakes with the same probability as when the game was played with a male and a female. This questioning became the proposition that replaces the original question: "Can machines think?".

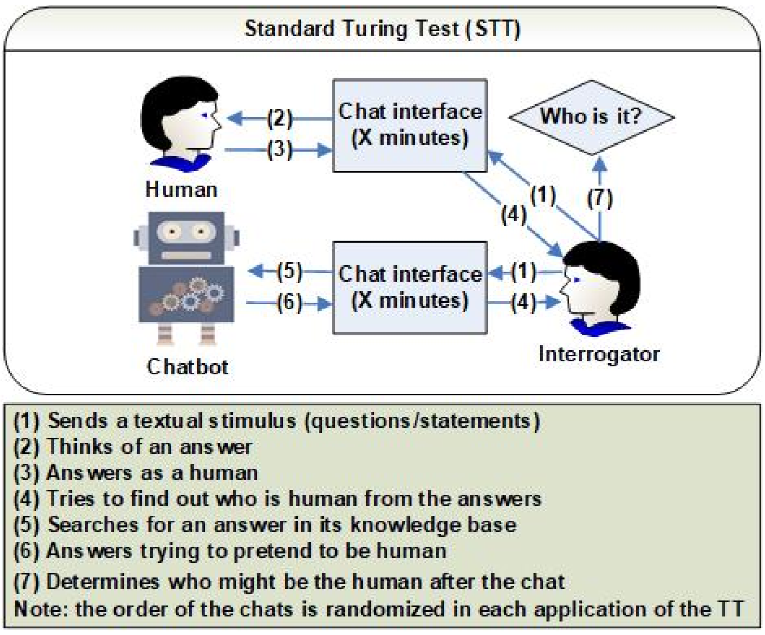
There are two interpretations of this "game of imitation":

1. Whether the machine would try to pass as a male or a female

2. Or the interrogator would have to decide who the human and the machine is, making gender unimportant for all players.

Later, the first game was named Original Imitation Game Test (OIG) and the latter Standard Turing Test (STT) [Sterrett, 2003]. It turned out that these games were not equivalent, and the STT (shown in Figure 2.1) was considered to be a flawed scientific experiment to measure machine's intelligence, although it is still the most common version used [Sterrett, 2003; Wallace, 2009; Lucas, 2009].

Figure 2.1. Standard Turing Test.


Standard Turing Test (STT)
Human
(2)
(3)
Chat interface (X minutes)
Who is it?
(1)
(4)
(7)
Chatbot
(5)
(6)
Chat interface (X minutes)
(1)
(4)
Interrogator

(1) Sends a textual stimulus (questions/statements)
(2) Thinks of an answer
(3) Answers as a human
(4) Tries to find out who is human from the answers
(5) Searches for an answer in its knowledge base
(6) Answers trying to pretend to be human
(7) Determines who might be the human after the chat
Note: the order of the chats is randomized in each application of the TT

At first, this test attempt would be to isolate strictly intellectual abilities to answer our initial question, since the presence of physical abilities in this game would make the differences between humans and machines evident: how does a man compete in a race with an airplane or a machine compete in a beauty contest with a woman? Although Alan Turing [1950] himself criticized his own experiment, showing, for example, how humans are slower and more inaccurate to perform arithmetic operations or how difficult it would be for a machine to write poetry.

Despite this criticism, the author believed that a machine could be constructed to play the game of imitation satisfactorily, simulating this slowness and the arithmetical errors. To do so, it would be necessary to have three basic parts: storage, unit of execution and control. Storage would be a way for the machine to simulate human memory, so it would have to have a virtually unlimited capacity of space to remember everything needed. Perhaps, one way of making this possible would be to create a selective memory, with the function of forgetting or disabling whichever was not relevant, as humans do. The execution module would be a set of operations a computer would be able to do. These operations can be seen as the simplest instructions of a hardware, be they logical, arithmetic, data, among others. Finally, the control part would be responsible for ensuring these operations are followed correctly.

More importantly, according to Alan Turing [1950], this machine has to be designed so as to produce a chain reaction from a series of injected ideas similar to the processes found in a nuclear reactor: a series of external neutrons with sufficient amount of energy is required to generate internal disturbances until it causes the bomb to explode. Likewise, if it were possible to "insert" ideas into a machine until this critical level of "activation energy" was reached, the machine would start a chain reaction producing new ideas and therefore, it would be *thinking*.

Many hindrances were pointed out by Turing for the creation of an intelligent machine, among which, the limitation of the finite states of a digital machine and technological restrictions of processing and storage. Even so, he believed that within fifty years (which would be around the year 2000), these obstacles would be overcome, and there would be computers capable of playing the game of imitation with a less than 70% chance of an average interrogator correctly identifying the human.

Alan Turing's conception of artificial intelligence runs through the idea that a machine should act humanly in order to be able to think. This implies that they must have abilities such as learning, representing knowledge and making mistakes, as shown in Russell and Norvig [2009]. However, in the decade following Alan Turing [1950], the high computational complexity involved in solving such problems comes to light. Possibly, this is one of the main factors that contributes to the failure of their predictions.

2.2 History of chatbots

In the study of artificial intelligence, the concept of intelligent agents is used for individuals that are inserted in an environment, who perceive and interact with it [Russell and Norvig, 2009]. To capture the context in which they are inserted, these agents use their "percepts", analogous to the human senses, usually devices called sensors. In order to perform their interaction with the medium, they use actuators, elements that produce the changes, which will eventually be perceived by feedback by their sensors and other agents in the medium. In this way, the "intelligence" of these agents lies in the computation of the variables observed in the environment and the decision making of which actions will be performed in response to this observed context.

In this context, chatbots are programs or machines capable of interacting in a text-based chat environment. Their perceptions are the inputs of data made by those talking to them. Their actions are the responses generated in the opposite direction to complete the communication.

Fifteen years after Turing, ELIZA, an agent capable of playing the imitation game, was created by Weizenbaum [1966] of the Massachusetts Institute of Technology (MIT). His idea was to create a Rogerian psychotherapist (person-centered approach), which basically answered questions with questions. ELIZA communicated in natural language through a simple program that used a set of transformation *rules*. Its basic operation is based on the identification of a keyword, in the original message, used to find a corresponding rule that describes the transformation of the message into ELIZA's response.

PARRY was a chatbot created in Stanford by 1972. It was an application that simulated paranoid schizophrenic behavior, maintaining internal variables that controlled their emotional state [Colby et al., 1972]. PARRY was put to talk to ELIZA, in a conversation that became known as "PARRY encounters the DOCTOR" (recorded session found in Cerf [1973]). Thereafter, it has become common practice to create experiments in which one chatbot interacts with another.

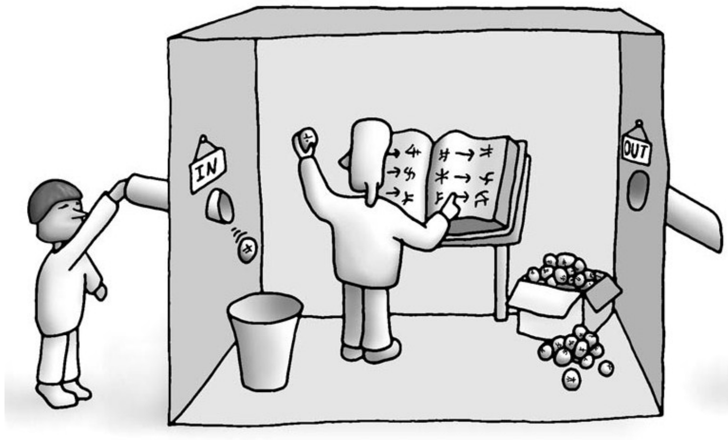
PARRY brought some innovations in relation to ELIZA since the first bot tried to eliminate any personality trait of its answers, which made its logic simpler. A sequence of pattern-matching rules was used to identify the input content and a response module used this content to generate the output considering PARRY's state of needs, desires and interests [Faught et al., 1974].

In Mauldin [1994], a prominent detail is shown in relation to the methodology of tests used on PARRY. Blind tests were performed, in which doctors recorded conversations with PARRY and three other human patients with a diagnosis of schizophrenia and paranoia, and later transcribed the conversations. These transcripts were reviewed by a group of psychiatrists and another group of computer scientists. However, neither group was significantly more successful than the other in attempts to distinguish the program from humans.

At this point, it is worth mentioning the distinction between two proposals of Artificial Intelligence (AI) considered among its theorists: strong AI and weak AI. Strong AI proposes the creation, on the part of man, of a form of intelligence capable of reasoning and solving problems, in a self-conscious way, being possible that the machines equate to or even surpass human reasoning. The Weak AI, however, has an approach that machines would not be able to truly reason and solve problems. Such agents with this characteristic of intelligence would act as if they were intelligent, but would have no self-consciousness or notion of self. In this way, they would, at most, be equal to humans [Russell and Norvig, 2009].

Searle [1980], a philosopher at the University of Berkeley, developed a remarkable critique of Strong AI. He proposed a mental experiment - *Gedanken* - called Chinese Room. Imagine a person placed in an enclosed room with only one slot that fits papers, in the middle of China. This person understands only English, has a rule book in English, a stack of blank paper, another one filled with Chinese codes, with one statement column and one corresponding answer column. This person slices through letters in Chinese, looks in the rulebook as he looks at the letter's statements in the stack of symbols, writes what is in the corresponding answer column and returns the answer to the Chinese through the slot (Figure 2.2).

Figure 2.2. Chinese Room.



Source:
http://deskarati.com/2014/07/01/john-searles-chinese-room-thought-experiment/

This experiment makes an analogy with the thinking machine, proposed in Alan Turing [1950], in which the room simulates the machine; the person would play the role of the processor; the rule book, the controller; stacks of paper, the storage unit. For the Chinese who are outside the room and receive satisfactory answers to their letters, the room is "smart" and knows Chinese. In this way, the system will pass the Turing Test, but neither the person in this room nor the rule book actually understand Chinese. The same would occur with computers: they process input only syntactically without any semantic understanding of the generated contents [Searle, 1980]. However, it is expected that such a machine, passing the Turing Test, will still produce reasonable and comprehensible output to the user.

The Chinese Room incited discussions on the real utility of the Turing Test. However, one of the counter-arguments to Searle's critique is that in a human decision-making simulation, a machine is not expected to actually do so. Likewise, in a fire simulation, people are not expected to burn. If a machine accurately imitated the human decision-making process in a test (like TT), even if it did not understand what it was doing in the same way as a human, there is no point in saying the human's is better [Anderson, 1987]. Regardless of criticism, the TT has became, in fact, the main reference for evaluation of chatbots in the following years.

In 1990, after more than forty years of discussions in the AI area about the Turing Test, Dr. Hugh Loebner, together with the Cambridge Center for Behavioral Studies, inaugurated the Loebner Prize, an annual contest that rewards the chatbot with the most similar to human behavior, based on the STT [Wallace, 2009].

According to Loebner [1994], the award was created in order to improve TT methodology in practice, encourage research in the area of AI and measure the state of the art of chatbot development. Originally, participants submitted their chatbots to be evaluated by a group of adjudicators, who were randomly chatting with a group of humans in this process. They should identify the bots and the humans. In addition to the award for the most humanlike chatbot, there is a prize of $100,000 if any bot can fool all adjudicators and thus pass the TT officially. The rules for the contest are reviewed and improved on an annual basis. An example of such changes was the inclusion of the audio-visual TT modality in 2011.

In the early 1990s, due to the advent of the Internet, chatbots were mass-tested and improved from the input of a large number of people. The Internet has enabled an increased amount of interaction and access from various users, not only those from academic and business environments in which they were usually built.

The CHATTERBOT, built by Mauldin [1994], participated in the Loebner Prize contest in its first three editions: 1991, 1992 and 1993. It was originally a character in a re-implementation of MultiUser Dungeon - MUD, an online game of the early 1990's, called TINYMUD. The CHATTERBOT was successful in the game since it was placed as an ordinary player and people did not suspect that they were under a TT. Therefore, it was reliable until it did something awkward. Various tricks were explored to make people believe this bot was an ordinary player.

During the first editions of the Loebner Prize, there were also new criticisms of the TT and the prize. Shieber [1994] questioned how appropriate a test is, based only on the specific ability to deceive people. It characterized the prize as a way of rewarding cheap tricks such as dislocating or simulating random errors. In this way, he argued that improving the rules of competition would not create better results, since the current technological level was still very restricted and so the Loebner Prize started carrying out totally useless tests from the scientific point of view, causing discredit in the academic community.

Mauldin [1994], on the other hand, argued that avoiding this type of qualitative evaluation of programmed knowledge is precisely the aim of TT because it is a practical test rather than an attempt to answer the question "Can machines think?". He describes some of the best tricks used by ELIZA, PARRY and CHATTERBOT, believing that one day, when a bot actually passes in TT, it will use most of them. ELIZA, for example, created the whole conversation above the lines of its interlocutor, which worked very well as long as the person could talk about himself/herself and be glad to have the attention of ELIZA. PARRY had a greater range of artifices such as: admitting ignorance (showing that it did not know the answer to a question); changing the subject (questioning why the person is talking about it); having a range of stories to tell in a fixed order; if there is no subject, introducing new topics, etc. CHATTERBOT implemented the tricks of the previous ones and developed some new ones based on the context of the conversation, such as:

- Having parts of conversations in its memory, similarly to some chess players who memorize game openings to try to lead the situation;
- Uttering controversial sentences such as "People don't own cats... it's the other way around.";
- Telling humorous jokes and phrases;
- Agreeing with the interlocutor in some cases;
- Searching for news on the internet to use as a subject;
- Simulating typing to include realistic delays, mimicking the rhythm a person would type in sentences.

Since CHATTERBOT, several bots competed and won in the Loebner Prize, although none have passed their Turing Test. A particular case is ALICE (Artificial Linguistic Internet Computer Entity), the winner of the most humanlike chatbot award in the years 2000, 2001 and 2004. Its choice was not by chance since it became the basis for several other Bots (ALICEBOTs). It was the first bot made in a free language called AIML (Artificial Intelligence Markup Language), designed to create bots by the stimulus-response paradigm through the use of categories. A category is a rule defining a structure, composed of a known syntactic pattern and a corresponding response template for this pattern. The received message is matched with the category patterns to get the bot response, an implementation based on Behaviorist theory. An example category for a response "Hello! How are you?" when the input matches "HI" is represented in AIML as follows:

```
<category>
  <pattern>HI</pattern>
  <template>Hello, how are you?</template>
</category>
```

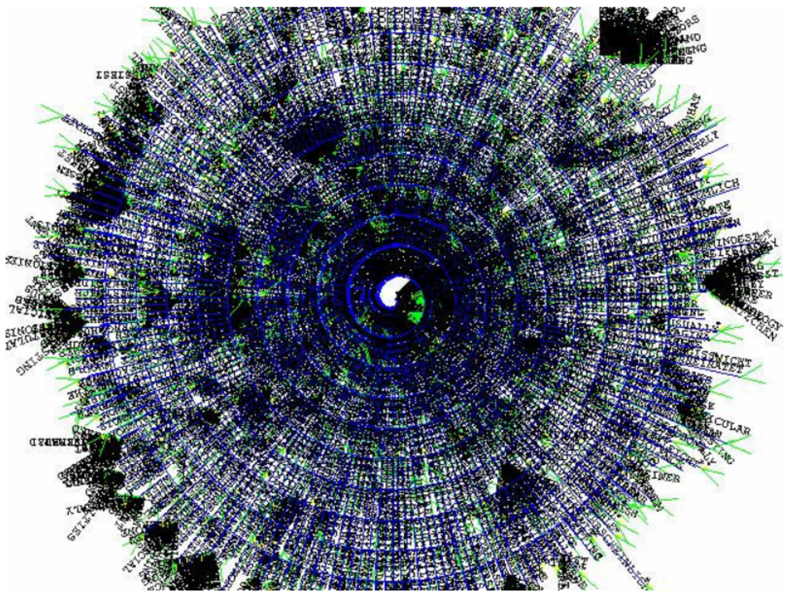
ALICE could be seen as a simple extension of the old ELIZA, considering its stimulus-response architecture. Nonetheless, it had around 200 categories of knowledge, while ALICE had more than 40,000 in 2004, as well as other innovations, having had a great impact on adjudicators' perceptions of intelligence [Wallace, 2009]. Several improvements were observed with the introduction of AIML by ALICE, compared to its precursor ELIZA. However, both bots have conceptual limitations. The model used in both ELIZA and ALICE is one response for each stimulus. In practice, this creates a pattern of conversation distinct from the human, in which there may be an answer to several stimuli or several responses to the same stimulus. It is not necessary to generate a response for every message received. If one of the parties is telling a story, for example, the tendency is for your interlocutor to act as a listener for most sentences without saying anything.

In order to have ALICE's "brain" group all these categories on this unprecedented scale, the use of the Internet to collect data has been critical. This collection was developed by more than 500 volunteers around the world, who played the role of instructors or botmasters, selecting what ALICE should respond in each situation. In Figure 2.3, one can see a representation of the ramifications of the categories of ALICE plotted in a spiral graph.

The learning model used in ALICE is called supervised learning. It is a Machine Learning task that consists of learning a function that maps inputs to outputs from a training set of data. This estimated function is used to yield answers to new inputs. In an optimal situation, this learning algorithm generalizes input messages out of the training data [Alpaydin, 2014]. In the case of ALICE, botmasters play a key role to train bots to perform such task.

On the other hand, the general purpose learning described by Alan Turing [1950] implies a hypothetical robot would grow like a child and learn to respond the same

Figure 2.3. ALICE's "brain" graph plotted as a spiral with 24,000 categories loaded. In the blue line are the choices for the first word with a total of 2001 words. This first level was branched with the corresponding next words of the categories until the third level.



Source: Wallace [2003]

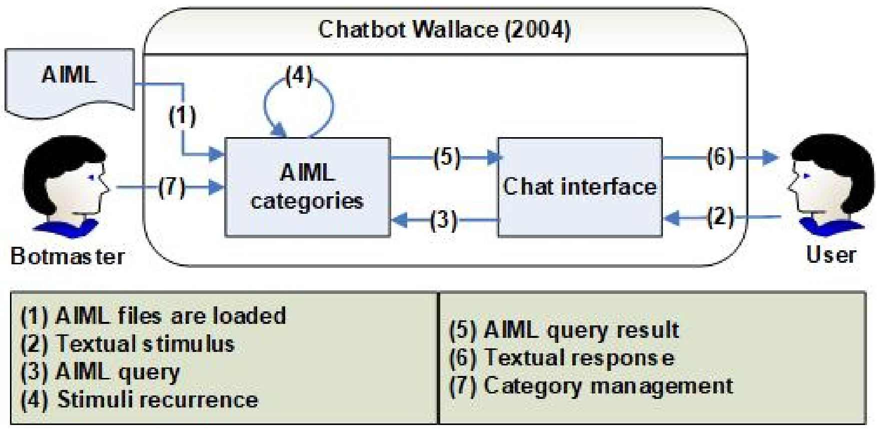
way as humans by itself, without the need of botmasters [Wallace, 2009]. However, a child cannot be left unprotected in the world without qualified supervision. A self-learning bot may be indiscriminately exposed to biased concepts, beliefs or unreliable information, thus replicating them without a filter.

Communication based on pattern matching also generates unwanted distortions. For instance, saying to the bot, "I am BLABLABLA.", with a pattern matching "I am *" (the wildcard matches any words), and a corresponding answer "How long have you been *", the bot is programmed to respond "How long have you been BLABLABLA?", a completely irrational question. Searle [1980] illustrates another problem when telling a machine the following story: "A man went into a restaurant and ordered a hamburger. When the burger was arrived it was as burned to a crisp, the man stormed out of the restaurant angrily, without paying for the hamburger or leaving a tip. Then, the question is: "Did the man eat the hamburger?" The machine would not know what to say, although there were possible answers such as "Yes, he did", "No" or "Probably not". An improvement brought about by AIML that minimizes this problem was the use of internal variables and context tags to identify subjects and facts during the conversation. In this way, the capacity for memorization and consistency in the talks was maximized in ALICE.

Another novelty absent in ELIZA is the use of recursion, which allows several applications such as the creation of synonyms and the combination of categories for the creation of new ones. This resource can be seen as a form of brain connection between the known contents, also increasing the growth potential of the knowledge generated.

In addition, a notable advantage of ALICE, compared to ELIZA, is the ability to scale to a much higher level of knowledge in a number of categories. The use of the internet has played an important role in the training of ALICE, with the collaboration of more than 500 volunteers around the world. The use of AIML in relation to the ELIZA script has brought some advantages by having more structured and documented commands, allowing extension by anyone, without having to know all the pre-existing categories in the script. The development of an automated training module was also essential, with the automatic detection of standards for the correct configuration of the bot, enabling agility in the creation of categories, besides avoiding errors and inconsistencies. Finally, creating an optimized data structure for pattern matching provides more scalability, thus making it possible to load a larger category tree. We can visualize the architecture used by ALICE in Figure 2.4.

Figure 2.4. ALICE's operational architecture.



In relation to learning, both chatbots use supervised learning. However, ALICE has a training module with several automations that allows faster creation of new categories. Even so, it is very complicated to make any comparison to human learning, since the latter is based on experiences and perceptions of all the senses, while a chatbot is mostly limited to pure texting. Even so, knowledge remains an instance of a category in an AIML file. In this way, the acquired knowledge of data on the Internet, for example, will not be updated automatically, and thus this form of static storage of knowledge would not be adequate for dynamic update [Tzou et al., 2007]. Despite the many improvements, ALICE's core is similar to ELIZA's, regarding the generation of stimulus-response, pattern matching, supervised learning, and purely syntactic input analysis.

The following editions of 2003, 2005 and 2006 had winners using Markov Chain Models. Conversation datasets were used to calculate probabilities based on the occurrence of words (or letters) in a sequence with a fixed length. This method can be used with or without pattern matching [Bradeško and Mladenić, 2012]. Rollo Carpenter invented the core concepts of this approach in a chatbot named CLEVERBOT and was later launched as the Cleverscript language [Masche and Le, 2017; Fryer and Carpenter, 2006].

In 2014, a chatbot named EUGENE GOOSTMAN was considered to have passed a Turing Test in the University of Reading contest by fooling one in three adjudicators. It passed off as a 13-year-old Ukrainian boy who was not a native English speaker. The use of this artifice did not sustain its exposure to the public since people quickly discovered that it was a robot [Adams, 2017].

Until 2018, no chatbot passed the Turing Test in a Loebner Prize. Some chatbots managed to pass the TT outside this contest, not for their intelligence but for human ingenuity [Lucas, 2009]. Nevertheless, competitions like the Loebner Prize stimulate the development of artificial intelligence and allow comparison between different approaches [Bradeško and Mladenić, 2012].

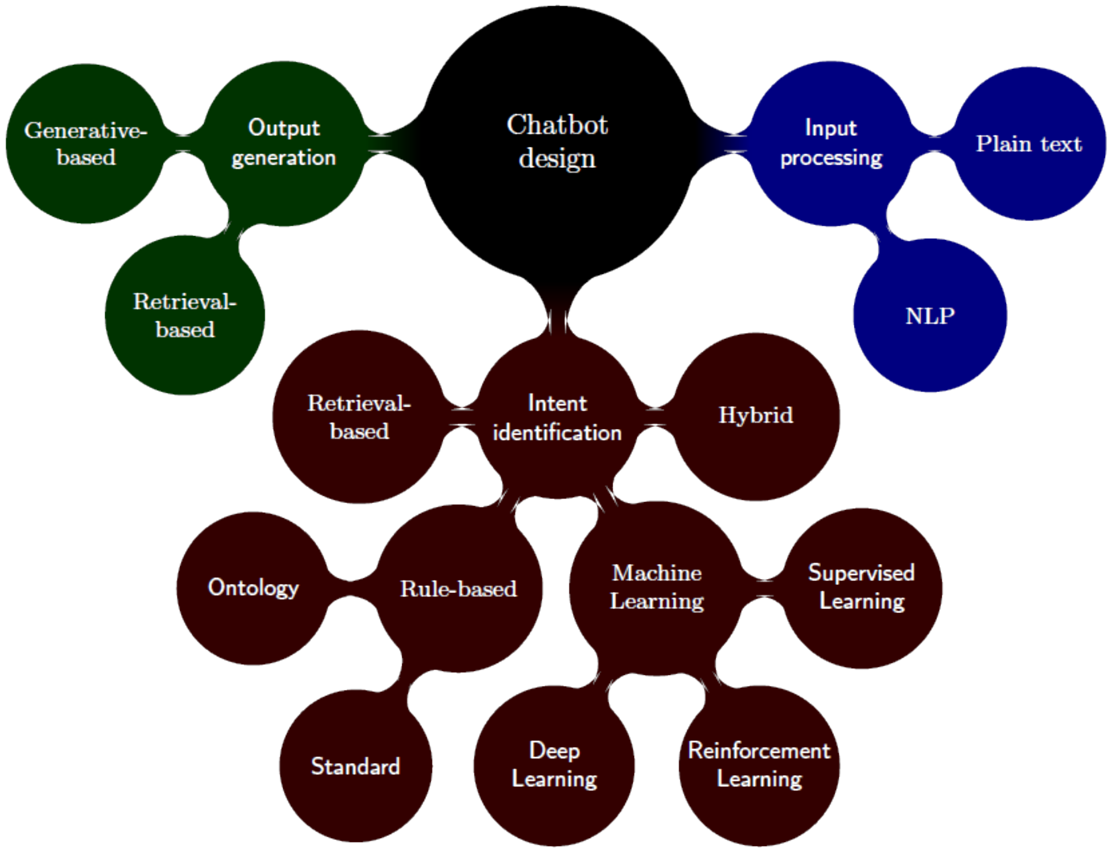
MITSUKO and ROSE, both rule-based approaches, won the competitions from 2013 to 2018. Other strategies were used in academia and industry, with tendencies to use more NLP techniques, bigger datasets and also Machine Learning [Masche and Le, 2017; Gao et al., 2018].

2.3 Contemporary approaches

Chatbot approaches consist of a combination of design techniques categorized according to how their input is processed; the intent of the input is identified; the output is generated. Figure 2.5 summarizes these categorizations.

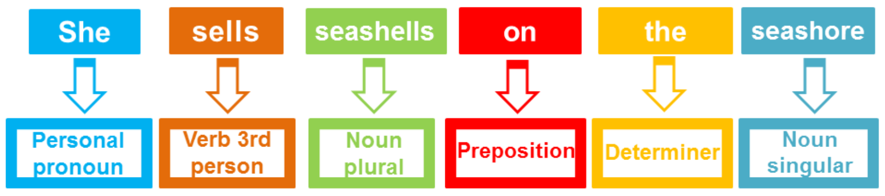
Input processing is how the user's message is pre-processed before intent identification. Plain text inputs keep all the lexis, letter cases, punctuation and accents unchanged. ELIZA is an example of raw input chatbot. Natural Language Processing offers some possible processing choices such as lower-casing, dropping punctuation, accents and/or special characters. Lists of substitutions are also used as a form of normalization with a database table [Silva, 2012] or a file of abbreviations and colloquialisms [Moraes and de Souza, 2015]. More complex NLP methods for normalization, stem-

Figure 2.5. Chatbot design structure.



ming, POS tagging (Figure 2.6), word/phrase analyzers can also be used [Higashinaka et al., 2014].

Figure 2.6. POS tagging example.



Input intent identification is how the bot identifies the goals or purposes in user's message request. Rule-based approaches use a set of rules mapping patterns/triggers to corresponding responses. Earlier chatterbots used simple regular expressions evolving into more complex rule-engines such as AIML, Chatscript and Rivescript [Petherbridge, 2018a]. ELIZA and ALICE are rule-based examples. Ontology-based approaches extend standard rules, creating semantic patterns through the use of Resource Description Framework (RDF) and Web Ontology Language (OWL) [Dias et al., 2007; Tzou et al., 2007; Freese, 2007]. In Athreya et al. [2018] a chatbot was created using Rivescript and DBPedia ontology database to search for factual questions as RDF literals with QANARY [Both et al., 2016].

Retrieval-based approaches rely on search algorithms to retrieve the bots reply. This allows the use of structured and/or unstructured data as sources for a response without the need to match the intent with a previously known message or pattern. DocChat uses a collection of unstructured documents to find sentences relevant to question answering and chat through a response ranking model [Yan et al., 2016].

Machine Learning approaches for chatbots use either Supervised Learning (SL), Reinforcement Learning (RL) or Deep Learning (DL). Supervised Learning relies on a set of previous human-labeled question-answer pairs to create a model for mapping inputs into outputs. In contrast, Reinforcement Learning involves interacting and using rewards to learn how to act through Markov models. Lastly, Deep Learning gets semantic and context information from the inputs based on training Deep Neural Networks (DNNs) [Gao et al., 2018].

In Higashinaka et al. [2014], Support Vector Machine (SVM) and Logistic Regression classifiers (Supervised Learning algorithms) were used to identify the intention in the user messages. Li et al. [2016] used Reinforcement Learning by simulating dialogues between two agents with a sequence-to-sequence (seq2seq) model. Artificial Neural Networks (ANN) can also be used to compute the confidence level of an intent or a response in a dataset given the input. IBM Watson, the bot who won the Jeopardy American TV quiz show, uses NLP and Neural Networks to search for answers [Ferrucci et al., 2010]. Kadlec et al. [2015] used the Ubuntu Dialogue Corpus (in English) to compare different ANN architectures to rank the best answers among the utterances in the dataset.

Maeda and Moraes [2017] built a Deep Learning chatbot for the Portuguese language using Long short-term memory (LSTM) seq2seq. The study achieved low performance and had the challenge of the absence of a benchmark corpus since Portuguese has been considered a low-resource language. Apart from this, SL and RL approaches have a difficulty of generalization, not being able to respond adequately to unseen inputs [Gao et al., 2018].

Hybrid approaches are combinations of the previous approaches. Bartl and Spanakis [2017] proposed a method that combines retrieval-based bots with machine learning by retrieving answers from distributed vector representations of the inputs (embeddings) using Ubuntu Dialogue Corpus (in English) and Vodafone corpus (in Dutch).

Output generation is how the bot reply is given [Surmenok, 2016]. In retrieval-based approaches, pre-defined responses are mapped to the intention detected [Yan et al., 2016]. On the other hand, words/phrases are generated from scratch in a way that the responses are not required in generative-based approaches. Shang et al. [2015] proposed a neural network-based response generator for Short-Text Conversation (STC)[1] tasks. Higashinaka et al. [2014] proposed a solution using NLP modules to generate responses in an open-domain chatbot, but the system could not reach the level the AIML rule-based approach used in a comparison through a questionnaire survey.

New approaches could be proposed and other classifications are available. Masche and Le [2017] also present a review of 59 conversational agents summarizing technologies, language tricks, special features and evaluation methods used in dialogue systems. The classifications hereby proposed are neither unique nor exhaustive.

Rule-based approaches have been the state-of-the-art for Loebner Prizes until the date of writing through the use of engines such as AIML or ChatScript. Despite this fact, research is converging into the use of ontologies [Bradeško and Mladenić, 2012; Abdul-Kader and Woods, 2015; al Rifaie, 2017]. Nevertheless, rule creation usually depends on the skill of its developers and one of its drawbacks lies in handling unexpected inputs. A downside of retrieval-based intent identification is in the lower quality responses due to noise from data sources [Higashinaka et al., 2014]. This study focuses on the case of KINO, a chatbot that uses natural language processed input, rule-based intent identification and retrieval-based output generation.

[1]STCs are a special case of the dialogue problem where the conversation has one round constituted of a two short-texts: input from the user; and the answer of the machine [Shang et al., 2015].

Chapter 3

KINO

KINO is a rule-based closed-domain chatbot that answers questions about movies in Brazilian Portuguese using pre-defined answers (retrieval-based). This domain was selected due to its usefulness for users who search for information about movies online and the availability of a free API to view information about movies in Portuguese. The API used was The Movie Database (TMDb) [TMDb, 2018].

This study was developed in two phases:

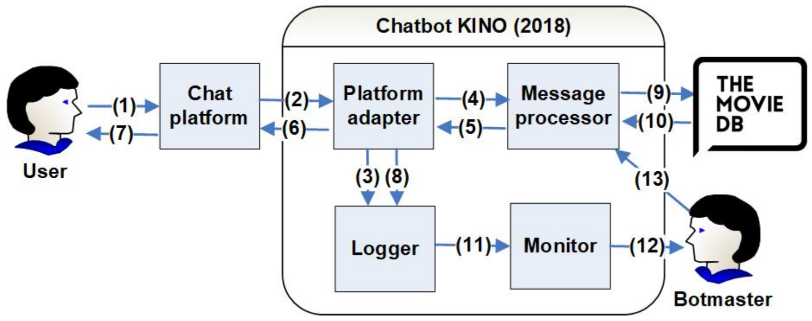
- Implementation phase: simultaneous development of the chatbot architecture (Section 3.1) and creation of the knowledge base (Section 3.2). The author, experienced in creating rules for chatbots as a software engineer, devoted about 20 hours a week - September 27, 2017 to January 28, 2018 - to accomplish this work.

- Validation phase: an experiment validated the chatbot and proposed evaluation and monitoring process (Section 3.3) - January 29 to February 8, 2018.

3.1 Architecture overview

The proposed architecture for KINO's conversational system is shown in Figure 3.1. The user interacts with a chat platform (Slack or Facebook Messenger) by sending requests in the form of messages and receiving its corresponding responses. The architecture has four main modules: platform adapter, message processor, logger, and monitor. Moreover, an external source of data was used to extend the knowledge base which in this case is TMDb.

Figure 3.1. KINO's operational architecture.



(1) Send request	(8) Log message response
(2) Listen to request	(9) Query info
(3) Log message request	(10) Return query result
(4) Process request	(11) Generate dashboard
(5) Generate response	(12) Evaluate behavior
(6) Deliver response	(13) Change behavior
(7) Receive response	

3.1.1 Platform adapter module

This module is responsible for listening to request events from a selected chat platform and triggering back response events. The types and forms of events are specific to the chat platform, but the text message is the central event in the communication. Two platform adapters were developed to handle those events: Slack adapter and Facebook Messenger adapter.

The **Slack adapter** handles private messages by the users for simple text messages. It also detects user mentions on channels, and is capable of answering back. This adapter only uses a subset of Slack's Real Time Messaging (RTM) API features [Slack, 2018], that also includes threads, attachments in messages, reactions/emojis and handlers for other events.

Minor Challenge 1 *Encouraging users to have conversation sessions*

Slack needs the registration of users to be able to chat with the bot. This can be a barrier during the development, which requires chat sessions for rule creation (further explained on Section 3.2), and also to production usage of the bot.

Solution to Minor Challenge 1 *Using Facebook Messenger*

Due to the lack of volunteers and the resistance of new users to register in Slack, an adapter was made for Facebook Messenger. Facebook announced Messenger had 1.3 billion users monthly by September 2017 [Facebook, 2017].

The **Facebook Messenger adapter** handles a subset of Facebook Messenger API features [Facebook, 2018]: text, audio, image, attachments, postback input messages. Postbacks are user events triggered by getting started buttons, persistent menu or custom postback buttons. The adapter also has custom configuration and responses for getting started and help postbacks.

Compared to Slack, Facebook requires additional effort in building a bot as it requires the creation of a HTTPS webhook with a valid certificate. In order to make the chatbot public on Facebook, there is a need for an App Review (which is not required for Slack). The requirements for approval include writing a Privacy Policy, drawing the app icon and providing conversation test cases.

As Facebook API also provides basic user information, it is persisted on a database for use in the conversation. This persistence provides the ability to remember the name of the users at anytime within the conversations.

Minor Challenge 2 *Specific requirements for size and timing of the answer messages*

Facebook Message Platform has a maximum length of message in characters and also time requirements to answer a message after it has been received. These rules differ from Slack and other platforms like Telegram or Kik.

Solution to Minor Challenge 2 *Implementing specific API logic for each adapter*

All relevant requirements for the expected bot operation and behavior must be implemented on each platform adapter, in order to have the bot interact simultaneously in multiple platforms.

3.1.2 Message processor module

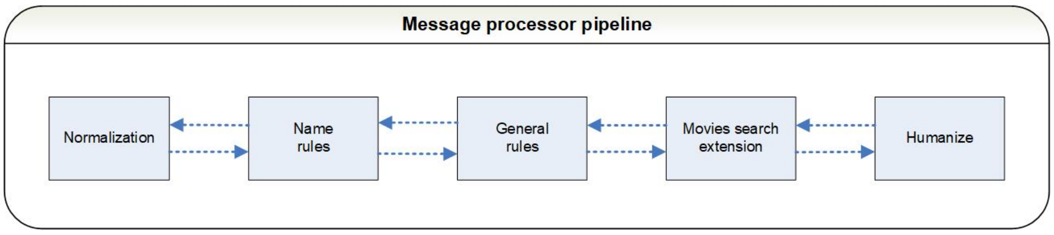
The message processor module (Figure 3.2) is a pipeline composed of five main components: normalization, name rules, general rules, movies search extension and humanize. In this pipeline, the user's input message flows from the initial components to the next if necessary. The first three components were written using Rivescript [Petherbridge, 2018a].

Rivescript is a rule engine in which the author already had previous experience. It has some features that are not present in AIML such as environment variables, object macros and topic inheritance [Petherbridge, 2018b]. Object macros are the ability to call functions in other programming languages and were extensively used in KINO to make the message flow through its components.

Minor Challenge 3 *Handling exceptions in message processing*

Computer software is prone to fail due to unexpected conditions (exceptions) like memory and input errors. If any component in the message processing fails, chatbot users still expect an answer from the bot.

Figure 3.2. KINO's message processor pipeline.



Solution to Minor Challenge 3 *Using user-friendly error messages*

Exception handling can partially solve this problem by programming predefined answers, considering the bot fails, such as: "Sorry I am not in good mood right now. Call me back later!". A message processor with multiple components can send different messages to identify the point of failure. For example, a two component processor could reply: "My head aches a lot in the back of my brain, can't answer right now." or "My head aches right under my eyes. Please wait until I get better." depending on which component the problem lies.

Major Challenge 1 *Intractable number of ways to write the same intent in a message*

In a rule-based approach, rules must exist for every possible way of expressing some intent.

Solution to Major Challenge 1 *Using text normalization*

Text normalization is a set of NLP techniques that can be used to reduce noise introduced from word contractions, acronyms, shortened versions of words, slangs, misspelling, omission, wordplay, censorship avoidance or in emoticons [Bertaglia, 2017]. One of the most simple forms of normalization is a list of substitutions. Rivescript has a *"! sub"* command that changes the first term into the second in the list. An example substitutions file to illustrate each of those categories could be:

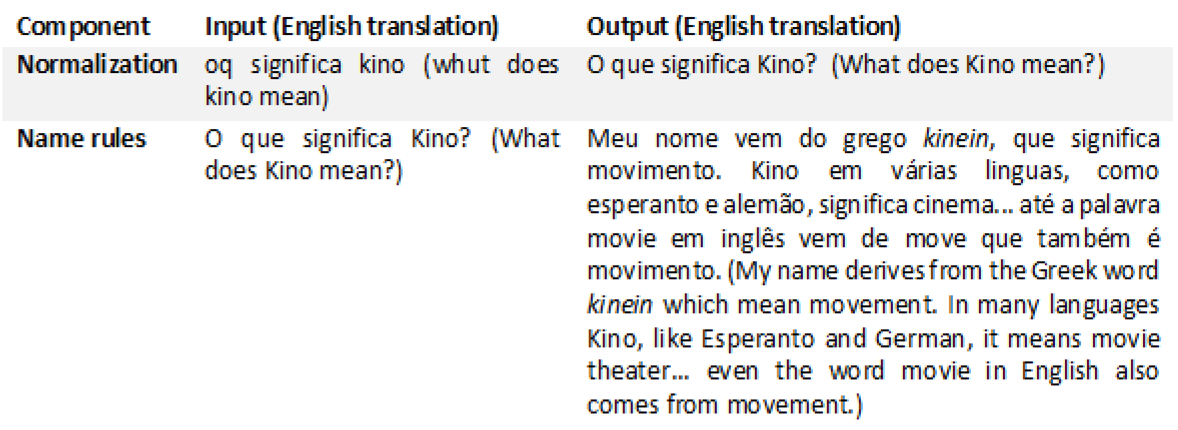
```
// contration example
! sub i'm = i am
// acronym example
! sub NY = New York
// shortened word example
! sub u = you
// slang example
! sub veg = vegetable
```

```
// misspeling example
! sub begining = beginning
// wordplay example
! sub booorring = boring
// censorship avoidance example
! sub h4te = hate
// emoticon example
! sub XD = laughing
```

Suppose two users entered the messages: "Wut is the director of LOTR?" and "what's the diretor of lord of the rings". If a normalizer transforms both into a canonical form "What is the director of Lord Of The Rings?", just one rule could be used to match the inputs. The problem is minimized with this solution by a reduction of the total number of rules needed to write the same intention.

The **normalization component** receives the message input and returns the normalized output. It has a list of Rivescript substitutions created manually from user logs (see Section 3.2 for details).

Table 3.1. Message processor with an input about the name example.



Component	Input (English translation)	Output (English translation)
Normalization	oq significa kino (whut does kino mean)	O que significa Kino? (What does Kino mean?)
Name rules	O que significa Kino? (What does Kino mean?)	Meu nome vem do grego *kinein*, que significa movimento. Kino em várias linguas, como esperanto e alemão, significa cinema... até a palavra movie em inglês vem de move que também é movimento. (My name derives from the Greek word *kinein* which mean movement. In many languages Kino, like Esperanto and German, it means movie theater... even the word movie in English also comes from movement.)

Translated by the author.

Minor Challenge 4 *Choosing the name of the bot*

The bot name was OSCAR at the beginning of the training phase. As in the movies the Oscar is an annual award to movies and their cast, this name choice for the bot had some side-effects identified. Firstly, the name of the bot created association to the award in a way users were asking questions about the award instead of asking the information the bot said he knew. Secondly, some input messages were ambiguous. The message "oscar" could mean the name of the bot or the information the user wanted to know. This ambiguity created a challenge for intent identification in the user's message. As a consequence of these side-effects, some dialog sessions were frustrated leading to low bot understanding (assertiveness - see Section 3.1.3) and user engagement.

Solution to Minor Challenge 4 *Changing the bot name*

The name was changed to KINO to minimize these effects, a neutral word that is not in Portuguese, neither associated to movies nor ambiguous to any information related to the bot.

Minor Challenge 5 *Identifying the position of the bot name in a message*

In some cases, users can insert the name of the bot in a message in any position eg.: "KINO, who is the director of Matrix?", "who is the director of Matrix, KINO?" or "can you tell me, KINO, who the director of Matrix is?". Rules would need to optionally have the name of the bot in any position for it to match all these cases, for positional rule engines, such as Rivescript. If the bot is called by a list of names or nicknames this problem increases.

Solution to Minor Challenge 5 *Breaking the rule sets into separated components*

As a solution to this problem, the name rule set was separated from general rules, creating a new component in the pipeline that filters the name. This additional layer enables the bot name to be used in any position multiple times. The drawbacks of this solution is the increased complexity and performance overhead by adding another layer.

The **name rules component** matches the normalized message with a set of rules about the bot name (example on Table 3.1). If no rule is matched, this component extracts all occurrences of the bot name from the message and sends it to the general rules component.

The **general rules component** processes all other input messages that do not regard the bot name. The rule sets were logically divided into:

- Basic personality: introductions, farewell, opinion, help or other general chatting phrases such as "hi", "how are you?", "what do you know?", "goodbye" or "do you have a girlfriend?".
- Feedback rules: inputs regarding the user's positive or negative opinion about the bot answers such as "thank you", "i like it", "this answer is wrong" or "I didn't like your answer".

- Domain specific rules (questions about movies): questions about movie information such as "who is the director of Star Wars?", "who is the producer of Fargo?" or "when is Thor 3 going to be released?".
- Catch-all rule: when no other rules apply, this rule catches the input. This rule is used as a source of measure of how much current rules are sufficient to recognize the intents.

A Rivescript example of each rule set is provided in Appendix A. The features that differentiate KINO from a pure question-answering dialogue system are the basic personality and context-awareness (as discussed in Chapter 1). The basic personality gives the ability to respond to some of the most common chat interactions. Context-awareness is provided by Rivescript through the use of conversation history, topics, and variables extensively used in feedback rules.

Major Challenge 2 *Handling multiple intents in one message*

Users' inputs are not restricted to one intent in the same input. It is possible to have two or more intents, eg.: "hello, what is the length of Avatar?" (two intents: "hello" and "length of Avatar") and "ok. very good! and what is the story of The Shape of Water?" (three intents: "ok", "very good" and "story of The Shape of Water"). The input identification method needs to handle those situations to be able to answer properly.

Solution to Major Challenge 2 *Using rules partially matching the input messages*

KINO uses the Rivescript feature of rule *redirections* by partially matching the input to the rules. Whenever some part of the input has a rule match, the answer is added to the output followed by the answer to the rest of the input. This way, the answer to "hello, what is the length of Avatar?" is the same answer to "hello" followed by the answer to "what is the length of Avatar?". The creation of rules to enable multiple intents requires additional skill of the rule developer to predict what rules are followed by other rules.

Major Challenge 3 *Removing conflict in rules*

Suppose a bot has a rule set that only answers to the inputs (Rivescript *"+"* command) "Good morning" and the feedback message "Good *", expecting a follow-up intent as in "Good. And who directed Titanic?", it also outputs (Rivescript *"-"* command) "I did not understand you." for everything else:

```
+ good *
- Thank you.

+ good morning
- Morning!

+ *
- I did not understand you.
```

The intents can be mistakenly classified as a feedback followed by "morning" in a rule conflict when the rule engine receives the message "good morning". The resulting answer is "Thank you. I did not understand you." in this case, although the expected answer is "Morning!". As the bot knowledge base grows, this kind of conflicts is more likely to occur.

Solution to Major Challenge 3 *Changing and retesting rules*

One downside of rule-based approaches is fixing conflicting rules. Manual changing and testing both rules combined with an understanding of priority of the rule triggers (input patterns) can be done, but as the number of rules escalate these conflicts demand more time to fix. Algorithms can be made to automate these tasks with a list of inputs mapped with the expected outputs. These automations were not used in this work.

The **movies search extension component** retrieves information about movies using TMDb API and generates the response structure. This component uses a strategy that extends the bot knowledge with rules by searching online for answers. If dates, years, numbers and lists are present in the answer, humanize functions are called.

Minor Challenge 6 *TMDb API rate limit*

A problem derived from using external APIs are restrictions to the rate of requests. TMDb's limit was four requests per second. This limit impacts simultaneous conversations demanding movie information and automatic testing of answers.

Solution to Minor Challenge 6 *Wrapping API requests into existing rate limit libraries*

Request wrappers can be used to wait for the required time if this limit is reached. There are open-source free libraries available with this functionality such as *ratelimit* <https://github.com/tomasbasham/ratelimit> and *requests-respectful* <https://github.com/SerpentAI/requests-respectful>.

Moreover, it is possible to mock responses when doing automatic testing (see Solution to Major Challenge 3) with API calls. Another way to handle this is to let the component fail and use a predefined answer (see Solution to Minor Challenge 3).

The **humanize component** compiles a selection of utility functions that converts programming language objects (such as dates, lists and numbers) into human readable output. An example of a list of directors retrieved as the format [Lana Wachowski, Lilly Wachowski] to "Lana Wachowski and Lilly Wachowski". These functions are used to expedite the generation of responses by the other components.

Major Challenge 4 *Lack of Brazilian Portuguese support in open-source software*

Although there are several software libraries for natural language processing and packages to deal with the problems arising from the development of chatbots, several of them need to be setup or have additional steps to work using Brazilian Portuguese. Moreover, some algorithms simply do not support the language or do not function properly.

The open-source humanize[1] component is an example of software that has Portuguese translation, however, it contains mistranslations and displays the data outside the formatting used in Brazil. Besides, Rivescript does not use the encoding of characters with accents and cedilla used in Portuguese (UTF-8) by default. Likewise, the TMDb API searches for movie names and other TMDb information in native English.

Solution to Major Challenge 4 *Implementing the Brazilian Portuguese version*

The humanize component code has been rewritten and adapted to Brazilian Portuguese and the modified version for Brazilian Portuguese is publicly available for use in: <https://github.com/staticdev/humanizer-portugues>. Additionally, all components used in this proposed architecture were configured to use the proper encoding, language and localization (Brazil).

A complete example of the message processor pipeline can be seen in Table 3.2. The messages are transformed and redirected to all components proposed until the final output.

[1] Available on: https://github.com/jmoiron/humanize

Table 3.2. Message processor with all components example.

Component	Input (English translation)	Output (English translation)
Normalization	vc sabe a data de lancamento do filme 50 tons de liberdade kino (do u know the release date to 50 shades freed kino)	Você sabe a data de lançamento do filme Cinquenta Tons de Liberdade, Kino? (Do you know the release date to Fifty Shades Freed, Kino?)
Name rules	Você sabe a data de lançamento do filme Cinquenta Tons de Liberdade, Kino? (Do you know the release date to Fifty Shades Freed, Kino?)	Você sabe a data de lançamento do filme Cinquenta Tons de Liberdade? (Do you know the release date to Fifty Shades Freed?)
General rules	Você sabe a data de lançamento do filme Cinquenta Tons de Liberdade? (Do you know the release date to Fifty Shades Freed?)	movie-search{[]}Cinquenta Tons de Liberdade, release-date{[]} (movie-search{[]}Fifty Shades Freed, release-date{[]})
Movies search extension	movie-search{[]}Cinquenta Tons de Liberdade, release-date{[]} (movie-search{[]}Fifty Shades Freed, release-date{[]})	O filme Cinquenta Tons de Liberdade (2018) release-date2018-02-07. (The movie filme Fifty Shades Freed (2018) release-date2018-02-07.)
Humanize	O filme Cinquenta Tons de Liberdade (2018) release-date2018-02-07. (The movie filme Fifty Shades Freed (2018) release-date2018-02-07.)	O filme Cinquenta Tons de Liberdade (2018) foi lançado hoje. (The movie filme Fifty Shades Freed (2018) was released today.)

Translated by the author.

3.1.3 Logger module

The logger module is responsible for storing conversation as utterances. Each utterance has its input, output, date, time, other information about the message processing and evaluation metrics. As a bot architecture evolves with modules and components, the information needed to understand the bot behavior also changes.

This fact can be illustrated with the incorporation of the normalization component before the message is sent to the rule sets. In order to learn how an input had matched a rule, it was necessary to know, in addition to the incoming message, the normalized message. In this way, new features/fields are added or changed in the logs.

Times and dates were also used for diagnosing long response times in message processing and delivery. When recorded through each step of the pipeline, it is possible to identify which component is consuming most of the time in the process.

Other features of the logging module are metrics, generated for evaluation and monitoring.

Minor Challenge 7 *Selecting metrics for chatbot evaluation*

When trying to assess the quality of a chatbot, dozens of different quality attributes are proposed in the literature and each implementation prioritizes them differently [Radziwill and Benton, 2017]. Each attribute can have some proposed metrics and each metric has a cost of defining, computing, storing and evaluating. This poses a problem for chatbot benchmarks in developments with limited resources of work and time like this study.

Solution to Minor Challenge 7 *Assertiveness and feedback metrics*

In an attempt to make a simple approach for analysis, two metrics were proposed in this study: assertiveness and feedback. Assertiveness was the name given to the measure of how much the bot has a response in its knowledge base for the incoming inputs. Feedback a metric related to the reaction of the users to bot outputs. The objective in selecting these metrics was to capture minimal, general and essential aspects to be able to compare and maintain chatbots, to the detriment of a more complete, and consequently complex, analysis for more robust projects.

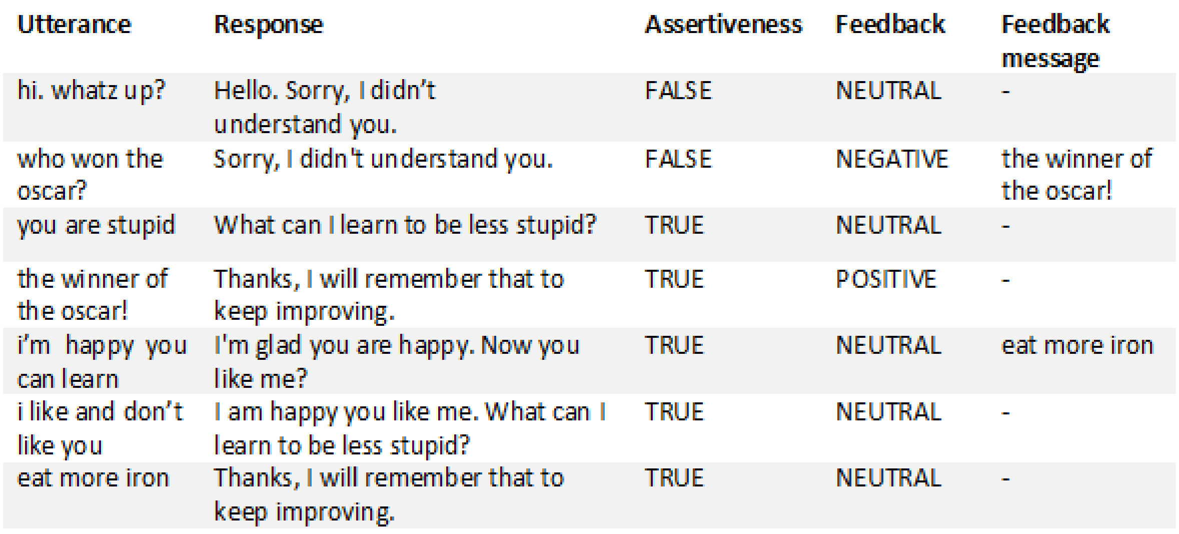
The way to compute assertiveness of KINO is through the catch-all rule. When an input matches this rule, it means the bot does not have in its rule sets the knowledge to answer completely (in case of multiple intents). The bot is assertive when it does not match the catch-all rule, and is not assertive otherwise. Table 3.3 shows an example in which the bot answers "Sorry, I didn't understand you" every time an intent does not have a response. This causes the assertiveness to be considered FALSE (not assertive), and for all other cases it is considered TRUE (or assertive). It is possible to apply the same idea for non rule-based bots adapting to the specifics of the input intent identification approach.

The feedback metric complements the bot evaluation with the user's perspective. When an input is received, there is a set of rules (see Section 3.1.2) that identifies positive and negative feedback inputs. These rules update the feedback metric of the last message of the bot with the values POSITIVE, NEGATIVE or NEUTRAL. If no feedback rule is applied, the feedback is considered NEUTRAL by default.

In the example (Table 3.3), the input "you are stupid" after bots output "Sorry, I didn't understand you.", caused the feedback field of the utterance right before having the NEGATIVE value. The same occurs when a rule matches a positive feedback rule like in "i'm happy you can learn". As for the NEUTRAL value, rules must occur simultaneously for POSITIVE and NEGATIVE value such as "i like (POSITIVE) and don't like you (NEGATIVE)".

Along with these metrics, there is another feature logged is the feedback message. This field value is the input message of the user when the feedback was given. KINO's

Table 3.3. Fictional example dialogue with corresponding logs of evaluation fields.



Utterance	Response	Assertiveness	Feedback	Feedback message
hi. whatz up?	Hello. Sorry, I didn't understand you.	FALSE	NEUTRAL	-
who won the oscar?	Sorry, I didn't understand you.	FALSE	NEGATIVE	the winner of the oscar!
you are stupid	What can I learn to be less stupid?	TRUE	NEUTRAL	-
the winner of the oscar!	Thanks, I will remember that to keep improving.	TRUE	POSITIVE	-
i'm happy you can learn	I'm glad you are happy. Now you like me?	TRUE	NEUTRAL	eat more iron
i like and don't like you	I am happy you like me. What can I learn to be less stupid?	TRUE	NEUTRAL	-
eat more iron	Thanks, I will remember that to keep improving.	TRUE	NEUTRAL	-

behavior is to log feedback messages only when negative feedback rules matched as in the example of the NEGATIVE and one NEUTRAL value (Table 3.3). It can also be possible to ask the user why it made a positive comment and log the information in the corresponding utterance.

Minor Challenge 8 *Visualizing log data*

The log data can grow in number of utterances as the number of users scales, making the task of reading dialogues and the fields of interest time-consuming.

Solution to Minor Challenge 8 *Creating a monitoring module*

Indicators and charts are ways to aggregate data for specific objectives and could be customized for on-demand visualization or even real-time monitoring. Both tools were used to mitigate this problem in the monitoring module.

3.1.4 Monitoring module

It is possible to make dashboards to aggregate and visualize the data for analysis and decision making, having all information stored as logs. Indicators can be used to aggregate metrics by dialogue sessions, users, dates, in a bot before and after rule changes, and so on. In this study the assertiveness metrics were aggregated in an indicator A as follows:

$$A = \frac{1}{n}\sum_{i=1}^{n} a_i * 100 \tag{3.1}$$

This indicator was constructed considering the value a_i of each utterance with a value 0 for FALSE; or 1 for TRUE for n utterances. The average of these values was made with the result multiplied by 100. This indicator value ranges from 0 to 100. In the Table 3.3 example, five out of the seven utterances had a value TRUE, obtaining an assertiveness indicator of 72.43.

$$F = \frac{1}{n}\sum_{i=1}^{n} f_i * 100 \tag{3.2}$$

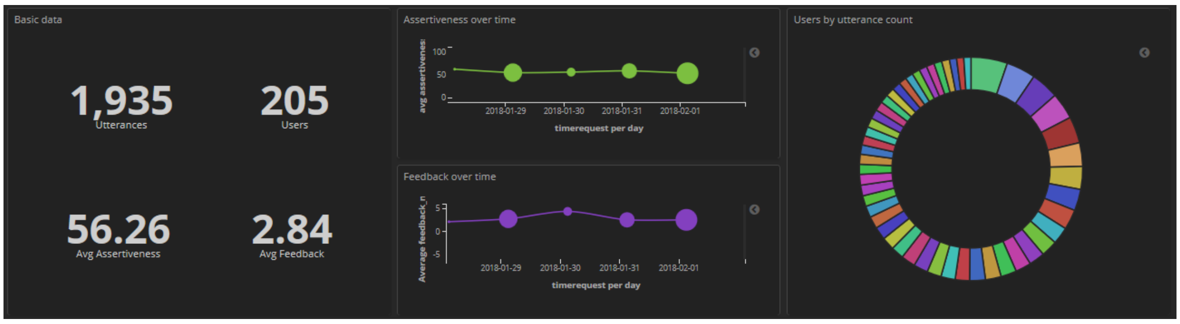
As for the feedback indicator F, the values considered were -1 for NEGATIVE; 0 for NEUTRAL; and 1 for POSITIVE, in each value f_i of the n utterances. Values were also averaged and multiplied by 100. When calculating the value F of the Table 3.3, five utterances had the value NEUTRAL, one POSITIVE and one NEGATIVE, making the feedback indicator score of 0.

Message counts and indicators can be shown in charts, combing fields in order to enable graphical visualization of the current behavior of the bot. Time-series charts may have indicators through time, giving the perception of evolution as well as allowing trend line fits and projections.

Four panels were designed to show interactive information in a given time range - 29-Jan and 2-Feb 2018 - in the example dashboard (Figure 3.3). The panel on the left shows basic data with counting of utterances and unique users, along with the assertiveness and feedback indicator of all utterances. The panels in the middle show a line chart with a day-by-day evolution of the assertiveness and feedback indicators with dot sizes representing the utterance count of the day. The panel on the right shows a pie chart split by the users sorted by their utterance count.

The information displayed on dashboards depends on the objectives of analysis and the information that is relevant to the context of the chatbot. In the experiment of this study that follows (see Chapter 4), three normalization methods were used and a dashboard was built to show the indicators of each of them. In other chatbot applications, it may require creating multiple dashboards for different target audience or roles in the process (eg. code developers, rule maintainers, and managers).

Figure 3.3. KINO's example dashboard between 29-Jan and 2-Feb 2018.



3.2 Knowledge base creation

25 non-paid volunteer users were recruited, 14 males and 11 females from 21 to 45 years old. Each participant was instructed to chat freely with the current bot prototype about movies. They were assisted by the bot creator to help with doubts or problems during the conversations. Every dialogue session was logged, and up to 3 of them were selected to be used in the knowledge base creation.

Each time a dialogue session from a different user was used in an attempt to improve assertiveness and feedback metrics of the latest version of the knowledge base. They were reviewed, utterance-by-utterance, for changes in the rule set (rule triggers, answers and feedback rules) and in the substitutions list (normalization) e feedback rules. During this process, the sessions used to identify behaviors to be corrected were also used for testing the changes in the chatbot. Changes in the code and in the rules were made until all problematic utterances had new behaviors to be retested in the next sessions.

Minor Challenge 9 *Defining an initial scope for the bot answers*

An initial scope of answers needed to be set, without prior knowledge of the user's most asked questions about movies and limited time for the development of rules and integration to the TMDb API.

Solution to Minor Challenge 9 *Choosing an initial question and incrementally adding other questions*

The only question about movies KINO was programmed to answer was the directors of a movie during the first dialogue sessions. Then, other recurring questions were implemented simultaneously with the rule corrections: **known movies**, **producers**, **release date**, **release year**, **overview**, **vote average** and **runtime** of a given movie.

These implemented questions are a subset of the information in TMDb and were chosen as a proof of concept.

Major Challenge 5 *Defining a strategy for rule corrections*

Consider the following rule set with just one rule with an input pattern and a corresponding output:

```
+ who directed the movie click
- The director of Click is Frank Coraci.
```

If a new input "what is the director of Click called?" has the same answer, the rule in the rule set should be corrected to match both inputs. Given a rule set and an input that does not have a match, different strategies can be used to make the rule set match this input.

Solution to Major Challenge 5 *Trying different strategies*

Some of the possible strategies identified: Strategy 1) Create one rule per input redirecting (@ command) all of them to a common answer. The resulting rule set would be:

```
+ who directed the movie click
- The director of Click is Frank Coraci.

+ what is the director of click called
@ who directed the movie click
```

One advantage of this solution is minimizing rule conflicts (see Major Challenge 3). Since every trigger is a unique input, changes in the behavior are straightforward: a) for a new output, create a new rule with the input as the trigger; b) for an existing output and a new input, create a new rule and redirecting the output to another rule with this output; c) for an existing input with a wrong answer, find the trigger and change the corresponding output.

The problem with this approach is that the bot model complexity is the size of the sample, causing a known problem of overfitting and not having rules the generalize to different inputs. Also, with a rule for every possible input, the rule sets could grow exponentially, causing potential performance and memory issues.

Strategy 2) Create one rule per output. The resulting rule set would be:

```
+ (who directed|what is the director) [of] [the movie] click
   [called]
- The director of Click is Frank Coraci.
```

In this example, the triggers have optional terms in brackets and alternative terms in parenthesis such as (who directed|what is the director) meaning if the terms "who directed" or "what is the director" are present in the input, this rule should match. The advantage of this approach is the generalization, all permutations of the rule will match the output. The problem with that is creating inductive bias, matching combinations of terms that does not make sense, for instance, "who directed of click called?".

A heuristic Strategy 3 was used in KINO in an attempt to balance the advantages and drawbacks of the Strategy 1 and 2.

Strategy 3) Create one rule per output, unless it creates nonsense combinations of terms (see the previous example). In this case, the rule is split by creating a new rule and redirecting the answer to the original. If a new rule is created, all rules that has the same output are reevaluated for expanding terms generalization.

```
+ who directed [the movie] click
- The director of Click is Frank Coraci.

+ what is the director of [the movie] click called
@ who directed the movie click
```

In this example, the creation of one rule per output created invalid combinations. Then, this rule is split creating a new rule "what is the director of click called" redirecting to the answer of the original rule. After evaluating both rules, the term "the movie" before the movie name, could optionally exist in this new rule, so it is added to allow more variation of inputs.

The idea of this approach is to learn new variations of inputs and at the same time avoiding invalid combinations of terms. The advantage of this is minimizing rule conflict and also having some degree of generalization. One disadvantage is the added cost of evaluating a rule for invalid combinations and making another evaluation for variations after a split, making the rule maintenance more complex task.

Another way of simplifying the knowledge base creation process was avoiding a Rivescript resource called *topics*. It allows organizing sequences of messages about a subject, providing more context. However, there is an added cost to the conversational design in creating association flows from one input to another input.

Along with the rule sets, the list of substitutions for normalization (seen in Section 3.1.2) was also created during the dialogue sessions review.

Minor Challenge 10 *Handling misspelling with a substitution list*

From the logs used to create the substitutions list, the misspelling of words seemed to be the most difficult to summarize in a substitutions list. They occurred frequently and virtually any letters missing or incorrectly placed created misspelling. The total number of combinations of a word with one character added or removed for any possible word could make the approach taken unfeasible.

Solution to Minor Challenge 10 *Using different normalization approaches*

In Portuguese words have accents and cedilla, and chat users not always put those when writing. Therefore, a word like "duração" (Portuguese for duration) could be found written as "duracao", "duracão" or "duraçao". All this omissions could be added to the list, but to keep consistency with the heuristic for rule change (Strategy 3), this list was also minimal. Only substitutions that existed in session logs were added. "duracao" was the only occurrence of misspelling in the sessions log, so the only substitution added was:

```
! sub duracao = duração
```

In order to further explore options to a substitutions list, the experiment performed to validate KINO's development, evaluation, and monitoring also used other approaches. Two Brazilian Portuguese methods were used with the substitutions list as a baseline in the experiment. In the next section, these methods and the experiment are detailed.

3.3 Experimental setup

The validation experiment was prepared after the knowledge base creation, from 9 to 28 January 2018. During this period microservices were implemented (Section 3.3.1), the methods were configured and tested by the author with minor fixes in KINO's code and rules. The complete set of rules used in the experiment is available on <https://github.com/staticdev/kino-rivescript-experiment>. The message processor module was adapted to normalize randomly each input message with a one-third chance for each method (RiveNorm, UGCNormal and Enelvo).

3.3.1 Normalization methods

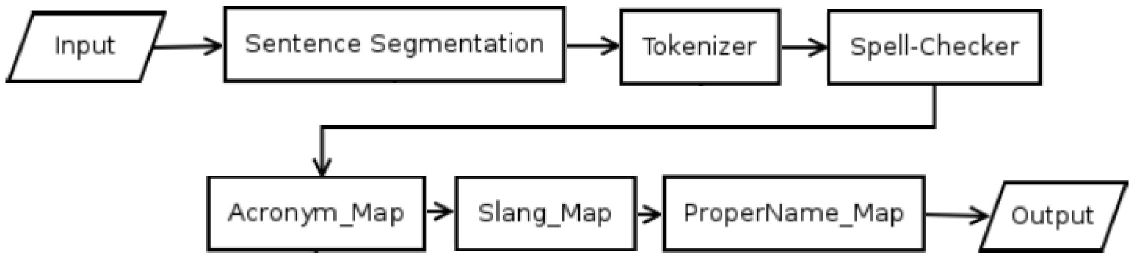
For the validation three methods were used for KINO's normalization:

- KINO's RiveNorm (Rivescript substitutions list shown in Section 3.2)
- UGCNormal [Duran et al., 2015]
- Enelvo [Bertaglia and Nunes, 2016]

Both methods were chosen for being open-source and for having the higher reported accuracy found for Portuguese Brazilian normalization. The TUB-UGC corpus, a collection of UGC from Twitter, UOL, and Buscape, was annotated and used by Bertaglia and Nunes [2016] to compare both methods. Enelvo has shown higher accuracy in six out of seven categories of noise for normalization.

UGCNormal stands for User-Generated Content Normalization and is a multistage method as shown in Figure 3.4. Its code is licensed under Creative Commons Attribution-NonCommercial 3.0 and is available on <https://github.com/avanco/UGCNormal>. The input text passes through the Sentence Segmentation that delimits each sentence, followed by the Tokenizer that breaks the sentences into tokens (usually words) and by the Spell-Checker that corrects spelling errors. Then, the Acronym_map leaves the acronyms in upper case, the Slang_map transforms contractions and slangs into formal language, and finally, the Propername_map outputs the normalized text with capitalized proper names [Avanço, 2015].

Figure 3.4. UGCNormal pipeline.

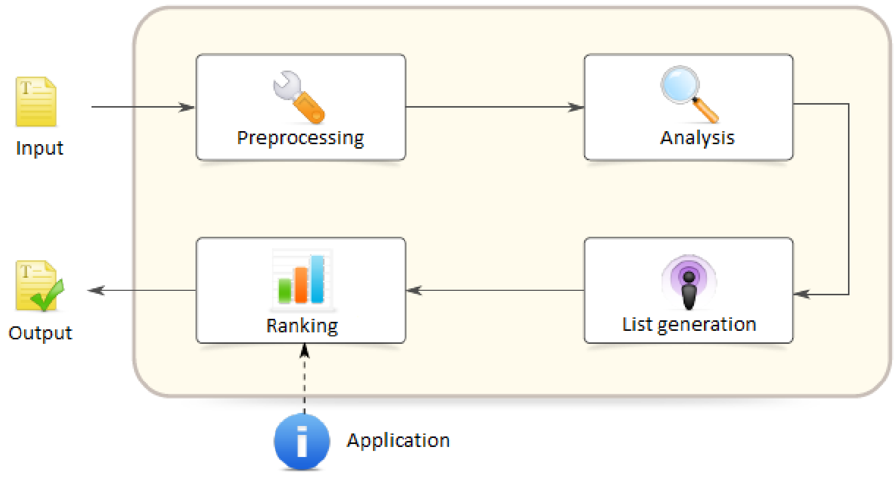


Source: Adapted from Duran et al. [2015]. Edited and translated by the author.

The Enelvo normalizer (Figure 3.5) is also multistage and based on User-Generated Content (UGC). Its code is available on

<https://github.com/tfcbertaglia/enelvo> under MIT License. The Preprocessing does sentence segmentation, tokenization, inserts punctuation and capitalization and also removes sequences of characters with more than three repetitions like in "booooring". The Analysis stage identifies the noisy words to be corrected. In the List generation, each word to be corrected are assigned with a list of candidates for substitution. The Ranking module is responsible for determining the suitable candidates, replace the noisy words with them and to output the final normalized text.

Figure 3.5. Enelvo pipeline.



Source: Adapted from Bertaglia [2017]. Edited and translated by the author.

Minor Challenge 11 *Adapting text normalization methods to chatbots*

The text normalization methods used were not designed for chatbots but to normalize text files. Each input message had to be written in a file for the normalization method to be called and then an output file was read after the processing. One problem that arose from this situation was the need to adapt these solutions due to normalization times in the order of magnitude of seconds.

Solution to Minor Challenge 11 *Configuring and using microservices*

In an effort to minimize the time two measures were taken. First, the authors of the methods were contacted for configuration of an interactive mode, in which the normalization was done as a function instead of using files. And second, containers with independently deployable normalization services (microservices) were developed to guarantee even with simultaneous users that the normalization functions did not have to load all its objects every time a method was called. These combined actions reduced the normalization times to under one second.

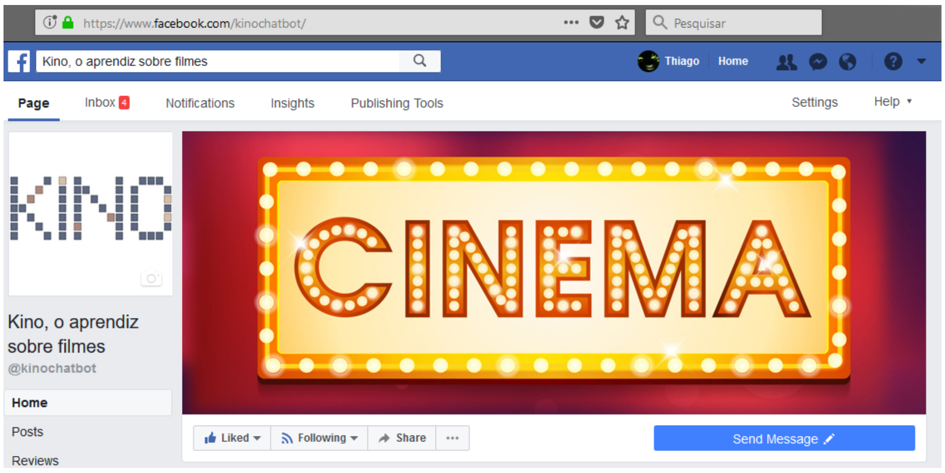
The normalization microservices developed for this experiment are open-source and licensed under MIT License:

- Kino-rivenorm-microservice available on <https://github.com/staticdev/kino-rivenorm-microservice>
- Ugcnormal-microservice available on <https://github.com/staticdev/ugcnormal-microservice>
- Enelvo-microservice available on <https://github.com/staticdev/enelvo-microservice>

3.3.2 Data gathering

All the conversations used Facebook Messenger and were gathered from January 29 to February 8. A Facebook page was created (Figure 3.6) with a visual identity to entice users. Users for the experiment were obtained through a sponsored post. The images and content of the page and post were made with the help of a social network analyst with a degree in graphic design.

Figure 3.6. KINO's Facebook page with a cover photo with "cinema" (movie theater in English) written. The page name is "Kino, o aprendiz sobre filmes" which means Kino, the apprentice about movies.



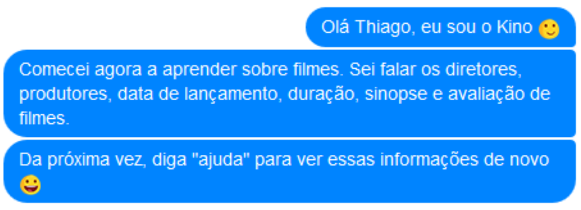
The post had an art (Figure 3.7) with the purpose of informing it was related to a robot and introducing an idea associated with movies. Along with the image, there was also a short text explaining which information KINO could answer and how to start a conversation with it. An ad campaign was created for the experiment in the period from January 31 to February 7, with a budget of 15 BRL (Brazilian Real) and a target audience of men and women, ages 21 - 45 who live in Minas Gerais - Brazil. The target audience parameters were chosen to have the same structure of the training users in an attempt to avoid bias from differences in gender, age or location. At the same time, the post was individually shared and word of mouth was shared with friends and acquaintances.

Figure 3.7. KINO's promoted post image.

Source: Made by Fernanda D'Alessandro Portilho.

In order to reinforce information about the chatbot function, an initial message was configured for the chat interface (Figure 3.8) along with a help menu. In the initial message, the chatbot introduces himself and informs what data about a movie he knows. Furthermore, a help option is given which also shows the movie data options on the menu.

Figure 3.8. KINO's initial message can be translated as: "Hi %Username, I'm Kino =). I've just started learning about movies. I know about directors, producers, release date, runtime, synopsis and average votes of movies. Next time, say "help" to see this information again.".



Olá Thiago, eu sou o Kino 🙂

Comecei agora a aprender sobre filmes. Sei falar os diretores, produtores, data de lançamento, duração, sinopse e avaliação de filmes.

Da próxima vez, diga "ajuda" para ver essas informações de novo 😀

Throughout the validation phase, the knowledge base with the chatbot rules have not changed. Likewise, the database and the source code have been left untouched. These were also ways of minimizing disturbances in the metrics and variables used in the analysis.

An experiment to evaluate KINO using different normalization methods was set up for testing the knowledge base using proposed evaluation and monitoring presented in this chapter. The next chapter presents the analysis and results in order to assess the feasibility of this study's proposal and determine the scope for improvement.

Chapter 4

Experimental Evaluation

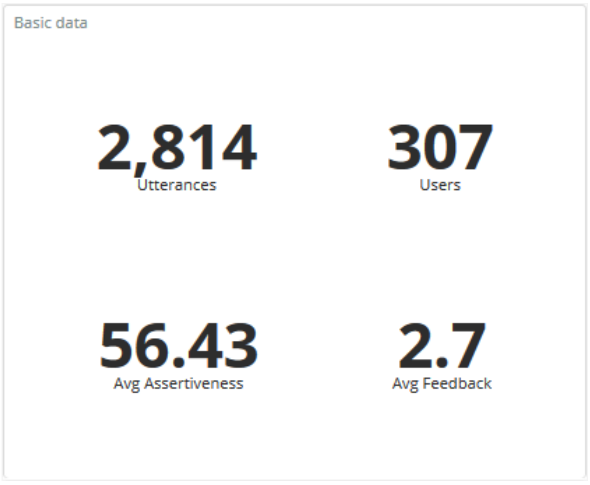
In this chapter, the experimental validation of KINO's development, evaluation and monitoring is described. Assertiveness and feedback metrics and indicators were used as the main reference for analysis. A dashboard was developed to individually monitor each method in terms of the number of total utterances, users, indicators of assertiveness and feedback. Along with this dashboard, the data was described in Section 4.1. Then, hypothesis tests were performed to evaluate differences in indicators between normalization methods in Section 4.2. A regression analysis was carried out to find words correlated with the assertiveness in Section 4.3. In Section 4.4 an analysis of the chatbot types of error was made through a manual classification in the collected dialogue messages. Finally, in Section 4.5, a brief review and discussion relate some of the contents presented.

4.1 Monitoring results

For the purpose of analysis, from all the data collected during the experiment period (Figure 4.1), the conversations of users who participated in the knowledge base creation and utterances that caused exceptions (see Minor Challenge 3) were excluded. Of the total of 2814 messages and 307 users, 30 messages with exceptions, 246 postback messages and 26 messages from the implementation phase users were removed, leading to 2512 analyzed messages from 287 users.

In the panel on the left of Figure 4.2 it is possible to see that for the data analyzed the assertiveness indicator is 51.71 and the feedback indicator is 2.71. These values may contain a bias arising from the manner in which the experiment was disclosed and can not be evaluated without a benchmark. In the middle panel indicators over time are shown, providing a basis for time comparison. As in this period the chatbot

Figure 4.1. Original messages visualization.


Basic data

2,814 Utterances

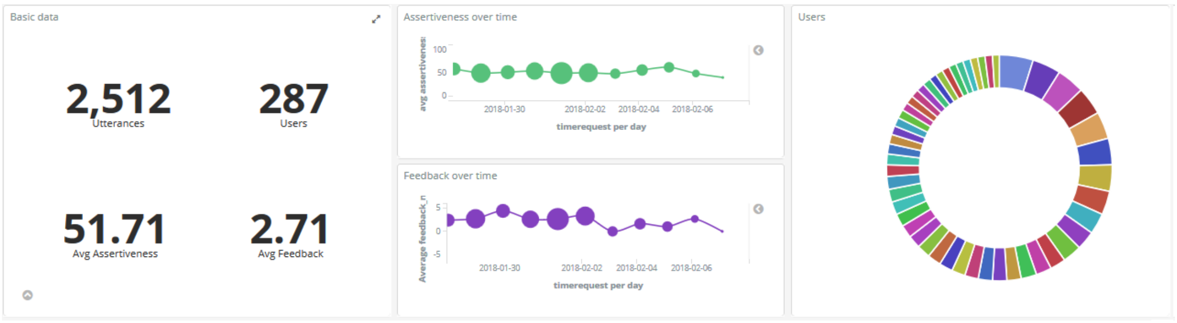
307 Users

56.43 Avg Assertiveness

2.7 Avg Feedback

was not changed, it was expected a constant behavior of assertiveness and feedback. The dot sizes in the time graphs show the volume of utterances and with this, it is possible to observe that the greatest variations of the indicators occurred on the days when there was less volume of messages. In the panel on the right, the 50 users with the most messages are shown. Observe that there is a variation between the number of messages from one user to another.

Figure 4.2. Dashboard for analyzed messages.


Basic data

2,512 Utterances

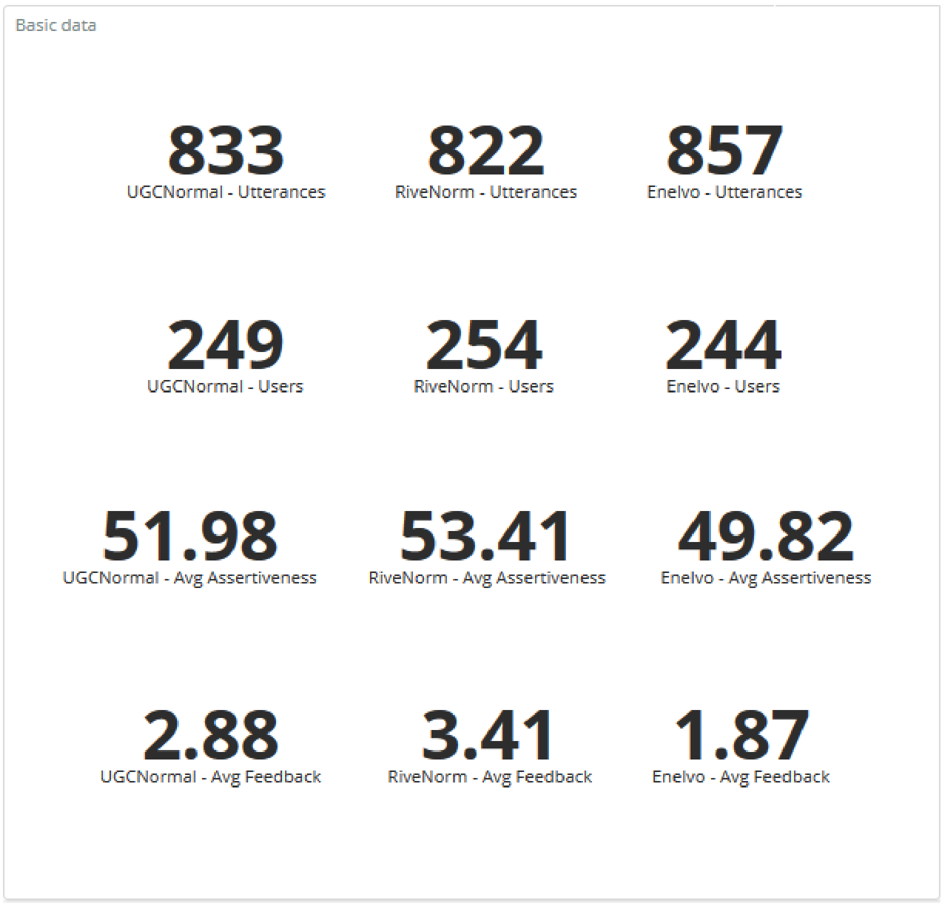
287 Users

51.71 Avg Assertiveness

2.71 Avg Feedback

Figure 4.3 shows the visualization for the methods. Approximately 837 utterances per user were expected, as this number corresponds to 1/3 of the total of 2512 messages. The method with the greatest variation of this value was Enelvo, with 20 more messages ($\approx$ 2%). In addition, this method had the lowest number of users and indicators of assertiveness and feedback in the period. On the other hand, the RiveNorm method obtained the lowest number of messages and the highest number of users, assertiveness, and feedback indicators. Also, the UGCNormal was in the middle in all the fields shown.

Figure 4.3. Normalization methods visualization.



Basic data

833
UGCNormal - Utterances

822
RiveNorm - Utterances

857
Enelvo - Utterances

249
UGCNormal - Users

254
RiveNorm - Users

244
Enelvo - Users

51.98
UGCNormal - Avg Assertiveness

53.41
RiveNorm - Avg Assertiveness

49.82
Enelvo - Avg Assertiveness

2.88
UGCNormal - Avg Feedback

3.41
RiveNorm - Avg Feedback

1.87
Enelvo - Avg Feedback

In a complementary way, Figure 4.4 shows the indicators of assertiveness and feedback throughout the days of the experiment. In both metrics, there was no dominance of a method every day and all of them had the highest indicator for at least one day.

During the experiment, 92 non-neutral feedback utterances were received ($\approx 3.7\%$ of the total) with 84 POSITIVE and 8 NEGATIVE feedback. Figure 4.5 shows the distribution of these feedbacks for each method: RiveNorm and UGCNormal had 30 feedbacks each and Enelvo had 24. The amount of NEGATIVE valued feedback samples in the experiment was lower than that recommended for subsequent statistical analyses [Cochran, 1954].

The low amount of non-neutral feedback has some possible causes. The frequency of messages giving feedback does not represent a high percentage of the total; the initial post, sponsored post, and chatbot responses did not instruct or incentivize the user to give feedback correctly, meaning there could be an experimental design failure; the feedback rules learned in the implementation phase were not enough to capture user

Figure 4.4. Assertiveness and feedback indicators over time.

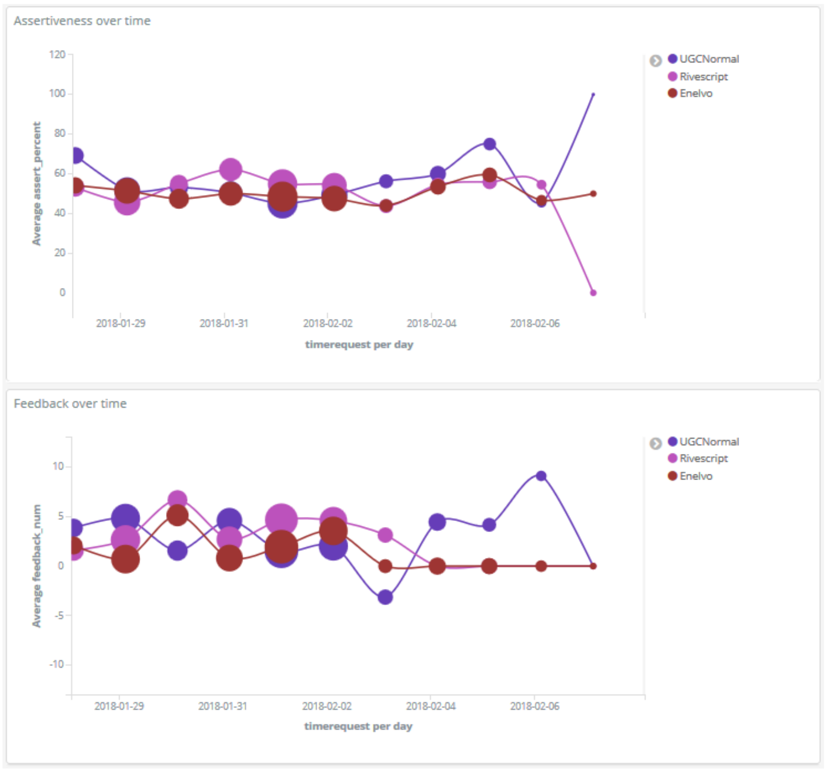
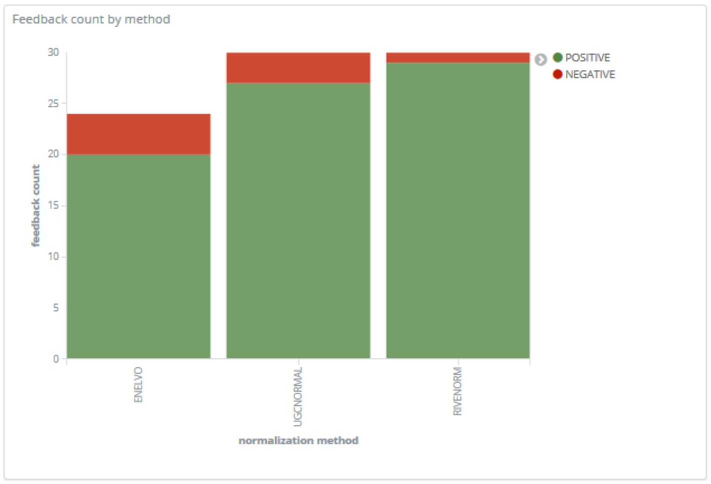


Figure 4.5. Non-neutral feedback count by method.



feedback or were not matching correctly the input messages.

4.2 Normalization methods comparison

In the attempt to use the described data to search for differences in indicators between normalization methods, statistical tests were performed. The assertiveness and feedback variables for an utterance were categorical with the labels {TRUE, FALSE} and {POSITIVE, NEUTRAL, NEGATIVE}, respectively. A statistical comparison among the three different normalization techniques was made considering both metrics had a Bernoulli distribution.

The comparison among the methods was made using the Pearson's Chi-squared Test of Independence with a significance level of $\alpha = 0.05$. A result is considered statistically significant when $p < \alpha$ [Agresti, 2007]. The null hypothesis was that the indicator values for all three methods were equal, and therefore the probability of a correct interpretation of the answer was independent of the normalization technique that had been used.

The assertiveness test was made by arranging a 3x2 contingency table (Table 4.1), with each line being the methods and the columns being the assertiveness values. With a resulting p-value of 0.33, the Chi-squared Test does not reject the null hypothesis. The usage of different methods did not have an statistical impact on the assertiveness of the chatbot.

Table 4.1. Assertiveness contingency table.

		Assertiveness		
		TRUE	FALSE	Total
Method	ENELVO	427	430	857
	RIVENORM	439	383	822
	UGCNORMAL	433	400	833
	Total	1299	1213	2512

As for the feedback, a 3x3 contingency table (Table 4.2) was made in an analogous way. This contingency table has 4 degrees of freedom $((3-1)*(3-1))$ and 3 out of 9 cells had less then 5 samples. In this case, the samples did not meet the minimum expectations recommendation for the test, so Fisher's Exact Test was used as an alternative [Cochran, 1954]. Fisher's Test had a p-value of 0.40, also not rejecting the null hypothesis. With this low size of samples for NEGATIVE values of feedback and the p-value higher than the significance level set, there is not enough statistical evidence of how the use of different normalization methods affected the user's feedback.

One hypothesis raised to have no statistical difference in assertiveness between the methods even with different complexities was that simpler methods like RiveNorm were hitting where the more complex like Enelvo and UGCNormal were erring and somehow compensating for more difficult normalizations. Another assumption would be that the correction of less common errors made by the more complex methods was negligible in the observed sample. Other analyses were carried out in an attempt to

Table 4.2. Feedback contingency table.

		Feedback			
		POSITIVE	NEUTRAL	NEGATIVE	Total
Method	ENELVO	20	833	4	857
	RIVENORM	29	792	1	822
	UGCNORMAL	27	803	3	833
	Total	76	2428	8	2512

further investigate the assertiveness behavior.

4.3 Words correlation to assertiveness

In order to verify if any words were particularly correlated with the assertiveness, a linear model with regularization was constructed (the Lasso regression) [Tibshirani, 1996]. The objective function of this algorithm is given in Equation 4.1:

$$Lasso = \min\left(\left(\sum_{\forall i \in \{1..n\}} y_i - \hat{y}_i\right)^2 + \lambda \sum_{\forall j \in \{1..p\}} |\beta_j|\right). \tag{4.1}$$

The first term is the Residual Sum of Squares (RSS) and the λ term is called the regularization term. The objective function is the same as that of linear regression, but a regularization term is added, which tends to shrink the β_j fit to the data. The regularization used by Lasso regression is also called l_1 regularization because the sum is taken over the modules of the parameters instead of the square. This has the effect that for a regularization parameter large enough the modules of any given parameter β_j will be shrunk to zero. In contrast, with $l_2 - norm$ regularization the parameters will be shrunk to small values, but not to zero. This makes the Lasso regression an interesting algorithm for interpretability when a large number of parameters is present [James et al., 2013].

In this case, a total of 1766 unique words were identified and encoded using one-hot-encoding before being used to train the model of the assertiveness as a function of the words present. One model was built for each regularization algorithm and another for the full dataset and the values of 10, 30 and 50 were tested for regularization parameter. When performing the regressions, the coefficients of the words with the greatest effect on the assertiveness were sorted in ascending order. The value of 30 words was chosen for allowing greater interpretability of the results.

The 10 words with the highest coefficient in all methods were colored for verifying words positively related to assertiveness (Table 4.3). The same colors were used in for a repeated word in each method to see how these words affected each one. At least 8 of them were present in each method indicating these words still had positive coefficients despite the difference in the methods.

Table 4.3. Lasso regression with 30 words sorted by assertiveness coefficient.

	ALL		RIVENORM		ENELVO		UGCNORMAL	
#	word	coeff	word	coeff	word	coeff	word	coeff
1	me	-0.170	filmes	-0.167	me	-0.245	ator	-0.165
2	ator	-0.149	elenco	-0.119	ator	-0.198	fale	-0.160
3	filmes	-0.108	me	-0.090	filmes	-0.044	me	-0.127
4	elenco	-0.065	fale	-0.053	tem	-0.006	filmes	-0.082
5	fale	-0.029	principal	-0.018	você	-0.001	mais	-0.065
6	é	0.003	o	-0.017	elenco	0.000	e	-0.051
7	nota	0.005	casa	-0.013	o	0.019	principal	-0.025
8	ano	0.021	sabe	0.007	do	0.023	saber	0.000
9	do	0.033	de	0.008	sabe	0.043	o	0.014
10	de	0.036	quem	0.053	de	0.044	dirigiu	0.031
11	bom	0.040	chame	0.068	é	0.055	sabe	0.032
12	não	0.048	do	0.069	olá	0.069	que	0.032
13	kino	0.053	lançado	0.074	muito	0.071	de	0.044
14	sabe	0.055	a	0.108	filme	0.081	lançado	0.057
15	a	0.070	ano	0.112	qual	0.096	kino	0.064
16	filme	0.077	lançamento	0.149	tchau	0.105	olá	0.071
17	lançado	0.101	bom	0.160	enredo	0.115	tudo	0.085
18	lançamento	0.123	historia	0.175	não	0.117	lançamento	0.102
19	qual	0.171	qual	0.198	kino	0.117	a	0.107
20	enredo	0.179	duração	0.221	nota	0.128	filme	0.136
21	olá	0.184	obrigada	0.242	lançado	0.170	enredo	0.136
22	duração	0.207	enredo	0.258	lançamento	0.170	qual	0.184
23	obrigado	0.290	avaliação	0.277	melhor	0.194	história	0.257
24	quando	0.296	sinopse	0.283	obrigada	0.262	sinopse	0.304
25	avaliação	0.318	olá	0.297	obrigado	0.283	avaliação	0.324
26	sinopse	0.321	história	0.308	oi	0.295	oi	0.335
27	obrigada	0.322	diretor	0.389	diretor	0.380	duração	0.355
28	história	0.327	quando	0.420	avaliação	0.387	diretor	0.403
29	oi	0.383	obrigado	0.448	sinopse	0.405	quando	0.407
30	diretor	0.387	oi	0.485	história	0.479	obrigada	0.431

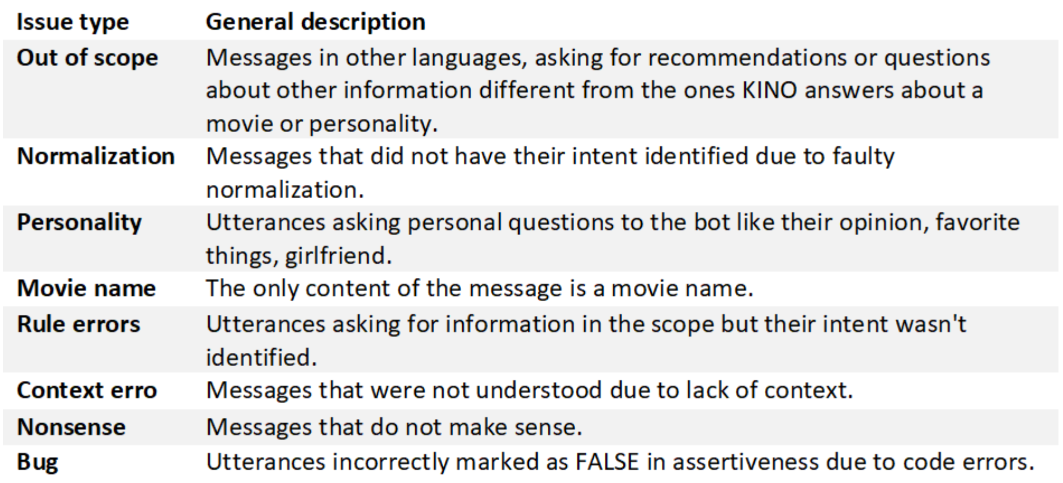
The words "diretor", "oi", "história", "obrigada", "sinopse", "avaliação", "quando", "obrigado", "duração" and "olá" (translations: "director", "hi", "overview", "thanks", "overview", "average vote", "when", "thanks", "runtime" and "hi") were associated with messages that the chatbot had matching rules in its knowledge base. From the 10 positive related words, 5 are related with information from the API (director, runtime, overview - twice and average vote), 2 were greetings, 2 were feedback messages (thanks) and 1 is a question word (when). These results indicate a correlation with the trained knowledge and positive assertiveness.

The same procedure was applied to the 5 words with a negative coefficient for the regression (also in Table 4.3). It is possible to identify 4 out these 5 words repeated in each method. The words "me", "ator", "filmes", "elenco" and "fale" (translations: "me", "actor", "movies", "cast" and "tell") were associated with messages that did not have a matching rule in the knowledge base. There was a total of 100 utterances with the word "me" with 81 of them having FALSE assertiveness, so Lasso could be used to incorporate words with the lowest assertiveness coefficients in an attempt to increase it. Although "actor" and "cast" were movies information that was not in the scope the knowledge base, the words "me", "movies" and "tell" did not give enough information about what types of issues caused the messages to have false assertiveness.

4.4 Issues related to low assertiveness

In an attempt to identify the type of issues in the chatbot assertiveness, a manual classification analysis was performed. The 1213 messages with FALSE value were read by a human agent to enumerate a list of issues as in Majumder et al. [2018]. From the complete list, an order of precedence was defined for classification of the messages in the Table 4.4.

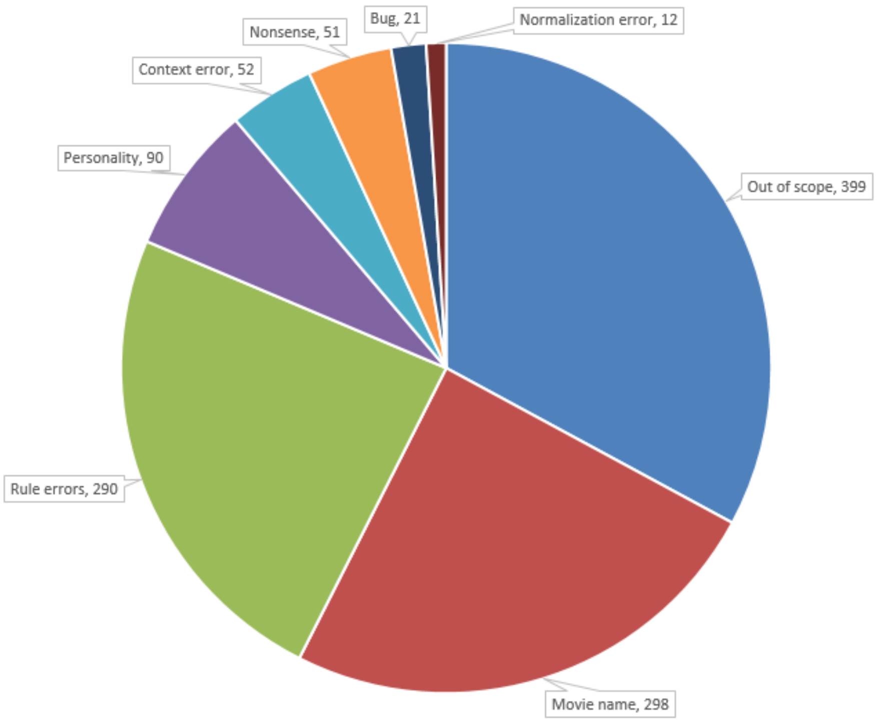
Table 4.4. FALSE assertiveness utterances types of issues sorted by precedence. The category "movie name" was created separately because of the high frequency in which they occurred in the utterances with only the name of a movie, but they could also have been counted as "rule errors".



Issue type	General description
Out of scope	Messages in other languages, asking for recommendations or questions about other information different from the ones KINO answers about a movie or personality.
Normalization	Messages that did not have their intent identified due to faulty normalization.
Personality	Utterances asking personal questions to the bot like their opinion, favorite things, girlfriend.
Movie name	The only content of the message is a movie name.
Rule errors	Utterances asking for information in the scope but their intent wasn't identified.
Context erro	Messages that were not understood due to lack of context.
Nonsense	Messages that do not make sense.
Bug	Utterances incorrectly marked as FALSE in assertiveness due to code errors.

After the utterances have been classified within the issue categories shown, the message count by type of error can be seen in Figure 4.6. It can be observed that 298 of the 1213 utterances were just movie names and showing that users communicate in 24.6% of messages outside the knowledge base were a form of expression untreated in the knowledge base creation. Furthermore, it was possible to help identify how users expect to interact with a chatbot with this type of analysis.

Figure 4.6. Issue types distribution by total number of FALSE assertiveness utterances.

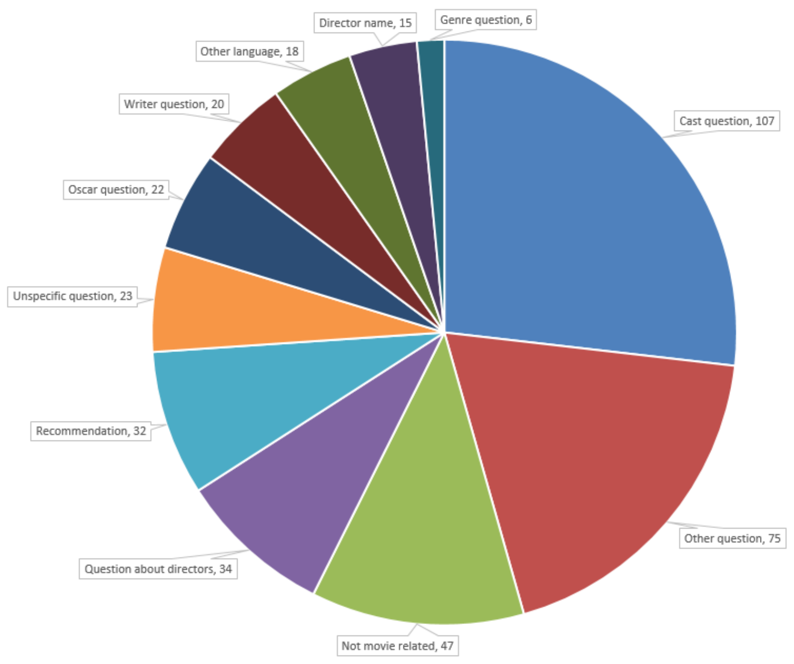


Note that due to the proportion of rule errors, out of scope and other categories, errors resulting from faulty normalization totaled less than 1% of the total. This result is particularly relevant to complement the result of the statistical tests for normalization methods. It reinforces the result of the hypothesis tested that the normalization methods did not affect statistically the assertiveness since this type of issue did not occur in most utterances. It should be noted that this information was obtained from the data of the experiment, and it was not possible to know in advance that the normalization corresponded to this small percentage of the total.

Another information obtained from this analysis was a situation in which utterances were incorrectly marked as FALSE was also found in the data. It was the case utterances of the "bug" category in which users sent messages such as "ok" and there was a rule to answer nothing in case of a match. In this case, assertiveness should have been considered TRUE since there was a rule match. In the case of "nonsense" messages a similar situation occurred since the messages received did not make sense even to a human. Thus, there was no place for a rule to match, creating a false negative error in the computation of the assertiveness metric.

One can also see that in the number of out-of-scope utterances it was 399 or 32.9% of the total. This can be explained due to the restricted number of information about movies that are given to users. To better understand which entries were out of scope, these were further subdivided into the categories in the Figure 4.7.

Figure 4.7. Out of scope subcategories by total number of FALSE assertiveness utterances.



From the total of 399, 107 were questions about the cast (26.8%), 34 about directors (8.5%), 32 asking for movie recommendation (8.0%), 22 asking about the Oscar (5.5%), 20 about movie writers (5.0%), 18 were utterances written in other languages than Portuguese (4.5%), 15 were director names (3.8%) and 6 were asking movie genres (1.5%). That can be used to increase the scope of the bot by giving the information the users most wanted to know. There were also 75 other questions about movies (18.8%), 47 questions not related to movies (11.8%) and 23 unspecific questions where the users did not say specifically what they want about the movie (5.8%).

4.5 Summary and discussion

In this Chapter, the proposed evaluation and monitoring processes were validated through the metrics and indicators. The indicators were first used to aggregate and give an overview comparison of three methods of normalization. The assertiveness and feedback metrics were thereupon used to analyze quantitatively and qualitatively with a hypothesis test, Lasso regression and a classification analysis of issues using the NEGATIVE assertiveness metric.

An attempt of classification analysis was also performed for utterances with NEGATIVE feedback, but there were few samples, and it was not possible to organize them in categories of issues. Even so, it was possible to find a situation in which UGC-Normal and Enelvo normalization methods were distorting names of foreign language movies trying to correct the names based on a Portuguese reference frame. In this case, there is a match of rules giving the answer that the movie was not found, generating the TRUE value for assertiveness without the user receiving the response from the requested movie. This is a limitation of the assertiveness metric, which can be considered as a false positive error.

That problem did not occur with the Rivenorm method, since it is a list of substitutions that does not contain foreign words. It is also a limitation of UGCNormal and Enelvo normalization methods in Brazilian Portuguese, since there is no option to ignore foreign words.

Some issues identified in this section are related to described challenges in Chapter 3. Out of scope input messages related to the scope definition (Minor Challenge 9), rule errors to rule conflicts (Major Challenge 3) and bugs to exception handling (Minor Challenge 3). Other issues are also possible indicators of future challenges in a chatbot lifecycle. The knowledge of the challenges and issues presented in this work may help anticipate some problems faced when developing similar bots.

The issues also revealed that normalization errors, bugs, and nonsense were yielded in a significantly smaller number than out of scope, rule errors, and messages with just the name of a movie. Less effort would have been spent on challenges derived from the use of more sophisticated normalization methods (Minor Challenges 10 and 11) with this information prior to the development of the chatbot. Using a substitution list is a simpler solution that had similar results in this case.

Chapter 5

Conclusions and Future Work

The use of chatbots for several applications brings potential benefits and transformation in people's lives. However, the task of developing, evaluating, and monitoring chatbots are accompanied by challenges. This study tried to elucidate some of them and to analyze a practical case through the KINO chatbot.

Metrics and indicators of assertiveness and feedback were presented for evaluating and monitoring chatbots. The feasibility of its use was also shown in a real case by using three methods of normalization of text in Brazilian Portuguese. As a result of this usage it was possible to observe that the use of different methods had no significant impact on the assertiveness for KINO. Regarding the feedback, the results were not conclusive, requiring a larger number of samples. Thus, for the presented case, the use of the simpler method of normalization through a substitution list would be the most recommended since it had the lowest development cost.

It was found regression analysis with the Lasso algorithm and one-hot-encoding can be used to identify words positively and negatively related to assertiveness. It was possible to verify that the positively associated words were part of the information in which the bot had been programmed to search in its knowledge base. A manual classification was necessary for the words negatively related to understand and identify the types of issues in the identification of the user's intentions.

Manual classification of problem types, while costly, allowed for a better understanding of chatbot behavior and brought some findings:

- Although restricting itself to a closed-domain of movies, chatbot had as the highest volume of issues utterances out of scope. This category was subdivided to show more recurring information demanded by the users and can be used as a reference for improvements in the knowledge base;

- The second category with greater volume was the one the user wrote only the name of the movie. This shows it is crucial to understand how users want to interact with a bot. Using this type of analysis can allow the identification of the way the users communicate to a bot to mitigate that issue;

- Despite the experiment being made by using different normalization methods, only 1% of the messages did not have a matching because of spelling errors and the like. This fact in conjunction with the hypothesis test suggests that normalization is not a major concern for the assertiveness of a chatbot in the initial versions of their knowledge base compared to other issues such as scope;

- There were also limitations in the assertiveness metrics. The manual evaluation of utterances with negative feedback showed that the use of normalization in Portuguese (Enelvo and UGCNormal) generated false positive assertiveness, since they distorted the name of films in other languages. Thus, similar false negative values were generated in "nonsense" and "bug" in FALSE valued assertiveness. Therefore, this metric should not be analyzed in isolation and should not have the sole purpose of increasing itself.

Finally, it should be noted that the metrics proposed are simple and even though analyzed together they can not capture all quality aspects related to chatbots (see Radziwill and Benton [2017]). However, they were sufficient to enable comparison among different methods and offered analysis forms capable of generating insights for improvements and maintenance of the bot.

The conclusions drawn here are not exhaustive and refer to the KINO case, therefore, more studies need to be done to validate them in other bots. Although the use of more complex methods of normalization did not make any difference for the analyzed case, in a more mature chatbot where other categories of problems have already been overcome this can change. Moreover, the use of the proposed metrics needs to be validated on other domains for evaluation and monitoring.

The problem of dialogue in artificial intelligence, even in closed domain, still has challenges. It is possible to propose several possibilities for future work from them. The collected chat data can be used as a basis for retraining the KINO using cross-validation as the Holdout method where typically 2/3 of the data is used for training and 1/3 for testing [Kohavi, 1995]. It would be necessary to improve the automated test methods to reduce conflict in rules (Major Challenge 3) to do so.

In addition, other engines such as Chatscript can be compared to the use of Rivescript and also frameworks using Named Entity Extraction such as Rasa NLU

[Gregori, 2017]. Open databases of ontologies such as DBPedia[1] can be used as sources to extend the domain of conversations and increase the semantic value [Ismayilov et al., 2018]. Other approaches can also be tested as artificial neural networks in an attempt to improve the user's intent identification and even for output generation.

Other fronts of research are needed in conversational design and in how to capture user feedback. Usability studies, satisfaction surveys, and conversation strategies have the potential to improve the results obtained. Feeling analysis techniques can also be used to automate the user's feedback. Finally, the use of speech recognition and synthesis can be used to improve the user's experience, especially those with visual difficulties as in Manfio [2017].

[1]http://wiki.dbpedia.org/

Bibliography

Abdul-Kader, S. A. and Woods, J. (2015). Survey on chatbot design techniques in speech conversation systems. *Int. J. Adv. Comput. Sci. Appl.(IJACSA)*, 6(7).

Adams, T. (2017). Ai-powered social bots. *arXiv preprint arXiv:1706.05143*.

Agresti, A. (2007). *An introduction to categorical data analysis*, volume 135. Wiley New York, 2nd edition.

al Rifaie, M. M. (2017). Loebner Prize. `http://www.aisb.org.uk/events/loebner-prize`. [Online; accessed 5-October-2017].

Alan Turing, M. (1950). Turing. Computing machinery and intelligence. *Mind*, 59(236):433--460. ISSN 0026-4423.

Alpaydin, E. (2014). *Introduction to machine learning*. MIT press.

Anderson, D. (1987). Is the Chinese room the real thing? *Philosophy*, 62(241):389--393.

Athreya, R. G., Ngonga Ngomo, A.-C., and Usbeck, R. (2018). Enhancing community interactions with data-driven chatbots–the dbpedia chatbot. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, pages 143--146, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

Avanço, L. V. (2015). Sobre normalização e classificação de polaridade de textos opinativos na web. Master's thesis, Universidade de São Paulo.

Bartl, A. and Spanakis, G. (2017). A retrieval-based dialogue system utilizing utterance and context embeddings. *arXiv preprint arXiv:1710.05780*.

Berglund, F. (2017). Chatbots as interaction modality: An explorative design study on elderly classical music concert subscribers.

Bertaglia, T. F. C. (2017). Normalização textual de conteúdo gerado por usuário. Master's thesis, Universidade de São Paulo.

Bertaglia, T. F. C. and Nunes, M. d. G. V. (2016). Exploring word embeddings for unsupervised textual user-generated content normalization. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 112--120, Osaka, Japan. The COLING 2016 Organizing Committee.

Both, A., Diefenbach, D., Singh, K., Shekarpour, S., Cherix, D., and Lange, C. (2016). Qanary — a methodology for vocabulary-driven open question answering systems. In *Proceedings of the 13th International Conference on The Semantic Web. Latest Advances and New Domains - Volume 9678*, pages 625--641, Berlin, Heidelberg. Springer-Verlag.

Bradeško, L. and Mladenić, D. (2012). A survey of chatbot systems through a loebner prize competition. In *Proceedings of Slovenian Language Technologies Society Eighth Conference of Language Technologies*, pages 34--37.

Cerf, V. (1973). Rfc 439: Parry encounters the doctor. RFC, RFC Editor. `https://tools.ietf.org/html/rfc439`, [Online; accessed 17-May-2018].

Cifuentes, J. (2018). Bots landscape. Technical report, VBProfiles. [Online; accessed 16-April-2018].

Cochran, W. G. (1954). Some methods for strengthening the common χ 2 tests. *Biometrics*, 10(4):417--451.

Colby, K. M., Hilf, F. D., Weber, S., and Kraemer, H. (1972). Experimental validation of a computer simulation of paranoid processes. *Mathematical Biosciences*, 15(1-2):187--191. ISSN 00255564.

Deryugina, O. (2010). Chatterbots. *Scientific and Technical Information Processing*, 37(2):143--147.

Dias, G. A., de Brito Neves, D. A., de Morais Silva, J. W., do Nascimento Neto, G. H., and de Almeida, M. J. S. C. (2007). Representando o conhecimento através de ontologias: o caso do chatterbot lunmi. *Encontro Nacional de Pesquisa em Ciência da Informação*, 8.

Duran, M. S., Volpe Nunes, M. d. G., and Avanço, L. (2015). A normalizer for ugc in brazilian portuguese. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 38--47, Beijing, China. Association for Computational Linguistics.

Facebook (2017). 1.3 billion people on messenger. `https://www.facebook.com/messenger/photos/a.882538591865822.1073741828.602814669838217/1530168793769462/?type=3&theater`. [Online; accessed 4-February-2018].

Facebook (2018). Sending messages. `https://developers.facebook.com/docs/messenger-platform/send-messages/`. [Online; accessed 4-February-2018].

Faught, B., Colby, K. M., and Parkison, R. C. (1974). The interaction of inferences, affects, and intentions, in a model of paranoia. Technical report, STANFORD UNIV CA DEPT OF COMPUTER SCIENCE.

Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., Lally, A., Murdock, J. W., Nyberg, E., Prager, J., et al. (2010). Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59--79.

Freese, E. (2007). Enhancing aiml bots using semantic web technologies. In *Proc. of Extreme Markup Languages*, pages 1--27.

Fryer, L. and Carpenter, R. (2006). Bots as language learning tools. *Language Learning & Technology*.

Gao, J., Galley, M., and Li, L. (2018). Neural approaches to conversational ai. *arXiv preprint arXiv:1809.08267*.

Gregori, E. (2017). Evaluation of modern tools for an omscs advisor chatbot. Master's thesis, Georgia Institute of Technology.

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*, pages 928--939.

Ismayilov, A., Kontokostas, D., Auer, S., Lehmann, J., Hellmann, S., et al. (2018). Wikidata through the eyes of dbpedia. *Semantic Web*, 1(Preprint):1--11.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*, volume 112. Springer.

Kadlec, R., Schmid, M., and Kleindienst, J. (2015). Improved deep learning baselines for ubuntu corpus dialogs. *CoRR*, abs/1510.03753.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, pages 1137--1143, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Li, J., Monroe, W., Ritter, A., Galley, M., Gao, J., and Jurafsky, D. (2016). Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541*.

Loebner, H. G. (1994). In response. *Commun. ACM*, 37(6):79--82. ISSN 0001-0782.

Lucas, J. (2009). Commentary on Turing's "Computing Machinery and Intelligence". *Parsing the Turing Test*, pages 67--70.

Maeda, A. and Moraes, S. (2017). Chatbot baseado em deep learning: um estudo para língua portuguesa. In *Symposium on Knowledge Discovery, Mining and Learning, 5th*.

Majumder, A., Pande, A., Vonteru, K., Gangwar, A., Maji, S., Bhatia, P., and Goyal, P. (2018). Automated assistance in e-commerce: An approach based on category-sensitive retrieval. In Pasi, G., Piwowarski, B., Azzopardi, L., and Hanbury, A., editors, *Advances in Information Retrieval*, pages 604--610, Cham. Springer International Publishing.

Manfio, E. R. (2017). Um robô linguista que 'ouve'e 'fala': Geolinguística, pln e tabelas hash em concurso. *Signum: Estudos da Linguagem*, 19(2):415--439.

Masche, J. and Le, N.-T. (2017). A review of technologies for conversational systems. In Le, N.-T., van Do, T., Nguyen, N. T., and Thi, H. A. L., editors, *Advanced Computational Methods for Knowledge Engineering*, pages 212--225, Cham. Springer International Publishing.

Mauldin, M. L. (1994). Chatterbots, tinymuds, and the turing test: Entering the loebner prize competition. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (Vol. 1)*, AAAI '94, pages 16--21, Menlo Park, CA, USA. American Association for Artificial Intelligence.

Moraes, S. M. and de Souza, L. S. (2015). Uma abordagem semiautomática para expansão e enriquecimento linguístico de bases aiml para chatbots. *Nuevas Ideas en Informática Educativa, TISE*, pages 600--605.

Oliveira, S. F. G. (2017). Interfaces conversacionais-chatbot para a casa da música. Master's thesis, Universidade do Porto.

Petherbridge, N. (2018a). Rivescript. `https://www.rivescript.com/about`. [Online; accessed 14-January-2018].

Petherbridge, N. (2018b). Rivescript vs. aiml. `https://www.rivescript.com/compare/aiml`. [Online; accessed 18-April-2018].

Plummer, D. C., Andrews, W., Marquis, H., McGuire, M., Chesini, F., Leow, A., Reynolds, M., Lovelock, J.-D., Hill, J. B., Furlonger, D., Smith, D. M., Golvin, C. S., Velosa, A., McCoy, D. W., Kutnick, D., Sicular, S., Perkins, E., and Revang, M. (2017). Top strategic predictions for 2018 and beyond: Pace yourself, for sanity's sake. Technical report, Gartner. [Online; accessed 16-April-2018].

Radziwill, N. M. and Benton, M. C. (2017). Evaluating quality of chatbots and intelligent conversational agents. *arXiv preprint arXiv:1704.04579*.

Russell, S. and Norvig, P. (2009). *Artificial Intelligence: A Modern Approach*. Prentice Hall Press.

Searle, J. R. (1980). Minds, brains, and programs. *Behavioral and brain sciences*, 3(3):417--424.

Shang, L., Lu, Z., and Li, H. (2015). Neural responding machine for short-text conversation. *CoRR*, abs/1503.02364.

Shieber, S. M. (1994). Lessons from a restricted Turing test. *arXiv preprint cmp-lg/9404002*.

Silva, J. M. (2012). Chatterbots podem ser úteis? Associate's thesis, Faculdade de Tecnologia de São Paulo.

Slack (2018). Real time messaging api. `https://api.slack.com/rtm`. [Online; accessed 4-February-2018].

Sônego, A. A., Bernardini, A. A., and Pozzebon, E. (2018). Chatbots: Uma análise bibliométrica do estado da arte da literatura. *ARTEFACTUM-Revista de estudos em Linguagens e Tecnologia*, 16(1).

Sterrett, S. G. (2003). Turing's two tests for intelligence. In *The turing test*, pages 79--97. Springer.

Surmenok, P. (2016). Chatbot architecture. `http://pavel.surmenok.com/2016/09/11/chatbot-architecture/`. [Online; accessed 18-April-2018].

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267--288.

TMDb (2018). Api overview - the movie database (tmdb). `https://www.themoviedb.org/documentation/api`. [Online; accessed 14-January-2018].

Tzou, M.-T., Lu, C.-H., Wang, C.-C., Lee, C.-W., and Hsu, W.-L. (2007). Extending knowledge of AIML by using RDF. *ICCE*.

Wallace, R. S. (2003). The elements of aiml style. *Alice AI Foundation*.

Wallace, R. S. (2009). The anatomy of ALICE. *Parsing the Turing Test*, pages 181--210.

Weizenbaum, J. (1966). Eliza - a computer program for the study of natural language communication between man and machine. *Commun. ACM*, 9(1):36--45. ISSN 0001-0782.

Yan, Z., Duan, N., Bao, J., Chen, P., Zhou, M., Li, Z., and Zhou, J. (2016). Docchat: An information retrieval approach for chatbot engines using unstructured documents. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 516--525, Berlin, Germany. Association for Computational Linguistics.

Appendix A

KINO's rules

An example for each category of rule in Section 3.1.2 are shown below. A complete list of rules used in the experiment is available on <https://github.com/staticdev/kino-rivescript-experiment>.

A.1 Name rules component

```
! var name = KINO

+ <bot name>
- Sou eu.
- <bot name> sou eu.
- Eu mesmo..

+ [<bot name>] o que <bot name> (é|significa)
- Meu nome vem do grego kinein que significa movimento. Kino em
  várias linguas, como esperanto e alemão, significa cinema..
  até a palavra movie em inglês vem de move que também é
  movimento.
```

A.2 General rules component

A.2.1 Basic personality

```
+ o que [você] (consegue|pode|sabe) [fazer|responder|sobre] [*]
- Comecei agora a aprender sobre filmes. Sei falar os
   diretores, produtores, data de lançamento, duração, sinopse e
   avaliação de filmes.
```

A.2.2 Feedback rules

```
! array agradecer =
   agradecer|agradece|agradeço|agradeces|agradecemos|agradecem|
   agradecendo|agradecido|agradecida|brigado|brigada|obrigado|
   obrigada|muito obrigado|muito obrigada|valeu

+ [*] (@agradecer) [*]
- {"reply":"Não há de quê.", "feedback":"POSITIVE"}
- {"reply":"De nada!", "feedback":"POSITIVE"}
- {"reply":"Por nada.", "feedback":"POSITIVE"}
- {"reply":"Estou às ordens, disponha.", "feedback":"POSITIVE"}
```

A.2.3 Domain specific rules

```
+ [e|mas] [você] [sabe|conhece] quem (dirige|dirigiu|irão
   dirigir|vão dirigir) [o filme] *
- <call>get_movie_predicate diretor <star2></call>
```

A.2.4 Catch-all rule

```
+ *
- {"reply":"Desculpe não consegui entender o que disse. Tente
   fazer perguntas sobre algum filme.", "assertiveness":"0"}
- {"reply":"Sou meio burrinho ainda. Consegue falar isso com
   outras palavras?", "assertiveness":"0"}
```

```
- {"reply":"Está difícil entender.. Talvez se você perguntar
   informações, do tipo 'qual a história do filme tal?', eu
   consiga te responder melhor.", "assertiveness":"0"}
- {"reply":"Não entendi. Para me ajudar, fale o nome do filme e
   a informação que quer saber sobre ele na mesma mensagem.",
   "assertiveness":"0"}
```