1 2 9 0

UNIVERSIDADE Đ
COIMBRA

Isabel Maria Rocha Correia de Carvalho

SENTIMENT ANALYSIS IN THE CONTEXT OF DIALOGUE

Dissertation in the context of the Master in Informatics Engineering, Specialization in Intelligent Systems, advised by Professor Hugo Oliveira (DEI) and Professor Catarina Silva (DEI) and presented to
Faculty of Sciences and Technology / Department of Informatics Engineering.

September 2022

Isabel Maria Rocha Correia de Carvalho
SENTIMENT ANALYSIS IN THE CONTEXT OF DIALOGUE
UNIVERSIDADE Đ COIMBRA
1 2 9 0

Faculty of Sciences and Technology
Department of Informatics Engineering

Sentiment analysis in the context of dialogue

Isabel Maria Rocha Correia de Carvalho

Dissertation in the context of the Master in Informatics Engineering, Specialization in Intelligent Systems, advised by Professor Hugo Oliveira (DEI) and Professor Catarina Silva (DEI) and presented to the
Faculty of Sciences and Technology / Department of Informatics Engineering.

September 2022

1 2 9 0

UNIVERSIDADE D
COIMBRA

Acknowledgments

This work was funded by the project POWER (grant number POCI-01-0247-FEDER-070365), co-financed by the European Regional Development Fund (FEDER), through Portugal 2020 (PT2020), and by the Competitiveness and Internationalization Operational Programme (COMPETE 2020).

My personal acknowledgments will be written in Portuguese, as any word of appreciation is more impactful when in our native language. Also, I want to make sure anyone who is part of my story is able to understand it.

Acredito que não somos mais que o conjunto de pessoas com quem dividimos espaço e tempo. Afinal, "aqueles que passam por nós, não vão sós, não nos deixam sós. Deixam um pouco de si, levam um pouco de nós". O meu caminho não seria o mesmo sem todos que comigo se cruzaram, com tudo de bom e tudo de mau, erros e aprendizagens, distanciamentos e aproximações. Assim sendo, só posso estar grata a todos que deixaram comigo uma parte de si.

No entanto, há sempre presenças mais notórias e brilhantes, que se fazem notar e nos guiam quando estamos mais perdidos, e que não posso deixar de nomear. A todo e qualquer nível, agradeço aos meus pais e ao meu irmão, com quem posso caminhar lado a lado, especialmente quando há um tropeção. À minha família do Norte que sempre soube estar presente mesmo quando estamos afastados, em especial ao meu padrinho e aos meus avós. À minha família paterna, a de laços de sangue mais intensos, que, sendo de Coimbra e nunca tendo participado numa Benção das Pastas (nem nas próprias), estiveram presentes para me apoiar nessa ocasião especial (como em muitas outras), e a de laços de sangue mais ténues, que nem por isso atenuam o carinho, amizade, e presença na minha vida. Também um grande agradecimento a quem desde o berço me acompanha e acarinha, como se de uma segunda mãe se tratasse.

A nível académico e profissional, tenho muito a agradecer ao Professor Hugo e à Professora Catarina, exemplos a seguir para qualquer pessoa que escolha esta tão digna profissão e a queira execer com preocupação e interesse pelos seus alunos. Graças a eles, saio desta fase mais rica e com mais experiência do que pensei vir a ter, e agradeço em especial a tolerância e o tempo dispensado quando, inevitável e regularmente, me deparava com problemas que não conseguia resolver. E claro, obrigada pelas oportunidades que me apresentaram, e a fé nelas demonstrada.

Por fim, a níveis que não sei nomear, e a quem enriquece qualquer percurso, aos amigos. Às de sempre e, por este andar, para sempre. Aos de aventuras passadas numa serra brilhante que permanecem até hoje, agora em qualquer lugar. Aos resistentes do ano de caloiro, que poderiam dizer que estamos com os pés para a cova. Aos que me permitiram crescer fora da zona de conforto e experimentar competências que desconhecia. E a quem caminha a meu lado, com um ombro e um ouvido disponíveis, deixando a sua pegada vincada na minha história.

Um muito obrigada a todos vós que, direta ou indiretamente, me guiaram até ao que alcancei hoje. Tem sido um prazer fazer este caminhada convosco.

Abstract

Sentiment Analysis (SA) in dialogue aims at detecting the sentiment expressed in utterances, which, as applied in this work, may improve human-computer interaction in natural language. In this dissertation, we explore different approaches for SA in written Portuguese dialogues, mainly in the domain of Telecommunications. If integrated into a conversational agent, this will enable the automatic identification and a quick reaction upon customers manifesting negative sentiments, possibly with human intervention, hopefully minimising the damage. We created two datasets of real data, with manually annotated sentiment: one with dialogues from a call center, provided by Altice Labs (AL); another of Twitter conversations primarily involving the accounts of Telecommunications companies. We compare the performance of different approaches with varying complexities, from lexicon-based models, to shallow learning classifiers (e.g., Random Forest, Logistic Regression) as well as more recent deep learning approaches (e.g., Fine-tuned Bidirectional Encoder Representations from Transformers (BERT), Few-Shot Learning). Since a dialogue is a sequence of utterances, the previous sentences may impact the sentiment of the current sentence. Hence, we also developed models that consider the context (e.g., BERT-Conditional Random Field (BERT-CRF)). Every Machine-Learning model, except the latter group, is analyzed with and without considering the previous utterances. When classifying the utterances, the best model (Fine-tuned BERT) achieved F1-Scores of 0.87 and 0.93 in the AL and Twitter datasets, respectively. The performance of the former was achieved without considering context, and the latter was achieved while considering it (by concatenating the current and previous utterances). However, in most scenarios, the context seems to decrease the performance of the classifiers, meaning that, in this application, the current utterance can be enough. These are interesting results and suggest that automated customer support may benefit from a sentiment detection feature. The developed approach will be made available for the consideration of AL, for integration into their customer assistance system.

Keywords

Natural Language Processing; Dialogue Analysis; Sentiment Analysis; Text classification; Data mining.

Resumo

A Análise de Sentimento em diálogo visa detectar o sentimento expresso em frases e, da forma em que foi aplicada neste trabalho, tem o potencial de melhorar a interação humano-computador em linguagem natural. Nesta dissertação, exploramos diferentes abordagens para Análise de Sentimento em diálogos escritos em Português, principalmente no domínio das Telecomunicações. Se integrado num agente conversacional, isto permitirá a identificação automática e uma reação rápida a clientes que manifestam sentimento negativo, possivelmente com intervenção humana, podendo assim minimizar os danos. Foram criados dois datasets de dados reais, com sentimento manualmente anotado: um com diálogos de call center, cedido pela Altice Labs (AL); outro a partir de conversas extraídas do Twitter, envolvendo principalmente contas de empresas de Telecomunicações. Comparamos o desempenho de diferentes abordagens com complexidades variáveis, desde modelos baseados em léxico, passando por classificadores mais tradicionais (e.g., Random Forest, Regressão Logística), até modelos mais recentes (e.g., Fine-tuned Bidirectional Encoder Representations from Transformers (BERT), Few-Shot Learning). Como um diálogo é uma sequência de falas, as frases anteriores podem ter impacto no sentimento da frase atual. Assim sendo, também desenvolvemos modelos que consideram contexto (e.g., BERT-Confitional Random Field (BERT-CRF)). Cada modelo de aprendizagem computacional, com a excepção do último grupo, é analisado com e sem a inclusão de falas anteriores. Ao classificar os datasets, o melhor modelo (Fine-tuned BERT) atingiu F1-Scores de 0,87 e 0,93 nos datasets da Altice Labs e do Twitter, respetivamente. O desempenho do primeiro foi alcançado sem considerar contexto, enquanto o do segundo foi alcançado considerando contexto (inclusão das falas anteriores). No entanto, na maioria dos cenários, a utilização de contexto parece diminuir o desempenho dos classificadores, significando que, para esta aplicação, a utilização da fala atual pode ser suficiente. Estes são resultados interessantes que sugerem que um apoio ao cliente automático pode beneficiar de um componente de análise de sentimento. A abordagem desenvolvida será disponibilizada à AL, que poderá considerar a sua integração no seu sistema de assistência ao cliente.

Palavras-Chave

Processamento de Linguagem Natural; Análise de Diálogo; Análise de Sentimentos; Classificação de Texto; Mineração de dados.

Contents

Acronyms

AL Altice Labs. vi, xi, xii, xiii, 1, 2, 3, 6, 39, 42, 43, 44, 48, 50, 51, 52, 53, 56, 57, 58, 59, 60, 65, 66, 69, 74, 77, 80, 81, 82, 83, 84, 85, 87, 89, 91, 92, 93, 94, 96, 97, 98, 99, 100, 112, 115, 116, 117, 118, 119, 121, 122

API Application Programming Interface. vi, 11, 24, 34, 41, 50, 51, 52, 65, 66, 69, 74, 77, 85, 86, 87, 88, 94, 95, 97

ASR Automatic Speech Recognition. vi, 40

AUC Area Under the ROC Curve. vi, 3, 18, 19, 79, 80, 82, 83, 84, 87, 94

BERT Bidirectional Encoder Representations from Transformers. vi, 11, 12, 37, 38, 39, 41, 45, 62, 63, 65, 66, 69, 70, 71, 72, 74, 77, 85, 86, 89, 93, 94, 95, 97, 98, 99, 120

BiLSTM Bi-directional LSTM. vi, 38, 39

BoF Bag-of-Features. vi, 9

BOS Beginning of Speech. vi, 70, 72, 77

BoW Bag of Words. vi, 9

CDCN Context-aware Dynamic Convolution Network. vi, 38

CNN Convolutional Neural Network. vi, 22, 34, 35, 38

Co-GAT Co-Interactive Graph Attention Network. vi, 38

CRF Conditional Random Field. vi, xii, 20, 34, 35, 45, 62, 63, 69, 70, 71, 72, 74, 76, 77, 85, 86, 87, 89, 92, 93, 94, 95, 97, 98, 99, 114, 115, 120

DA Dialogue Analysis. vi, 4, 5, 29, 31, 32

DAR Dialogue Act Recognition. vi, 29, 31, 33, 37, 38, 41, 55, 99

DCN Dynamic Convolution Network. vi, 38

DCR-Net Deep Co-Interactive Relation Network. vi, 38, 39

DEAP Distributed Evolutionary Algorithms in Python. vi, 15

DEI Department of Informatics Engineering. vi

Few-SL Few-Shot Learning. vi, 22, 23, 24, 45, 50, 66, 67, 68, 69, 74, 77, 84, 85, 87, 88, 99

FN False Negative. vi, 16, 17, 80, 82

FP False Positive. vi, 16, 17, 80

FPR False Positive Rate. vi, 18, 19

GA Genetic Algorithm. vi, 15, 74, 100

GAT Graph Attention Network. vi, 22, 39

GloVe Global Vector. vi, 10, 11, 35

GPT Generative Pre-trained Transformer. vi, 11, 12, 23, 44, 49, 50, 51, 66, 67, 84, 99

HAN Hierarchical Attention Network. vi

HMM Hidden Markov Model. vi, 20

K-NN K-Nearest Neighbors. vi, 21

L-BFGS Limited-Memory Broyden-Fletcher-Goldfarb-Shanno. vi, 75, 76

LEIA Léxico para Inferência Adaptada. vi, 27

LibLinear Library for Large Linear Classification. vi, 75

LIWC Linguistic Inquiry and Word Count. vi, 36

LSTM Long Short-Term Memory. vi, 22, 34, 70

ML Machine Learning. vi, 2, 3, 4, 5, 12, 13, 19, 20, 31, 32, 34, 43, 46, 62, 64, 121, 122

MLM Masked Language Model. vi, 12, 72

MLP Multilayer Perceptron. vi, 21, 22, 38

MultiNLI Multi-Genre Natural Language Inference. vi, 24

N.A. Not Applicable. vi, 84

NER Named-Entity Recognition. vi, 71

NLI Natural Language Inference. vi, 58, 65, 77, 83, 84, 87, 90, 91, 92, 94, 97, 100, 116

NLLL Negative Log Likelihood Loss. vi, 71, 72

NLP Natural Language Processing. vi, 4, 5, 6, 8, 11, 29, 31, 32, 33, 35, 63, 96, 97, 98, 100

NN Neural Network. vi, 10, 11, 21, 22

NSP Next Sequence Prediction. vi, 12, 72

NYI Not Yet Implemented. vi

P.O.S. Parts of Speech. vi

PBT Population-Based Training. vi, 15, 74, 100

PCA Principal Component Analysis. vi, 99

ReLi Resenha de Livros. vi, 36, 39, 40

RNN Recurrent Neural Network. vi, 22

ROC Receiver Operating Characteristic. vi, 18, 19

SA Sentiment Analysis. vi, 1, 2, 3, 4, 5, 12, 25, 26, 27, 29, 31, 32, 33, 34, 35, 36, 37, 38, 41, 55, 62, 63, 66, 69, 78, 82, 83, 85, 87, 93, 94, 95, 96, 97, 98, 99, 100

SemEval International Workshop on Semantic Evaluation. vi, 33

SVM Support Vector Machine. vi, 3, 25, 34, 35, 37, 43, 62, 64, 74, 75, 81, 82, 89, 90, 91, 94, 116

t-SNE t-distributed Stochastic Neighbor Embedding. vi, 8, 10, 58, 99

TF-IDF Term Frequency - Inverse Document Frequency. vi, xi, xii, 9, 10, 37, 58, 64, 65, 71, 77, 81, 85, 90, 91, 92, 97, 116, 117

TN True Negative. vi, 16, 17, 82

TNR True Negative Rate. vi, 18

TP True Positive. vi, 16

TPR True Positive Rate. vi, 18, 19

VADER Valence Aware Dictionary and Sentiment Reasoner. vi, 27

WOZ Wizard of Oz. vi, 40, 41, 42, 44, 99

Zero-SL Zero-Shot Learning. vi, 22, 23, 24, 63, 68, 69, 74, 77, 84, 85, 87, 88, 94, 95, 99, 119

List of Figures

List of Tables

This page is intentionally left blank.

Chapter 1

Introduction

Conversational systems, or chatbots, are becoming increasingly present in our daily lives (Dale, 2016), from guiding the user through a purchase to providing assistance when the user has a complaint about the provided service or product. These systems should understand the speaker and reply in natural language even though, in some cases, there is tolerance for incorrect responses. However, the replacement of human-to-human interaction by conversational agents may sometimes lead to a lack of understanding of the client's wishes, and frustration may build up and cause the loss of the client. This makes it relevant for companies that rely on conversational systems for assisting their clients to employ a fallback solution that considers such situations. For instance, if it was possible to automatically identify that the client would benefit from a person-to-person conversation, the switch could be performed between the human and the artificial agent, ideally, without the client even noticing. This would hopefully contribute to decreasing the number of unsatisfied clients, with further benefits to the companies' business.

Since the aforementioned situations often result in negative sentiments developed by the client, the main goal of this dissertation is sentiment analysis in the context of dialogue, either between humans or between humans and conversational agents. A potential use case would be an automatic monitorisation of the dialogue that, once an unusual amount of negative sentiment is identified, will mark the conversation as requiring the intervention of a human assistant (e.g., in a call center). Sentiment Analysis (SA) is a text classification technique used to determine the sentiment expressed in natural language. In the simplest case, this is a two or three-class problem (positive, negative, neutral) (Carvalho et al., 2011, Duarte, 2013, Kouloumpis et al., 2021).

The work developed must be applied to Portuguese, as it is part of a project in collaboration with Altice Labs (AL)[1], a company that is built upon more than 65 years of technological experience in telecommunications, and, as of 2020, provides fiber network for over 5 million houses and companies in Portugal (Altice Portugal, 2020). This language restriction poses another challenge because, while approaches to a similar goal have been proposed, they are generally applied to other languages, mostly English. It also means that conversational datasets used in the aforementioned works for training and evaluation are hardly usable in this work. Therefore, the compilation of one or more datasets of dialogue in Portuguese is also a goal of this work. AL defined their interest in the domains of eCommerce, TV, Health Care, Finance and FinTech, and Telecommunications, so, ideally, the datasets should contain these domains.

[1] `https://www.alticelabs.com/`

This dissertation was written in the context of the Master in Informatics Engineering, with a specialization in Intelligent Systems, and in collaboration with AL, and specifically to the project POWER (POCI-01-0247-FEDER-070365), subproject P4 Future Services, which, besides other activities, aims to develop the current state of the companies' chatbots.

In this chapter, we will go over the objectives defined for this dissertation and its contributions, the approach chosen to achieve them, and a brief presentation of each of the following chapters in this document.

1.1 Objectives

As mentioned earlier, the main objective of this dissertation is to develop research aiming at sentiment analysis in the context of dialogue. To achieve this, a series of smaller goals must be achieved, which are listed below:

- Creation, annotation, and analysis (e.g., number of dialogues, the average number of turns) of a data set to be used to train the models;
- Exploration of different models/solutions applied to the Portuguese language. Application of techniques that range from lexicon-based classification to supervised Machine Learning (ML) (including deep neural networks), to perform sentiment classification in each utterance. Analysis of results and evaluation metrics obtained throughout this dissertation;
- Identification of solutions, among the developed models, with better performance than the SA system currently being considered by AL;
- Writing of a scientific article related to the work developed in this dissertation.

The definition of these objectives is important as it sets the work direction for the dissertation, and makes its purpose clearer.

1.2 Approach

SA in the context of dialogue allows the classification of each utterance's polarity (e.g., Positive, Negative), and makes it possible to identify situations that cause the client to switch to negative sentiment. If they demonstrate negative sentiments and the chatbot is not able to overcome the negativity, or if, for example, they are repeating the same question over and over, then it is time for the fallback system to act and mark the conversation as requiring human intervention. Hopefully, the person will be better equipped to manage the situation and avoid the loss of a client.

Any solution requires data, and so this was the first challenge of this dissertation. To some extent, it is possible to rely on real anonymized transcriptions of conversations, provided by AL, where the sentiment has to be manually labeled. However, due to privacy reasons, there are limitations on the quantity of data provided and on what can be done with it. Therefore, we experimented with other options (e.g., data extraction from social networks, translation of available datasets, generation of data). Since there were some restrictions regarding the dataset (e.g., containing dialogues, being in the Portuguese language), the choice of the data source was not particularly easy. In the end, we developed two datasets, one with the samples provided by AL, and another with dialogues extracted from Twitter.

After the data collection, we annotated the sentiment associated with each utterance and analysed the datasets (e.g., frequency of each label, number of dialogues). The annotation was performed manually by a group of people, but we also experimented with emoji-based annotation. The data curation process will be explained in detail in Chapter 5.

We developed several solutions, ranging from a lexicon-based approach, to shallow ML models (e.g., Naive Bayes, Support Vector Machine (SVM)s), and to deep architectures (e.g., based on transformers). These models will be presented in Section 2.2.5, and a further explanation of the development process will be explained in Chapter 6.

We set the evaluation metrics which allow for an analysis of how well the classifier is differentiating between the Negative and Non-Negative classes. We focused on the Recall, F1-Score, and Area Under the ROC Curve (AUC) metrics, because we believe that it is more important to correctly determine the Negative sentiments than the Non-Negative (i.e., it is potentially worse to identify an unsatisfied client as satisfied and failing to manage that situation). These evaluation metrics, and more, will be presented in Section 2.2.4, and the analysis of the most relevant metrics for this work will be further explained in Chapter 7. Furthermore, we used tuning techniques in an attempt to explore different settings for each classifier and possibly improve their performance.

The evaluation and comparison of each solution allowed us to determine which models could be better adjusted to the goal of this dissertation (SA in dialogues).

Furthermore, based on the results and conclusions obtained throughout this dissertation, we had the chance to write scientific articles on this topic.

This whole process allowed us to tackle each of the objectives set and listed earlier.

1.3 Contributions

Overall, the work developed contributes to the SA field in the following ways:

- Creation of two manually-annotated datasets containing dialogues in the Portuguese language, and in the domains specified as relevant by AL (eCommerce, TV, Health Care, Finance and FinTech, and Telecommunications). Such datasets are currently lacking in the scientific community, which makes this a large contribution of this work, especially since one of the datasets was made available to the public;
- Development of solutions with some degree of innovation, even if just due to the fact that they are trained in the Portuguese language;
- Extensive experimentation with different classifiers and techniques, and a comparison of their performances for SA in dialogues;
- Identification of the models that perform better than the current solution used by AL, meaning the company can benefit from the integration of a chosen solution in their system;
- Creation and submission of scientific articles, which, if accepted into the correspond-

ing conferences (IberSPEECH 2022[2] and RECPAD 2022 [3]), would allow us to share the methods used and results obtained with a larger audience.

These contributions, as well as the objectives, can be used to evaluate the quality of this dissertation.

1.4 Document Structure

This section briefly presents what each chapter in this document refers to:

- Chapter 2 presents background on the relevant areas of knowledge, namely Natural Language Processing (NLP), ML, SA, and Dialogue Analysis (DA), which all relate to the main goal of this dissertation;
- Chapter 3 presents the related work that was studied and found relevant to the development of this dissertation, namely in the contexts of SA using Twitter data, SA in the Portuguese language, since these topics will be explored in the development of this work. In this chapter we also present works using SA in combination with DA, but we did not implement this approach;
- Chapter 4 presents the pre-arranged work plan envisioned during the first months of this work, and compares it with the real work plan, presenting the defined tasks and the time spent working on each of them;
- Chapter 5 presents the general ML approach used, and focuses on the data gathering, analysis, and annotation processes;
- Chapter 6 presents the experiments performed for each classifier, and explains each approach;
- Chapter 7 presents the final results and evaluation metrics for each classifier developed, and comparisons between some aspects (e.g., the use of context);
- Chapter 8 stresses the main conclusions taken from the work and analysis developed in this dissertation, as well as possible directions for future work.

[2] `http://iberspeech2022.ugr.es/`
[3] `https://recpad2022.ipleiria.pt/`

Chapter 2

Background

This chapter presents background on the topics related to Sentiment Analysis (SA) that constitute the support for the development of the solutions to this dissertation's problem.

In the first section, 2.1, the focus will be on Natural Language Processing (NLP). This domain consists of a set of techniques for computers to manipulate contents in human language. Since this dissertation needs to handle human-to-human and human-to-computer conversations, it is important to have some background knowledge on the workings of NLP.

In the second section, 2.2, the focus will be on Machine Learning (ML). This was found relevant since there are several approaches to sentiment analysis that are based on ML and, therefore, most of the solutions explored will belong to this domain.

In the third section, 2.3, the focus will be on SA. Since SA is the main goal of this dissertation, it is essential to provide some background knowledge on the subject. This section will present applications, challenges, approaches (besides ML), emotion models, and subtasks of SA.

In the fourth section, 2.4, we will present the Dialogue Analysis (DA) domain, which could be compelling because there is a relation between DA and SA, and their joint analysis could provide interesting results.

In the fifth and final section of this chapter, 2.5, a summary of the domains above will be presented.

2.1 Natural Language Processing

NLP aims to get computers to perform tasks involving human language. Some of these tasks are summarized below:

- Machine translation: automatic translation of a document in one language to another language (Brown et al., 1990);
- Text classification: automatic labeling of documents, sentences, or words with a class from a predefined set. SA is a common application of this task, mainly in product reviews or social media posts. Some applications include allowing marketeers to know how people respond to their products or advertisements, social scientists to study how emotions spread over social networks, and services to implement recommendation

systems (Duarte, 2013). The set of classes could consist, in a simple scenario, of Positive, Neutral, and Negative sentiments;

- Information retrieval: retrieval of the most useful information, given an information need (Manning et al., 2008). This is important to understand intents (i.e., what kind of information someone is looking for when they make a request) in question-answering, search systems, or dialogue systems;

- Dialogue system: a computational system that the user establishes a conversation with, using natural language. The system is capable of detecting and satisfying the user's intent (Einsenstein, 2018);

- Document summarization: finds elements of interest in documents to produce an overview of what is most important (Otter et al., 2021).

Although much progress has been made in the NLP field, there are still many challenges, mainly due to the diversity and constant evolution of human language (e.g., the number of different languages, their dialects, the creation of new words, and the use of slang). Knowing this, it is important that the algorithms are robust to samples that do not occur in the training data. This also applies to speech recognition, a domain that relates to this work because the sampled data from Altice Labs (AL) was built using speech recognition techniques. The more knowledge a system has about language, the more tolerant it will be to previously unseen vocabulary, either due to the evolution of language or to noisy speech recognition. Different challenges will be addressed in the remainder of this section, as well as some language properties.

One of the basic principles of language is the principle of compositionality. This means that phrases and documents are composed of smaller units, words, that give meaning to the whole composition. The meaning of a word itself is constructed from its constituent parts (Einsenstein, 2018). An example of this is the word *Novelists*, which is constructed of the word *Novelist*, which is itself constructed of the word *Novel*.

A NLP technique that will be explained further ahead, called stemming, retrieves the root form of a word, from which different words may derive.

However, we cannot always rely on this principle, since idiomatic phrases, for example, have meanings that are different from the sum of their parts. For situations like these, meaning could be approximated using a distributional perspective, considering the contexts in which the words appear (Einsenstein, 2018). Examples of idiomatic phrases would be *Spill the beans*, which refers to gossiping or revealing secret information, or *Torcer o nariz*, which means that you do not agree with or dislike something.

To correctly answer some questions, discourse knowledge is required, since text has an implicit relational nature. If someone writes "*I like that.*", coreference resolution is needed to understand what words like *that* or pronouns (e..g., *they*, *her*, *mine*) refer to, since their meaning was introduced in earlier messages (Jurafsky and Martin, 2021).

Sometimes, the meaning of a phrase can be ambiguous. This can be solved through lexical disambiguation, using part-of-speech tagging (e.g., decide if a word is an adjective or a verb), or word sense disambiguation (e.g., decide what the verb *make* means). Some situations may also require syntactic disambiguation (i.e., decide if the words refer to the same entity or not), or speech act interpretation (e.g., decide if a phrase is a question or an opinion) (Jurafsky and Martin, 2021). In their book, Jurafsky and Martin (2021) present a good example of ambiguity, in the sentence *I made her duck*. There are several possible interpretations, such as *I cooked a duck for her*, or *I cooked a duck belonging to her*, or *I caused her to lower her head*, or even *I turned her into a duck*.

There are other things to consider, such as the presence of Winograd schemas (i.e., one word changes the referent of a pronoun) or the use of slang or multilanguage (Einsenstein, 2018). One example, as presented by Einsenstein (2018), would be *The trophy doesn't fit into the brown suitcase because* ***it*** *is too small/large*. The final word changes what the pronoun, *it*, refers to. If it is *small*, then it would refer to the suitcase, otherwise, it would refer to the trophy.

The remainder of this section will focus on text preprocessing techniques and feature extraction or word/sub-word representation. These subjects are essential for the transformation of data into a format that machines can work with.

2.1.1 Text preprocessing

Preprocessing is used to generalize text (e.g., capitalization or lemmatization techniques) in an attempt to reduce feature redundancy (e.g., *Tree* and *tree* could be recognized as two different words, but there is no difference in meaning), and also to act as a sort of filter of significance (e.g., stop words or tokenization), which will focus on meaningful elements, while attempting to remove unnecessary or harmful characters or words. These operations are crucial for any classification operation because only discriminating information is useful for the algorithms. The sentence *I made her a duk, bro!* will be used as an example for the following preprocessing techniques for text data, as presented by Kowsari et al. (2019):

- **Tokenization**: Breaks a stream of text into meaningful elements, such as words, phrases, or symbols, called tokens. In the mentioned example, the following set of tokens would be generated: *I*, *made*, *her*, *a*, *duk*, ",", *bro*, "*!*";
- **Stop Words**: Text classification includes words that have little discriminating power (i.e., do not contribute much to the meaning of the text) for the classification algorithms, and are usually removed from the text. In the mentioned example, a stop word would be the token *a*;
- **Capitalization**: Diverse capitalization is usually handled by reducing every letter to lowercase. This will allow all the words to be projected into the same feature space. In the mentioned example, the token *I* would be reduced to *i*;
- **Slang & Abbreviation**: A common method of handling these text anomalies is to convert them into formal language. In the mentioned example, the token *bro* would be converted to *brother*;
- **Noise Removal**: Unnecessary characters, such as punctuation and special characters, can be detrimental to classification algorithms. In the mentioned example, the symbols "*,*" and "*!*" would be removed;
- **Spelling Correction**: Typographical errors (typos) are common in texts, especially in social media content. This problem can be addressed using spelling correction methods. In the mentioned example, the token *duk* would be corrected to *duck*;
- **Stemming**: Text stemming modifies words to obtain their root form, consolidating different word forms into the same feature space. In the mentioned example, the following set of tokens would be generated: *i*, ***mad***, *her*, *a*, *duk*, ",", *bro*, "*!*". This

was obtained using the Lancaster Stemmer algorithm[1]. The Porter Stemmer[2] makes no change to the original tokens;

- **Lemmatization**: Replaces the suffix of a word with a different one or removes it completely to get the form of the word as it appears in the dictionary (lemma). In the mentioned example, the following set of tokens would be generated: *i*, ***make***, *her*, *a*, *duk*, ",", *bro*, "!".

2.1.2 Representation

Data representation is a crucial step for any classification operation because it must be interpretable by the algorithm. Text data can be transformed into usable features that will represent its content in a different structure, which can be called an embedding. Some techniques for feature extraction or word/sub-word representation are presented below.

Vector semantics consists of a numeric way to represent word meaning in NLP. The main idea, made popular by Firth (1957), is that two words that occur in similar distributions, or contexts, will have similar meanings. In this sense, a word is a vector in a multidimensional semantic space derived from the distributions of the neighboring words (Jurafsky and Martin, 2021).

Figure 2.1 presents an example of the projection of word and phrase embeddings, and it confirms that words with similar meanings are nearby in space. The green color represents positive examples, the red color represents negative examples, and the blue color represents neutral examples. This projection was obtained using a technique called t-distributed Stochastic Neighbor Embedding (t-SNE) (Van der Maaten and Hinton, 2008) that allows the visualization of high-dimensional data, by mapping each embedding (in this case, a 60-dimensional representation) into a two or three-dimensional space. Briefly, the mapping is achieved by calculating a similarity measure between pairs of elements in both the high and low dimensional spaces and trying to optimize these two similarity measures.

Figure 2.1: Two dimensional projection of embeddings (Jurafsky and Martin, 2021)

The representation of words as vectors, besides allowing them to be positioned in a vector space, allows a similarity metric to be computed between them. This metric can be obtained by the cosine of the angle between the vectors, which will range from 1 (vectors pointing in the same direction) to 0 (orthogonal vectors). Since frequency values are non-negative, a cosine of -1 is not considered.

Another way of finding if two words are similar is by using a co-occurrence matrix.

[1] `https://www.nltk.org/_modules/nltk/stem/lancaster.html`
[2] `https://www.nltk.org/_modules/nltk/stem/porter.html`

There are two options for the co-occurrence matrix, term-document and term-term (Jurafsky and Martin, 2021).

In a **term-document matrix**, each document has the dimension of the vocabulary (terms). Each value will account for the number of occurrences of each term on the document. Similar documents contain similar words, and similar words appear in similar documents. Table 2.1 presents an example of a term-document matrix for four words in four Shakespeare plays.

Table 2.1: Term-document matrix for four words and documents, based on Jurafsky and Martin (2021)'s example

	As You Like It	**Twelfth Night**	**Julius Caesar**	**Henry V**
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

In a **term-term matrix**, each cell contains the number of times the row and column words co-occur in some context (for example, how often two words co-occur in the same document). Words that occur in similar contexts (defined by the counting of the words around the target, known as context words) will have similar meanings.

The following subsubsections will summarize vector semantics techniques.

N-Gram:

Instead of considering each word individually, N-gram considers sequences of words, which enabled more information to be extracted, such as context or terms that are used together, like, for example, *United States*, and that have different meanings if considered as individual words instead (Kowsari et al., 2019). An example of a 2-gram, or a bigram, based on the sentence *I made her duck*, would be the following set of tokens: *I made*, *made her*, and *her duck*.

Weighted Words:

This technique translates each document into a vector containing the frequency of the words in it, which means that the most commonly used words will dominate the representation, despite not offering much discriminating power (Kowsari et al., 2019). There are two common methods, Bag of Words (BoW) and Term Frequency - Inverse Document Frequency (TF-IDF).

In a BoW, text is represented as a non-ordered list of the words present in the document or sentence. The words are not repeated, even if they appear more than once, but the multiplicity is counted and registered in a Bag-of-Features (BoF) vector, that will contain the number of times the corresponding word for that position occurred (Kowsari et al., 2019). Based on the sentence *This house is big. That big dog lives in this house.*, its BoW would consist of the following words: *This*, *house*, *is*, *big*, *that*, *dog*, *lives*, and *in*. The corresponding BoF would consist of the following array *[2, 2, 1, 2, 1, 1, 1, 1]*.

TF-IDF is an extension of the BoW method and is usually used when the dimensions are documents. The TF part measures the relative (or absolute) frequency of each word in each document, and the IDF part attempts to devalue words such as *the*, which occur very often but have little discriminating value (Kowsari et al., 2019), by calculating the logarithm of the ratio between the number of documents and the number of documents the word appears in. When used with unseen data, the model is fitted to the samples in the training data, and will only use the words in those samples, ignoring any new occurrences.

Table 2.2 presents an example of TF-IDF applied to sentence A, "I like the colour yellow" and sentence B, "The yellow car is on the road".

Table 2.2: Example of TF-IDF applied to sentence A, "I like the colour yellow" and sentence B, "The yellow car is on the road".

Word	TF (A)	TF (B)	IDF	TF-IDF (A)	TF-IDF (B)
I	1/5	0	log(2/1)	0.06	0
like	1/5	0	log(2/1)	0.06	0
the	1/5	2/7	log(2/2)	0	0
colour	1/5	0	log(2/1)	0.06	0
yellow	1/5	1/7	log(2/2)	0	0
car	0	1/7	log(2/1)	0	0.04
is	0	1/7	log(2/1)	0	0.04
on	0	1/7	log(2/1)	0	0.04
road	0	1/7	log(2/1)	0	0.04

We can verify that words that occur often ("the", "yellow") are devalued using the TF-IDF technique, as they do not present a high discriminating value.

Word Embeddings:

In the previous methods, with the exception of the t-SNE technique used in Figure 2.1, the vectors are long and sparse (containing mostly zeros), since most words will not occur in the context of other words. Those methods face scalability issues because when there are several texts to be classified, every word in them is considered, adding to the dimension of the vectors. This can become a problem, because the vocabulary can potentially become too large, making those methods non-functional (Kowsari et al., 2019).

In the word representation domain, an embedding is a transformation of data into a short and dense vector with useful semantic properties, resulting in a large dimensionality reduction. The dimension of a dense vector will not have a clear interpretation, like when using sparse vectors, since an embedding is a mapping of each word or sentence to an N dimension vector of real numbers. Embeddings may allow for better generalization, avoiding overfitting issues, due to the smaller parameter space (Jurafsky and Martin, 2021).

Two well-known techniques are Word2Vec (Mikolov et al., 2013) and Global Vector (GloVe)(Pennington et al., 2014). These embeddings are static, meaning that the method learns one representation for each word in the vocabulary. After learning, these representations will be the same for every context (Jurafsky and Martin, 2021).

Word2Vec uses a Neural Network (NN) to find which words are more likely to occur nearby, while **GloVe** is based on the computation of co-occurrence matrices.

Word2Vec treats the word and its neighboring context as positive examples, and randomly samples other words as negative examples. Then, a simple NN is trained on a binary prediction task, to compute the probability that two words are likely to occur nearby in text. The learned classifier weights will be used as the word embeddings.

GloVe is based on the ratios of word co-occurrence probabilities over a huge corpus, using co-occurrence matrices. While in Word2Vec only the neighboring words are considered, in GloVe the co-occurrence matrices involve the whole corpus, allowing it to use the frequency of co-occurrence as relevant information (Kowsari et al., 2019). Figure 2.2 shows a visualization of the word distances, using a two-dimension projection technique, and it shows that in this kind of representation, there are syntactic and semantic relations in it, such as, in the case presented, the relation from male (left) to female (right).

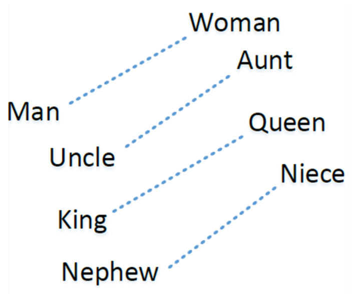


Figure 2.2: Example of GloVe for word representation (Kowsari et al., 2019)

Transformers:

Transformers (Vaswani et al., 2017) are based on encoder-decoder models, and are responsible for the latest developments in NLP. In an encoder-decoder model, the encoder generates a contextual embedding representation for each word. The output, given by the decoder, is a complex function of the entire input sequence.

A transformer contains a self-attention layer applied to it, avoiding the usage of recurrent connections, while still taking context into account. Self-attention layers compute the relevancy of each element in relation to the other elements in the sequence, allowing transformers to model how words over long distances are relevant for the processing of the current word. These models have proven to be more parallelizable and quicker to train than previous NN models since they calculate the element's embeddings concurrently.

Transformers make it possible to use a pre-trained model that can be fine-tuned for a variety of tasks. Fine-tuning consists mostly of adding and training one output layer to the model, according to the desired task (Devlin et al., 2018).

Two known types of transformers are the Generative Pre-trained Transformer (GPT) (Radford et al., 2018) and Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018). Both fine-tune a pre-trained model for the desired task. In the pre-training stage, the initial parameters of the model are learned, and in the fine-tuning stage, the parameters are adapted to the objective.

GPT uses a left-to-right architecture, meaning that the word (or sentence) representation will only consider context on the left side of it. GPT-2(Radford et al., 2019) is a GPT model trained with English data from the social media Reddit[3], and its task is the prediction of the next word in a sequence.

GPT-3 (Brown et al., 2020) is an upgraded version of GPT-2, trained with a larger number of parameters and on a larger combined dataset. Access to GPT-3 is currently limited through an Application Programming Interface (API)[4].

[3] `https://www.reddit.com/`
[4] `https://beta.openai.com/overview`

BERT is a pre-trained encoder that learns word (or sentence) representations that incorporate context from left-to-right and right-to-left directions.

BERT is pre-trained using the following strategies: Masked Language Model (MLM) and Next Sequence Prediction (NSP).

MLM randomly masks some of the tokens in the input and attempts to predict the masked word based only on its context. This strategy enables the concatenation of both left-to-right and right-to-left representations, resulting in a deep bidirectional transformer.

NSP jointly pre-trains text-pair representations. The input is a pair of sentences, and it attempts to predict if the second sentence is the actual next sentence in the original document or not. This strategy allows for a better understanding of sentence relationships.

Both GPT and BERT use special representation tokens, such as [SEP] and [CLS]. However, for the former model, these tokens are only generated during fine-tuning, while for the latter model, they are learned during pre-training. The [SEP] token is used as a sentence separator, but it is the [CLS] token that is the most interesting, since it contains an aggregate representation of the whole sentence. This token is a symbol for classification tasks, as is the case of SA.

2.2 Machine Learning

"Machine Learning is a unified algorithmic framework designed to identify computational models that accurately describe empirical data and the phenomena underlying it, with little or no human involvement" (Watt et al., 2020). In essence, it is "the science of programming computers so that they can learn from data" (Géron, 2019). Today, ML can be applied to several domains, such as search engines, medical diagnoses, loan applications, and many more.

There are four types of ML, supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning (Kotsiantis, 2007, Zhu and Goldberg, 2009). The first three differ in the amount of labeled data in the training dataset. In supervised learning, all the data is labeled (assigned a class), in semi-supervised learning, only some of the data is labeled, and there are many unlabeled data, and in unsupervised learning, all of the data is unlabeled. In reinforcement learning, the learner is not told which actions to take, it must discover which ones yield the best reward, by trying them out. The training information is provided by the environment in the form of a scalar reinforcement signal (i.e., a measure of how well the system is operating).

Unsupervised learning applies to tasks such as clustering or outlier detection, for example. In these algorithms, researchers hope to discover unknown and useful classes for their data. Supervised learning applies to tasks such as classification or regression, for example, where the classes belong to a predefined set. Semi-supervised learning designs algorithms (e.g., self-training, graph-based methods) that take advantage of a combination of labeled and unlabeled data.

Since SA is a (text) classification operation, and the main goal of this dissertation, supervised learning is the most suited type of ML, and some classification models will be trained to distinguish between different classes, representing sentiments or emotions.

A ML project life cycle includes the following stages: data collection and preparation, feature engineering, model training, and model evaluation (Burkov, 2020), which will be

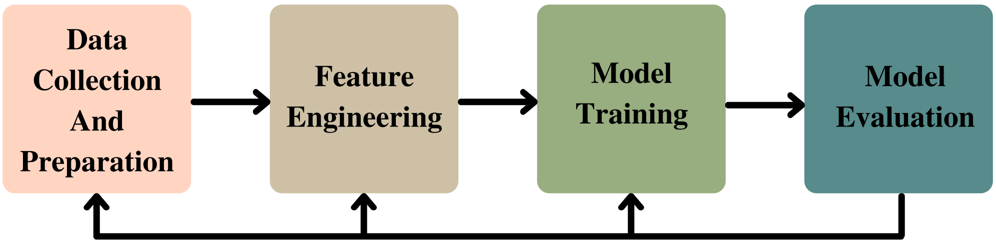


Figure 2.3: Part of a ML project life cycle, based on Burkov (2020)'s example

the focus of this section. These stages are illustrated in Figure 2.3, where it is possible to see that from the evaluation of a model, three stages can be modified for better results, data collection and preparation, feature engineering, and model training.

2.2.1 Data collection and preparation

The success of a classification model is dependent on the quality of the data available for training. If there is a large and diversified collection of samples, the better the model can perform its tasks, since it will be exposed to a varied collection of data, giving it more learning experience (Watt et al., 2020).

The data collection is usually split into three sets, the training dataset, the validation dataset, and the testing (or holdout) dataset. The first is used to train the model, and each sample is assumed to belong to a predefined class (the target label). The second is used to estimate the performance on unseen data, and compare it with the performance of different algorithms or configurations. The third is used to assess the performance of the best model, according to the validation dataset.

The analysis of the errors during the training set and either the validation or holdout sets can be helpful to determine if a model has a complexity fitting of the problem, translating into a good bias-variance tradeoff. If this happens, then the model is in the zone of solutions. An example of a simple error metric would be how many samples were badly classified.

Figure 2.4 allows for better visualization of how the error analysis can be helpful. If both the training and holdout errors are large, and the latter error is not stagnating, then the model's complexity should be increased, and the scenario is of underfitting. Once both values are low and the holdout error starts increasing at a fast pace, while the training error is stagnating, then the model's complexity should be simplified, and the scenario is of overfitting.

The bias-variance tradeoff and the concepts of overfitting and underfitting will be explained further ahead when model training is addressed.

2.2.2 Feature engineering

Features are a representation of a property of the data used by the model, such as the color or size of a fruit. The design of good features can be dependent on the classification goal. Features that are discriminating against certain classes may be useless for a different application. Color may be a relevant feature for distinguishing between blueberries and strawberries, but not between carrots and oranges, for example. In the context of this

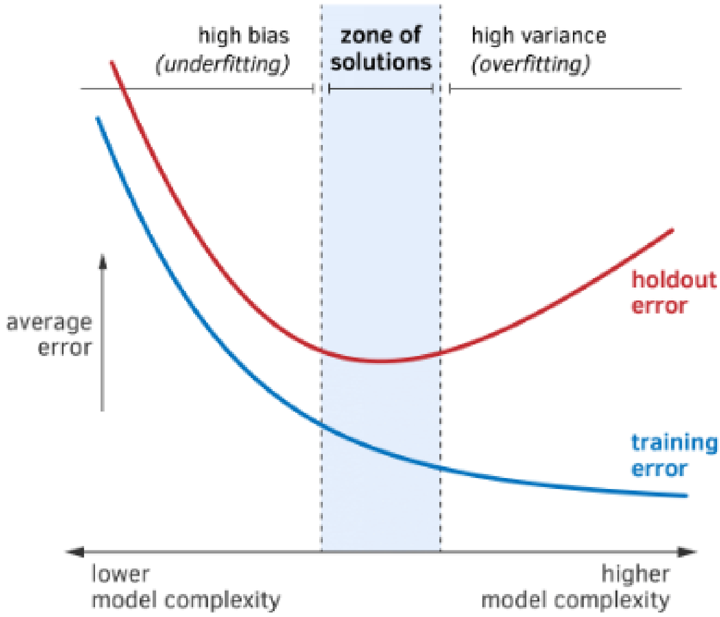


Figure 2.4: Training error vs. holdout error (Burkov, 2020)

work, we will be using word or sequence representations. In section 2.1.2 some techniques that include feature representation were presented, so the topic will not be further explored in this section.

2.2.3 Model training

This subsection will focus on hyperparameter tuning and the bias-variance tradeoff. The classification algorithms will be presented later, in subsection 2.2.5.

Hyperparameter tuning

Each classification algorithm has a unique set of parameters, hyperparameters, that control the inner workings of the algorithm, such as the number and type of layers, and the number of neurons per layer, in the case of neural networks. Hyperparameter tuning should be performed on the validation dataset, so it does not overlap the training set. The definition of the classification algorithm can have a big effect on its performance, so it is important to choose an appropriate set of hyperparameters. To accomplish this, one can use several techniques, such as:

- Grid search: Tries a set of defined values, evaluates the model for each value, and chooses the setting that maximizes the performance (Einsenstein, 2018);
- Random search: Does not use a defined set of values, it randomly samples values from a defined statistical distribution for each hyperparameter. A grid search can be performed in the region of the highest value found (Burkov, 2020);
- K-fold Cross-validation: Usually used when the size of the dataset is small. Cross-validation consists of randomly splitting the training dataset into training and testing sets. The model will train on the split training set and will be evaluated on the testing set. K-fold cross-validation consists of repeating this process K times, with different

randomly selected training and validation, or "Dev", sets. The evaluation of the model will be computed in the original testing set (Jurafsky and Martin, 2021). An extra fold can be used for a grid search. Figure 2.5 may help with the visualization of the k-folds, presenting an example for 10-fold cross-validation;

- Genetic Algorithm (GA) Search Cross-Validation[5]: Applies cross-validated evolutionary optimization over hyperparameters, using simple algorithms which can be accessed through the Distributed Evolutionary Algorithms in Python (DEAP) package;

- Population-Based Training (PBT) (Jaderberg et al., 2017): Optimization evolutionary algorithm that jointly optimizes a population of models and their hyperparameters. PBT has shown improvements in relation to other options, such as grid search, making it an interesting choice for hyperparameter tuning.

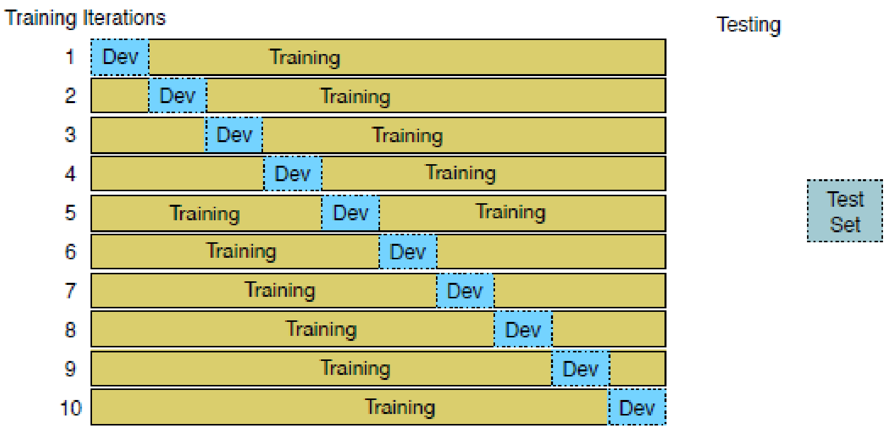


Figure 2.5: 10-fold cross-validation (Jurafsky and Martin, 2021)

The fine-tuning of the hyperparameters is usually done when a good bias-variance tradeoff is achieved.

Bias-variance tradeoff

Bias and variance are values that control if a model is underfitting, overfitting, or if it has good generalization. Bias can be thought of as a deviation from the regression line, while variance can be thought of as the regression line being a too close fit to the data. Underfitting is associated with high bias. This usually means the model is too simple for the data, or the features are not discriminative enough. In this scenario, the model makes several mistakes on the training data. Overfitting is associated with high variance. This usually means that the model is too complex for the data, or there are too many features and few training samples. In this scenario, the model makes several mistakes on the testing or validation data (Burkov, 2020).

Figure 2.6 provides good examples of the bias-variance tradeoff. In the figure to the left, there is an underfitting scenario, and it is possible to verify that the regression line does not follow the flow (the ups and downs) of the training examples, meaning that the model oversimplifies the data.

[5] `https://sklearn-genetic-opt.readthedocs.io/en/0.4.0/api/gasearchcv.html`

In the figure to the right, there is an overfitting scenario, and it is possible to verify that the regression line follows the training examples almost perfectly, meaning that the model is too complex for the data.

In the middle figure, there is a good tradeoff scenario, where both the bias and the variance are low. To reach this kind of solution, the complexity of the model must be increased (in case of underfitting) or decreased (in case of overfitting). Once a good fit is reached, the hyperparameters can be fine-tuned to optimize the model (Burkov, 2020).

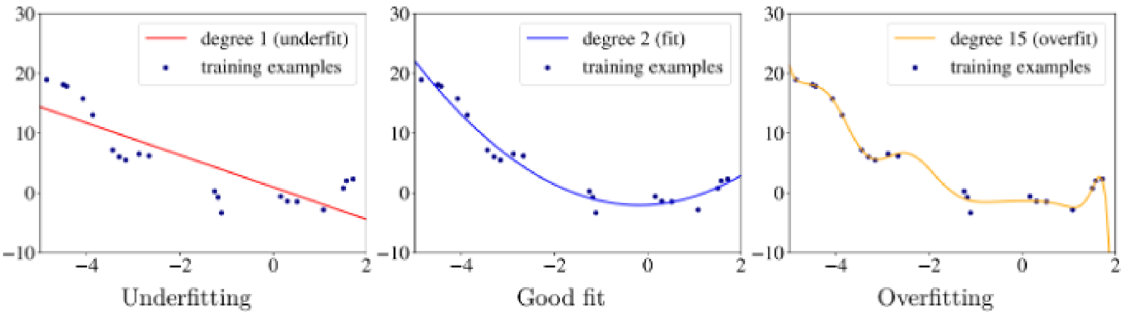


Figure 2.6: Examples of underfitting, good fit, and overfitting (Burkov, 2020).

There is a bias-variance tradeoff since by trying to reduce one, you increase the other.

2.2.4 Model evaluation

It is important to be able to quantify the performance of a model and establish a baseline value as a reference point so that we can say that the chosen approach performs better than the baseline model, or other solutions. Several evaluation metrics can be used to evaluate the performance of a classification model, and they are usually based on the interpretation of a confusion matrix, represented in Figure 2.7.

Figure 2.7 introduces the concepts of True Positive (TP), False Positive (FP), False Negative (FN), and True Negative (TN). Taking as an example the task of spam detection in emails, these concepts would refer to:

- TP: Spam being classified as spam;
- FP: Non-spam being classified as spam;
- TN: Non-spam being classified as non-spam;
- FN: Spam being classified as non-spam.

Essentially, TP and TN occur when the predicted and the true classes are the same. Otherwise, in the FP and FN cases, the true class is the opposite of the predicted class.

Applying this to the (simplified) purpose of this dissertation (i.e., to determine when, in a conversation between a chatbot and a client, the client's sentiment is so negative that a human should replace the conversational agent), a TP would be when the prediction of the negative sentiment reflects the real sentiment of the client (i.e., a negative sentiment has been correctly classified). A FP would be when the prediction is that the sentiment is negative, but in fact, it is not (i.e., a non-negative sentiment has been incorrectly classified). A FN would be when the prediction is a non-negative sentiment, but the client is reflecting

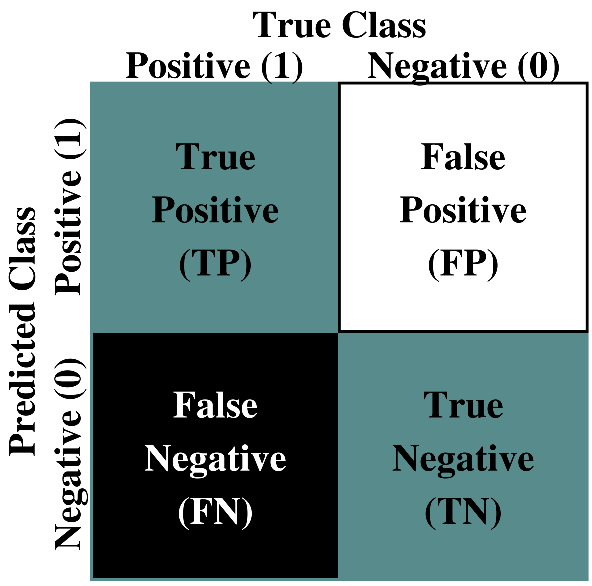


Figure 2.7: Confusion Matrix

a negative sentiment in reality (i.e., a negative sentiment is incorrectly classified). A TN would be when the prediction of a non-negative sentiment reflects the real sentiment of the client (i.e., a non-negative sentiment has been correctly classified).

It is important to understand which metric better applies to each problem, which is dependent on the data and the context of the problem itself. If more importance is given to reducing the FN predictions, then a valid metric to choose would be the recall metric. This is a common metric for clinical problems since it is of high importance to avoid sending home a patient that is sick (a false negative situation). However, if the problem is about distinguishing between real and fake images or if it is about spam detection, as presented earlier, a valid metric could be the precision metric, since it is more important to avoid wrongly placing a legitimate message in our spam folder, or to classify a real image as false (a false positive situation) (Burkov, 2020).

The evaluation metrics should be calculated for each dataset. If a model performs well on the validation and testing datasets, then, depending on the chosen evaluation metric, one could affirm that the model generalizes well, meaning it is capable of classifying unseen data correctly, within the given performance value. It may also be important to find out which evaluation metric should be maximized in comparison with the others, since sometimes models may not vary much between the evaluation results. Despite this, the choice of the final model must be justified, and so this analysis may be relevant. In a binary situation, this can be achieved by considering the problem at hand and figuring out which would be more important, either to reduce the number of FP or to reduce the number of FN.

Accuracy (equation 2.1) consists of the ratio of correct predictions over all predictions.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

Accuracy may not be a good performance metric for imbalanced datasets, since one class may consist of the majority of the samples, which will skew the accuracy results. If there are 90 samples of class A and 10 samples of class B, and the model is badly trained due to oversampling, it will classify all samples as class A, which will return an accuracy of 90%, when in reality the model's performance is low.

Precision (equation 2.2) consists of the ratio of correctly predicted positives to all positive predictions made (i.e., among the predicted positives, how many were real positives).

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

Recall (equation 2.3) consists of the ratio of known/real positive classifications that are correctly predicted (i.e., among the real positives, how many were correctly predicted). This metric can also be called sensitivity or True Positive Rate (TPR).

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

Specificity (equation 2.4), also known as True Negative Rate (TNR), consists of the ratio of known/real negative classifications that are correctly predicted, as given by equation 2.4 (i.e., among the real negatives, how many were correctly predicted).

$$Specificity = \frac{TN}{TN + FP} \tag{2.4}$$

F-score (equations 2.5 and 2.6) consists of the harmonic mean of precision and recall. The general approach (equation 2.5) can be referred to as the F_β-score. The β value will modify the weight given to the recall value, in relation to the precision value.

$$F - Score = (1 + \beta^2) \times \frac{precision \times recall}{(\beta^2 \times precision) + recall} \tag{2.5}$$

The F1-score (equation 2.6), where the value of β is defined as 1, is a common application of the F-Score metric, and is used when the same weight is given to both precision and recall (Burkov, 2020).

$$F1 - Score = 2 \times \frac{precision \times recall}{precision + recall} \tag{2.6}$$

The last metric to be presented is the Area Under the ROC Curve (AUC). The Receiver Operating Characteristic (ROC) curve is a plot of the TPR/Recall vs. the False Positive Rate (FPR), which consists of the ratio of wrong negative classifications to all real negative samples (i.e., among the real negatives, how many were wrongly predicted). The FPR is obtained by the equation 2.7, but can also be calculated as seen in equation 2.8, where the second term is the specificity.

$$F.P.R. = \frac{FP}{TN + FP} \tag{2.7}$$

$$F.P.R. = 1 - \frac{TN}{TN + FP} \tag{2.8}$$

The different points on the ROC curve correspond to different decision thresholds, depending on how the threshold is defined. To better explain this, Mandrekar (2010) proposes that multiple decision thresholds are possible for the classification of a sample, based on a given feature. He explains that if a feature value of 4 or above indicates a given class, then the FPR and the TPR would be different than if the limit were defined as a value of 3 or above, meaning that both the FPR and the TPR are specific to the selected decision threshold. A ROC curve is obtained by combining the FPR and TPR values for different decision thresholds.

The larger the AUC, the better the performance of the model, since a larger area means that the TPR is large while the FPR is small. Figure 2.8 should provide better visualization of this relation. If the AUC value is of 0.5, then the ROC curve falls on the diagonal line that connects the coordinates (0,0) and (1,1). What this means is that the model performs as well as a random classifier (i.e., as if the labels were given by pure chance). The goal would be for the area under the curve to be as close to 1 as possible.

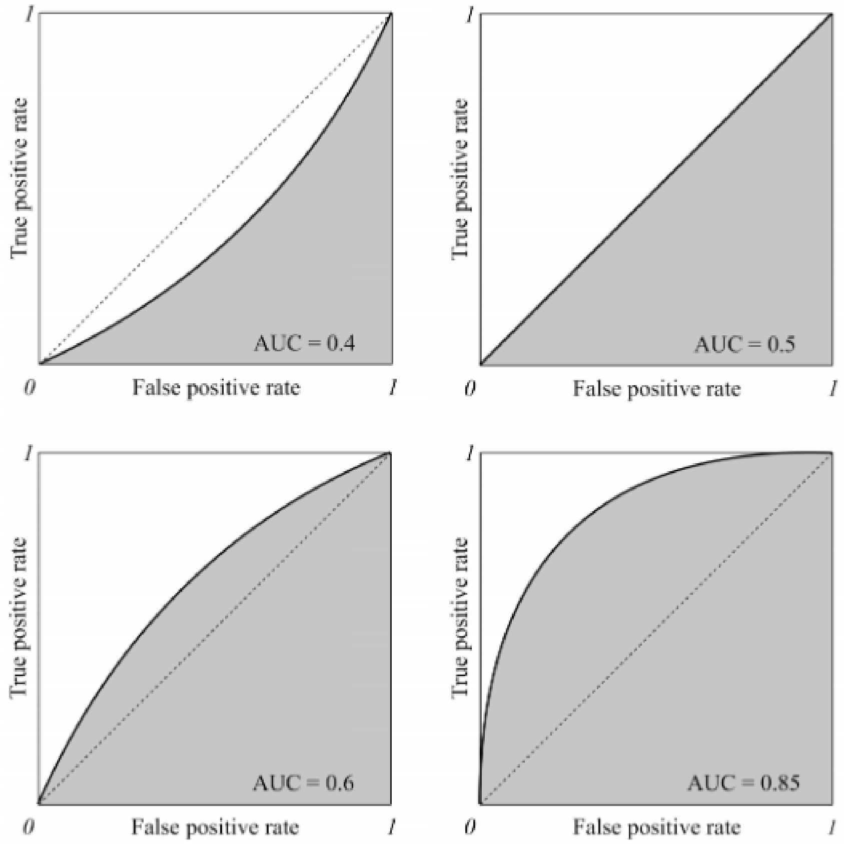


Figure 2.8: Area under the ROC curve (Burkov, 2020)

2.2.5 Classification algorithms

As the "No Free Lunch" theorem implies, there is no algorithm that is the best-performing for all problems (Wolpert and Macready, 1997). The best algorithm is dependent on the context, objectives, and domain of the problem. Properties such as interpretability, training and classification speed, tolerance for missing values or noise, scalability, size and nonlinearity of the data, and the available memory of the machine or server should be considered (Burkov, 2020). On the other hand, Domingos (2015) proposes a *Master Algorithm*, which is the combination of contemporary ML paradigms into an algorithm capable of fine-tuning itself, which would allow it to reach a perfect understanding of how the world and the people in it work. As the author states, this is yet a very distant accomplishment and can be referred to as the promise of ML.

In this subsection, some of the possible algorithms to be applied to the problem will be summarized.

Probabilistic models:

Probabilistic classification algorithms use statistical inference to find the best class for a given sample. These algorithms output the probability of the example being a member of each of the possible classes and assign the class with the highest probability (Deng et al., 2014).

Examples of this type of model are presented below. The last two, Hidden Markov Model and Conditional Random Field, classify sequences instead of just words.

- **Naive Bayes Classifier:**

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \tag{2.9}$$

  This classifier is based on the Bayes theorem, represented in equation 2.9. This theorem gives us the probability of A given B (probability of an event, A, given the sample's characteristics, B). The operation is based on the prior probability (probability of the event, A) and the conditional probabilities of each feature (P(B|A), representing the effect each characteristic has on the outcome).

  This method is referred to as "naive" because it assumes conditional independence between features (Jurafsky and Martin, 2021).

- **Logistic Regression:**

  Classifiers that use a linear combination of the inputs to perform a classification decision are called linear classifiers (Jurafsky and Martin, 2021). Logistic regression is a simple linear classifier that distinguishes between two classes, but it can be adapted to multiclass problems. This algorithm requires independence of data (Kowsari et al., 2019).

- **Hidden Markov Model (HMM)s:**

  HMM assign a label to each word in the input so that the output sequence has the same length as the input sequence. These models compute the probability distribution over possible sequences of labels and choose the best label sequence (Jurafsky and Martin, 2021).

- **Conditional Random Field (CRF):**

  CRF (Lafferty et al., 2001) is based on the logistic regression classifier and is best suited to tasks where contextual information has an effect on the current state. Hence, instead of assigning a class to a single observation, it produces a global probability for the whole sequence, taking context into consideration. CRF turns a sequence of words into a linear chain (i.e., a directed graph where each word is a node), and considers not only the probability of a label for each word but also the transitioning probability between labels. Taking this into consideration, the CRF model returns the prediction from the most probable path (Sutton et al., 2012). To account for unseen data, this classifier can be set to generate all the possible combinations between words and labels, and all the possible transitions between labels.

Instance-based models:

A classification process is usually a two-phase approach that is cleanly separated between constructing a model from training instances and using it to label unseen instances. In instance-based approaches, there is no explicit training phase. Instance-based or learning algorithms require a similarity or distance function to identify a locality around the test sample and use it to classify the current sample (Aggarwal, 2014b).

- **K-Nearest Neighbors (K-NN):**

  K-NN finds the K closest examples to the testing sample and classifies it based on the most common class in the neighborhood (Kowsari et al., 2019).

Rule-based models:

Rule-based models are highly interpretable systems and usually involve a large database of hand-written disambiguation rules, which will specify the class of a given sample (Jurafsky and Martin, 2021).

- **Decision Trees:**

  Decision trees have a hierarchical structure where each terminal node holds a class label, each internal node denotes a test on a feature, and each branch represents the outcome of that test. The hierarchy of the features is computed based on relevance metrics, such as information gain, which allow these models to automatically perform feature selection (Jurafsky and Martin, 2021).

- **Dictionary-based:**

  A dictionary is used to assign each word a list of potential classes, and then the disambiguation rules are used to narrow down the list to a single class for each word (Jurafsky and Martin, 2021).

Connectionist models:

Connectionist models can be deep or shallow NNs. A NN is a combination of basic units, neurons, arranged in layers, that are connected to form a network (Aggarwal, 2014a).

There are three types of layers, the input layer, the hidden layer(s), and the output layer. Shallow networks have a single hidden layer, while deep networks can have several. Only deep algorithms will be summarized.

Learning is achieved by computing weights between the neurons. First, the input is propagated forward, and the error is calculated when the output layer is reached. Then, a backpropagation algorithm is used to propagate the error backward and update the weights of the neurons (Aggarwal, 2014a).

- **Multilayer Perceptron (MLP)**

  A perceptron is a shallow NN with a binary output and a linear activation function. The latter means that such models could not solve non-linear problems (Jurafsky and Martin, 2021).

A MLP is a NN consisting of multiple layers of perceptrons (at least three layers), that can use non-linear activation functions, allowing it to distinguish data that is not linearly separable.

- **Graph Attention Network (GAT)**

  GAT is a type of NN used to process data that is represented by a graph structure. Attention mechanisms allow the model to deal with variable-sized input, focusing only on the most relevant parts of the input. When such a mechanism is used to represent a sequence, it is referred to as *self-attention* (Veličković et al., 2017).

  The attention-based architecture allows the computation of the hidden layers in the graph, by attending over its neighbors, which allows for parallelization across node-neighbor pairs. The self-attention on the nodes will compute the attention coefficients, indicating the importance of the features of one node to its neighbor (Veličković et al., 2017).

- **Recurrent Neural Network (RNN)**

  RNN contain a cycle within its network connections, allowing these models to consider context since the previous value will be used to compute the current value (Jurafsky and Martin, 2021).

  The sequential nature of these models makes it hard to perform computation in parallel. (Jurafsky and Martin, 2021)

- **Long Short-Term Memory (LSTM)**

  A LSTM model is a type of RNN that uses multiple gates to regulate the amount of information allowed into each neuron (Kowsari et al., 2019).

  The addition of gates allows LSTMs to handle more distant information than RNNs. However, passing information through many series of recurrent connections results in information loss and difficulties in training (Jurafsky and Martin, 2021).

- **Convolutional Neural Network (CNN)**

  The architecture of CNN is composed of convolutional layers, pooling layers, (typically) fully connected layers, and an output layer. The convolution layers provide filters on the input, while the pooling layers reduce the size of the output from one layer to the next, reducing the computational complexity (Kowsari et al., 2019).

- **Transformers**

  These models were already summarized in section 2.1.2 and differentiate themselves from the rest due to their composition of stacks of encoder and decoder blocks, which are multilayer networks made by combining feedforward layers and self-attention layers. Self-attention allows a network to directly extract and use information from arbitrarily large contexts without the need to pass it through intermediate recurrent connections (Jurafsky and Martin, 2021).

  Some interesting approaches using transformers have been developed and will be experimented with in this dissertation. Hence, they will be presented in more detail. These approaches are Few-Shot Learning (Few-SL) (Miller et al., 2000) and Zero-Shot Learning (Zero-SL) (Chang et al., 2008).

  – **Few-Shot Learning**

    Few-SL aims to train classifiers given only a few labeled examples of each class. This is better employed using large models, which can adapt to many contexts and generalize with just a few examples.

Examples of these models are GPT-3, already presented, GPT-Neo[6], GPT-J, and Meta-OPT[7].

GPT-Neo is a replication of the GPT-3 architecture, and was trained on the Pile[8], a large (825GB) dataset comprised of 22 smaller datasets containing data from many disparate domains, including books, webpages, chat logs, and even code. 97.4% of this dataset is in the English language.

Meta-OPT was also trained to roughly match the performance and sizes of the GPT-3 class of models, and it was trained on multi-language data (predominantly English) comprised of some data from the Pile dataset, and data from four other datasets, with emphasis on human-generated text.

GPT-J[9] is a GPT-2-like model, trained on the Pile dataset. This model seems to perform better than GPT-Neo, according to Xu et al. (2022), however, this was tested in completion of code, not text generation.

Regarding the data these models were trained on, when multi-language, as is the case for all (even if the samples are predominantly English), sometimes this can result in lower performances than using models trained on a single language.

In the Few-SL approach, the models will perform text completion based on the examples received. If you feed the model examples such as "I am unhappy with this service |sentiment| Negative", "Thank you for helping me |sentiment| Positive", and "Hi, I need some information about payments |sentiment| Neutral", and then give it as input the sentence "I want to speak with an assistant |sentiment| ", the model will, ideally, generate one of the sentiment labels provided in the examples, following the patterns set by them. However, this is not always the case, and the model can generate text that is not desirable (e.g., "information", "|sentiment|", or even just symbols). Hence, this kind of approach requires some processing of the text completion to evaluate if it is valid.

Besides the type of examples presented, there are other options as how to ask the model to generate what you want. Mi et al. (2022) experimented with including a task description, composed of a definition, a constraint, and a prompt. An example of these, provided by the authors, could be:

* Definition: Predict the intent of the input query. Intent is the main topic or purpose of a query.;
* Constraint: Model needs to select the most suitable intent from: {candidate labels + label descriptions};
* Prompt: What is the intent of the given query?

This was presented to keep in mind that the way the examples and the input is presented can affect how the model will generate the data.

– **Zero-SL**

Zero-SL has the most restrictive premise because there are no training samples for model training. Chang et al. (2008) claims that a concept or label is often sufficient for classification, and is based on the use of a source of world knowledge to analyse sentiments and sentences (in our application) from a semantic point of view, allowing them to be compared to perform classification. This requires large models, similarly to the Few-SL approach, but not many are trained for zero-shot classification.

[6] `https://huggingface.co/EleutherAI/gpt-neo-2.7B`
[7] `https://huggingface.co/facebook/opt-2.7b`
[8] `https://pile.eleuther.ai/`
[9] `https://huggingface.co/docs/transformers/model_doc/gptj`

BART is a large model that is particularly effective for text generation and text comprehension (Lewis et al., 2019) that allows for Zero-Shot classification. This model is available on the Hugging Face website[10], and it was trained on the Multi-Genre Natural Language Inference (MultiNLI) corpus (Williams et al., 2018), a crowd-sourced collection of English sentence pairs (a premise and a hypothesis) annotated with textual entailment information (entailment, neutral, or contradiction).

In Zero-SL, you send as input a premise (i.e., the sentence to be labeled) and a hypothesis (i.e., the possible label), and this model will return the probabilities of textual entailment regarding that hypothesis. For better understanding, take as an example the premise "My TV is not working, you better fix it now!". Some hypotheses could be "Happy", "Unhappy", "Problem", and "Praise". Ideally, the model would return high entailment for "Unhappy" and "Problem", and low entailment (or high contradiction) for the remaining hypotheses. In fact, testing this example in the Hugging Face Inference API[11], and allowing more than one hypothesis to be true, the model returns the expected probabilities.

Since in this approach we do not work with text generation, it does not face the same problem as the Few-SL approach. Every output will be valid, meaning every sample will have a corresponding prediction.

Ensemble models:

Ensemble models combine weak learners, which perform slightly better than random guessing, and they will converge into a final strong learner. The new sample is classified based on voting techniques (Kowsari et al., 2019).

- **Boosting:**

  Boosting consists of iteratively training weak classifiers. Freund et al. (1999) developed AdaBoost, which is an adaptive boosting algorithm, in the sense that subsequent weak learners are tweaked, so that the weights of incorrectly classified examples are increased and the weak learner is forced to focus on the hard examples.

- **Bagging:**

  A bagging algorithm generates different uniform samples from the training data, trains a classifier for each dataset, and in the end, applies a voting technique (Kowsari et al., 2019).

- **Random Forest:**

  Random Forest is an example of bagging that combines the output from multiple decision trees. The application of bagging allows each tree to contain different samples from the dataset, resulting in different trees. Furthermore, this classifier allows each tree to consider only a random subset of features, resulting in a low correlation between trees, which creates a model that is more robust and less prone to error (Breiman, 2001).

[10] `https://huggingface.co/facebook/bart-large-mnli`

[11] `https://huggingface.co/facebook/bart-large-mnli?candidateLabels=happy%2Cunhappy%2Cproblem%2Cpraise&multiClass=true&text=My+TV+is+not+working%2C+you+better+fix+it+now%21`

Hyperplane-based learning model:

The Support Vector Machine (SVM) is a classifier that does not completely fit any of the presented categories, hence we defined its own category. SVMs were originally developed for binary classification tasks but were later adapted into a nonlinear formulation (Kowsari et al., 2019). These models use a kernel function (e.g., linear, polynomial) to transform the data and try to find a hyperplane that splits the data, and attempt to maximize the hyperplane's margin, allowing for higher generalization (Aggarwal, 2014a).

It could be partially considered instance-based, since during the training stage, the SVM will estimate the hyperplane from the positions of the instances in the training dataset, and then maximize the margin. However, during evaluation, the classifier will not compare the new sample with the existing instances, but against the hyperplane, which is where this model differs from the regular instance-based models. Hence, we classify it as hyperplane-based.

2.3 Sentiment Analysis

SA aims to classify the opinions and emotions expressed by humans, for example, on social media posts, and can be useful, for example, in the following applications:

- Recommender systems: recommend items to the user based on their opinions (Asani et al., 2021);
- Editorial sites: create summaries of people's experiences and opinions extracted from reviews (Roy et al., 2020);
- Information extraction: systems could flag statements and queries regarding opinions rather than facts (Wiebe and Riloff, 2011);
- Market intelligence: allows businesses to find consumer opinions automatically (Rambocas and Gama, 2013);
- Political studies: prediction of election results and the study of political standpoints (Sharma and Moh, 2016);
- Advertisement placement: place advertisements in social media content if users like one product, for example (Adibi et al., 2018);
- Dialogue systems: allow the system to adjust to the user's sentiment (Rinaldi et al., 2017).

Social media posts are usually informal and contain slang, irony, sarcasm, abbreviations, and emoticons, which are hard for a computer to understand (Pereira, 2021). These difficulties are common to text classification problems, but there are challenges specific to this domain, as mentioned by Einsenstein (2018):

- Targeted SA: mixed overall sentiment (positive towards one entity, negative towards another). Requires identifying the entities and linking them to specific sentiment words;

- Emotion classification: the identification of emotions (e.g., anger, happiness, surprise) instead of sentiment (e.g., negative, positive) can highly increase the number of possible classes, making annotation more difficult and resulting in low accuracy;
- Domain adaptation: in different domains, the same word may express different sentiments (e.g., terrifying has positive sentiment if the domain is horror movies, but in most domains, it would bear negative sentiment);
- Sentiment shifters: words/phrases that can change sentiment orientation (negation, sarcasm).

SA can be performed using deep learning approaches, such as the ones presented in section 2.2, but there are other options, such as lexicon-based approaches.

In lexicon-based classification, dictionaries of words annotated with their sentiment polarity are used, and the number of words of each label that is present in the input is calculated. The final label is the one with the higher count (Einsenstein, 2018). The labeled dictionaries can be created manually or automatically, using, for example, seed words to expand the list of words. What this later creation process does is to use a set of words to search for synonyms or antonyms in a dictionary, adding them to the lexicon with the corresponding sentiment polarity.

Examples of available sentiment lexicons are SentiWordNet (Baccianella et al., 2010), AFINN (Nielsen, 2011), or Opinion Words (Hu and Liu, 2004). SentiWordNet is an example of a sentiment dictionary, built for supporting sentiment classification and opinion mining applications, and created using seed words. Table 2.3 presents the top five positive and negative words (or expressions) found in this lexicon. AFINN is a list of words and phrases manually rated for valence (pleasantness of a stimulus), with values that range from *-5* (e.g., *prick* or *twat*) to *5* (e.g., *hurrah* or *outstanding*). Opinion Words is a dictionary of opinion words, extracted from reviews, and their sentiment polarity (e.g., *arrogant* or *cranky* as negative words, and *mercy* or *relief* as positive words).

Table 2.3: Top 5 ranked positive and negative words or expressions in SentiWordNet.

Rank	**Positive**	**Negative**
1	good, goodness	abject
2	better off	deplorable, distressing, lamentable, pitiful, sad, sorry
3	divine, elysian, inspired	bad, unfit, unsound
4	good enough	scrimy
5	solid	cheapjack, shoddy, tawdry

A corpus-based approach can be used as opposed to the dictionary-based approach, where the set of words is used to search for new words in a corpus, through context-specific orientations (which can sometimes be composed of multi-word expressions associated with a specific sentiment) (Pereira, 2021).

According to Taboada et al. (2011), supervised classifiers (i.e., based on the use of labeled data) are capable of performing very well in the domain they are trained on, but the performance drops (almost to pure chance) if the same classifier is used on a different domain. The authors also stress that valence shifters should be taken into account (e.g., intensifiers, downtoners, and negation), to avoid misclassification, since the presence of negation, for example, can completely shift the text's sentiment polarity (e.g., *not good* vs. *good*.

There is a lexicon and rule-based sentiment analysis tool, Valence Aware Dictionary and Sentiment Reasoner (VADER)(Hutto and Gilbert, 2014), that is specifically attuned to sentiments expressed in microblog-like contexts. Some translated solutions are available, such as Léxico para Inferência Adaptada (LEIA)(Almeida, 2018). Otherwise, VADER offers two approaches to its application to other languages: either the samples are translated into English and the VADER algorithm is applied to it (Costa et al., 2021), or the lexicon is manually edited, along with files containing negation examples or special meanings for the language (Amin et al., 2019).

In the following sub-sections we will go over four substasks related to SA, and we will discuss models for one of those substasks, Emotion Classification.

2.3.1 Subtasks

According to Pereira (2021), there are four subtasks related to the task of SA:

1. **Polarity:** The polarity classification subtask consists of identifying a degree of positivity or negativity in a given input. Polarity can consist of only positive and negative labels, or also include a neutral class;
2. **Subjectivity:** Sentences can be classified as objective or subjective, and only the latter are used for sentiment classification;
3. **Emotion:** Emotion classification may refer to feelings such as sadness or fear, and will be explored further in sub-section 2.3.2;
4. **Aspect-based:** At the aspect level, the sentiment regarding specific aspects of the entities is classified. This demands the identification of the entities and their aspects first.

In the context of this dissertation, the most focused subtask is polarity, but emotion can be considered for future work. Subjectivity could be helpful to filter out sentences with no sentiment (objective utterances), but since *Neutral* can be a polarity label, this subtask may not be required. The aspect-based subtask may be the least relevant since this work will not focus on sentiment regarding entities' specificities (e.g., such as price or service).

2.3.2 Emotion models

SA is usually framed in terms of positive and negative categories Einsenstein (2018), but it can also include a neutral category. Emotion Analysis uses a more broad spectrum of emotions, and there have been efforts to define which emotions should be taken into consideration:

- Plutchik (1984) argues that there are eight basic emotions: fear, anger, joy, sadness, disgust, acceptance, surprise, anticipation (or curiosity), and proposes a wheel of emotions, where each of these emotions is represented with different levels of arousal (maximum arousal at the top and milder versions at the bottom). This representation can be seen in Figure 2.9;

- Ekman (1992) proposes six universal emotions: happiness, surprise, fear, sadness, anger, and contempt;
- Pearl and Steyvers (2010) consider complex emotions: politeness, rudeness, embarrassment, formality, persuasion, deception, confidence, and disbelief;
- Cowen and Keltner (2017) identified 27 types of emotions elicited by emotionally evocative short videos.
- Russell (1980) proposed a non-categorical circumplex model, that suggests that emotions are distributed in a two-dimensional (or three-dimensional, if the Dominance dimension is considered), containing Arousal and Valence dimensions. Valence represents the pleasantness of the stimulus, arousal represents the intensity of emotion provoked by the stimulus, and dominance represents the degree of control exerted by the stimulus.

Emotion models make text classification harder due to there being more classes, which makes even human annotators frequently disagree with each other regarding the chosen class (Einsenstein, 2018), meaning it is hard to isolate the most prominent emotion.

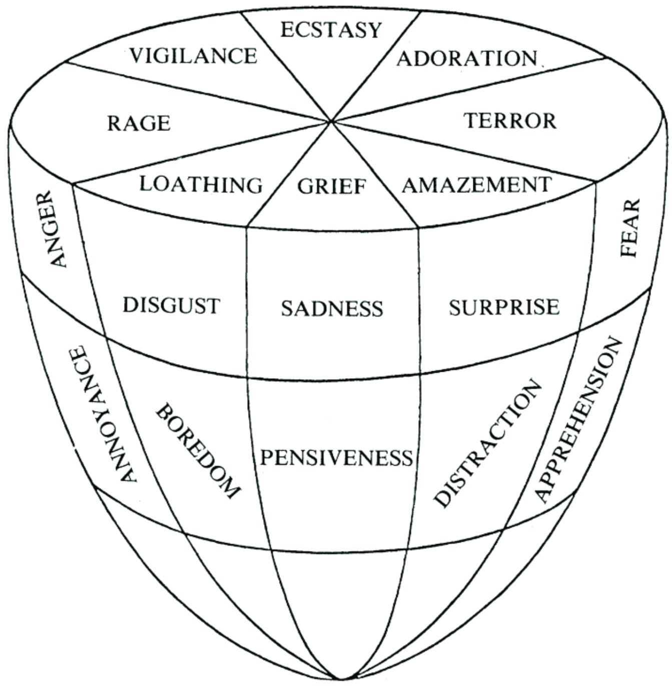


Figure 2.9: Plutchick's Wheel of Emotion (Plutchik, 1984)

The choice of an emotion model would depend on the goals and context of the classification problem. For the context of this dissertation, Ekman's or Plutchik's proposals are likely to fit the objectives better, since distinguishing between some of the complex emotions, such as formality or confidence, probably would not be relevant enough to make up for the larger number of labels and the consequently increased difficulty. Whether a client is using formal speech or communicating with confidence does not necessarily make a difference to help understand if they require human assistance or not. Russell's proposal may also be interesting because there is a relation between the valence dimension and sentiment polarity (a pleasant stimulus could be associated with a positive sentiment) that could be experimented with.

2.4 Dialogue Analysis

A dialogue is a conversation between two or more participants where each speaks in turns (ideally). A turn can consist of a single sentence, a single word, or multiple sentences, making it potentially very complex (Jurafsky and Martin, 2021). Similarly to NLP and SA, which also handle human language, DA also faces several challenges, which will be presented in this section.

DA allows for the modeling and automatic detection of a discourse's structure, usually through Dialogue Act Recognition (DAR), which will be explained further ahead in this section. This kind of operation may be useful for chatbots, where it is helpful to know whether a question was asked or an order was issued, or for meeting summarizers, where it is important to keep track of who said what to whom (Stolcke et al., 2000). Examples of dialogue acts would be Answer, Question, Request, Greet, or Acknowledgment.

Besides chatbots, which are designed for extended conversations, another type of dialogue system is the task-oriented dialogue system, where the conversational agent helps the user complete tasks (e.g., they can give directions, make restaurant reservations, or make phone calls) (Jurafsky and Martin, 2021). This is not the focus of this work, so we will not delve deeper into this subject.

Dialogue analysis may not be sufficient to perform SA, since the structure of a conversation may not give enough information to assess the sentiment behind some utterances. For example, knowing that an answer follows a question, or that someone is giving an opinion, does not, by itself, provide information on the sentiment portrayed.

2.4.1 Challenging properties of human conversation

Jurafsky and Martin (2021) explored some of the properties of human conversation, which are among the reasons why it is hard to build dialogue systems capable of carrying on natural conversations. These challenges are summarized in this subsection, and Table 2.4 will be used as an example to better visualize the challenges.

It is important for a system to understand when a user has finished talking, so that it can process its utterance and reply to it, and also when the system itself should stop talking, due to an interruption from the user or a similar situation.

These tasks, called *Endpoint Detection*, can be challenging since in conversation, speakers start or stop talking almost immediately after an interruption or the end of an utterance. This is also made harder due to noise. An example of an interruption is present in turns 16 and 17 of Table 2.4.

In a conversation, it is important to establish a common ground between the participants, which is one by grounding each other's utterances.

Grounding is an acknowledgment that what was said by one was understood by the other. A system can perform this kind of action by explicitly agreeing with the other user's utterance, by repeating it or parts of it, or by implying that the utterance was understood and moving to another action (e.g., by using the word "And" at the start of the utterance). Examples of grounding are present in turns 8 or 10 of Table 2.4, where the agent says *OK*, or in turn 2, where the agent repeats *in May*.

In human conversations, initiative shifts back and forth between the participants. *Initiative* in this context means actions such as asking questions, changing topics, or asking

Table 2.4: Example of a conversation between a human travel agent and a client. The symbol # indicates overlaps in speech (Jurafsky and Martin, 2021).

Speaker	**Turn Number**	**Utterance**
Client	1	... I need to travel in May.
Travel Agent	2	And, what day in May did you want to travel?
Client	3	OK uh I need to be there for a meeting that's from the 12th to the 15th.
Travel Agent	4	And you're flying into what city?
Client	5	Seattle.
Travel Agent	6	And what time would you like to leave Pittsburgh?
Client	7	Uh hmm I don't think there's many options for non-stop.
Travel Agent	8	Right. There's three non-stops today.
Client	9	What are they?
Travel Agent	10	The first one departs PGH at 10:00am, arrives Seattle at 12:05 their time. The second flight departs PGH at 5:55pm, arrives Seattle at 8pm. And the last flight departs PGH at 8:15pm, arrives Seattle at 10:28pm.
Client	11	OK I'll take the 5ish flight on the night before on the 11th.
Travel Agent	12	On the 11th? OK. Departing at 5:55pm arrives Seattle at 8pm, U.S. Air flight 115.
Client	13	OK.
Travel Agent	14	And you said returning on May 15th?
Client	15	Uh, yeah, at the end of the day.
Travel Agent	16	Ok. There's #two non-stops...#
Client	17	#Act...actually #, what day of the week is the 15th?
Travel Agent	18	It's a Friday.
Client	19	Uh hmm. I would consider staying there an extra day til Sunday.
Travel Agent	20	OK... OK. On Sunday I have...

for clarification.

This is difficult for dialogue systems to achieve, as they are usually passive responders, reacting only to the user's initiative.

Inference can be crucial in the understanding of dialog. Sometimes users will not explicitly reply to a question but will mention something related to it instead.

An example of inference is present in turns 2 and 3 of Table 2.4, where the client mentions the need to be in Seattle from the 12th to the 15th. The agent knows that to have a meeting it is required to be at the place where the meeting is held, and therefore that the meeting may be a reason for the travel. Since people like to arrive the day before a meeting, the agent infers that the flight should be on the 11th.

2.4.2 Dialogue Acts

Dialogue acts are labels (e.g., statement, opinion, question, response, and agreement) over utterances, which roughly correspond to the speaker's intention (Einsenstein, 2018).

Some dialogue acts compose adjacency pairs, which represent the relation between the first pair part and the second pair part. Questions are usually followed by answers and proposals by acceptance or rejection. However, it is important to take into consideration that the second pair part may not follow the first pair part immediately, meaning they can be separated by a side sequence (e.g., a clarification or correction) (Jurafsky and Martin, 2021).

Dialogue analysis and dialogue acts are usually defined according to a particular application, meaning the lack of domain independence is a common challenge (Stolcke et al., 2000). In these situations, the label and acts sets must be designed according to the needs of the problem at hand.

According to (Stolcke et al., 2000), the most frequent dialogue acts are:

- Statement and Opinion, that can be combined or distinguished, if relevant. An opinion is usually countered by other opinions and often includes expressions such as *I think* or *I believe*;
- Question, that can be split into yes or no questions, wh-questions, and declarative questions, which lack indicative words such as *who*, *where*, *when*, as opposed to wh-questions;
- Backchannel, that has a discourse-structuring role, and is usually an indication that the speaker should continue talking (e.g., *um* or *uh-huh*), and also commonly occur at syntactic boundaries;
- Abandoned utterances, that happen when the speaker does not finish the utterance and are usually followed by a restart;
- Answer and Agreement, that can be distinguished by the previous dialogue act (if it is a question or an opinion or proposal), since *yes* could be an answer or an agreement.

In the case of task-oriented dialogue systems, the dialogue acts can be related to the task at hand, and could include *Hello*, *Inform_ Restaurant*, *Request_ Hotel*, *Deny_ Train_ Booking*, or *Bye*, for example. *Inform*, *Request*, and *Deny* acts apply to several domains, as in the examples given.

DAR and SA are closely related and mutually promote each other by being jointly performed. The former's goal is to label each utterance in a dialogue and identify the underlying intention, while the latter's goal is to detect the sentiment transmitted by each utterance, which can help to better capture a speaker's intention (Qin et al., 2020a). Generally, the same sentiment will be expressed once the dialogue act *Agreement* is assigned. Knowing the sentiment information also contributes to the dialogue act prediction, because when the speaker changes the sentiment from *Negative* to *Neutral*, there is a tendency for the dialogues to transition into *Statement* (Li et al., 2020). An Opinion dialogue act usually contains a Non-Neutral sentiment, while a Question is more likely to be Neutral.

2.5 Summary

This section offers a summary regarding the present chapter, which presented NLP, ML, SA, and DA.

This work will deal with human language, both for SA and for DA, which will require NLP techniques so that machines know how to handle the supplied data correctly. Both

SA and DA use classification models. These models can use supervised ML algorithms, which may internally apply NLP techniques.

The above paragraph summarizes how the presented domains relate to each other and the goal of this dissertation, SA in the context of dialogue.

In the following chapter, some of the work done in contexts that are relevant to the development of this dissertation will be presented.

Chapter 3

Related Work

In this chapter, some approaches to sentiment analysis will be summarized, in three different contexts: in Twitter data, its application to the Portuguese language, and in combination with Dialogue Act Recognition (DAR), also known as Dialogue Act Classification.

These three contexts were chosen because they relate to the development of this work. The first is because Twitter has been a popular target for Sentiment Analysis (SA), and also because one of the goals is the creation of a dataset, and dialogues can be established on Twitter. The second, as it is a requirement that the models developed are applied to Portuguese. The third, because the combination of SA and DAR has proved to improve the results, in comparison to using SA alone (Li et al., 2020).

The available datasets for SA will also be summarized, in addition to some techniques for dataset creation.

3.1 Sentiment Analysis with Twitter data

This section will present some of the work developed for SA with Twitter data.

One of the most studied Natural Language Processing (NLP) tasks on Twitter is sentiment recognition (Cerisara et al., 2018), which is for example a recurrent task at the International Workshop on Semantic Evaluation (SemEval)(Rosenthal et al., 2017) evaluation campaign. SemEval consists of a series of evaluations of computational semantic analysis systems and explores tasks such as SA of product review and their aspects (Pontiki et al., 2016), SA of figurative language on Twitter (Ghosh et al., 2015), detecting stance in tweets (Mohammad et al., 2016), and emotion detection (Strapparava and Mihalcea, 2007). In fact, the work by Severyn and Moschitti (2015) at SemEval-2015 Task 10: SA in Twitter, was the winner for subtask A: Phrase-level Polarity and placed second for subtask B: Message-Level Polarity (Rosenthal et al., 2015). This work will be explained further ahead.

Furthermore, Twitter data can be used to compile datasets of dialogue (Ritter et al., 2010), which is a goal of this work. Burkov (2020) mentions that a dataset should reflect real inputs, i.e., the training samples should contain inputs like the ones that the model will receive in a real-world application. In the context of this dissertation, a good source of data would be Twitter, due to the short and informal messages typical of this social media, which resemble the real inputs that the solution will need to handle. Another advantage of using Twitter as a data source is that dialogues between two or more users are very present in this social network, and are of high importance for this work since the real inputs will consist of a conversation between two users.

Tweets often include abbreviations, private information, or misspelled words, so preprocessing is an important step of any work that handles Twitter data. Usernames, hashtags, URLs, or even emoticons (if not being used by the solution) can be replaced by identifiers (e.g., *USER*, *URL*) in order to protect the identity of the entities involved. Abbreviations can be converted to their original forms, and a spell checker can be applied to correct misspelled words.

Table 3.1 summarizes the works that applied SA using Twitter data, regarding the classification technique used and the classification result.

Table 3.1: Works about SA with Twitter data

	Classification Technique	**Result**
Duarte (2013)	Lexicon-based (SentiWordNet)	Sentiment (aspect-based)
Duarte et al. (2019)	Naive Bayes and SVM	Emotion
Madanian et al. (2021)	LSTM	Sentiment
Kouloumpis et al. (2021)	AdaBoost and SVM	Sentiment
Pak and Paroubek (2010)	Naive Bayes Classifier, SVM, and CRF	Sentiment
Severyn and Moschitti (2015)	CNN	Sentiment
Jianqiang et al. (2018)	CNN	Sentiment

Looking at Table 3.1, it is noteworthy that only the first of the related works used a Non-Machine Learning (ML) approach, (Duarte, 2013). Regarding the ML approaches, in the works presented, Support Vector Machine (SVM)s and Convolutional Neural Network (CNN)s were the most common models. It is also of note that only the oldest work used Conditional Random Field (CRF), which did not perform as well as a Naive Bayes classifier for sentiment classification of documents.

The first two entries in Table 3.1 will be explained in the next section since they target the Portuguese language. The remaining works will be summarized below.

Madanian et al. (2021) used Twitter to explore reactions and perceptions of the population towards Covid-19 health messaging. This work used Lamsal (2020)'s Covid-19 Tweets Dataset, which contains IDs and sentiment scores of tweets related to the pandemic. The data was collected using the Twitter Stream Application Programming Interface (API), and it looked up tweets with keywords or hashtags commonly used when referencing the Coronavirus. Before computing the sentiment scores, the tweets were cleaned, i.e., symbols, URLs, emojis, and similar components were removed, spelling correction was applied, and abbreviations were converted to their original forms. To obtain the sentiment scores, a Long Short-Term Memory (LSTM) deep network model was used, and the values are defined in the range [-1,+1], where a negative value means a *Negative* sentiment, a positive value means a *Positive* sentiment, and a value of zero means a *Neutral* sentiment. Madanian et al. (2021) used text extraction to explore sentiments regarding health messages such as *wash hands*, *facemask*, *social distancing*, and others. The outcome of tweets analysis such as this could help identify the concerns and reactions to covid-19 and provide a better perception of the situation to governments, supporting them in implementing appropriate policies.

Kouloumpis et al. (2021) uses a hashtagged data set (i.e., they selected messages containing hashtags that could be associated with sentiment) and an emoticon data set (i.e., they selected messages containing either *:)* or *:(*) for development and training, while for evaluation they used a manually annotated data set on certain topics. In this work, they found that AdaBoost performed better than SVMs, and concluded that the presence of intensifiers (e.g., all-caps, *I LOVE THIS*, and character repetitions, *happyyyyyyyyy*) and sentiment emoticons were very useful for SA.

Pak and Paroubek (2010) automatically collect a corpus for SA, containing sentiment polarity, and build sentiment classifiers for documents. The tweets collected were queried for happy and sad emoticons (e.g., *:-)*, *:D*, *=(*, *;(*) and for accounts of popular newspapers (e.g., *New York Times*, *Washington Posts*) to obtain objective texts. They constructed n-grams and verified that better results were obtained when using bigrams and that a Naive Bayes classifier outperformed the SVM and the CRF models.

Severyn and Moschitti (2015) and Jianqiang et al. (2018) used CNNs as their classification model. Jianqiang et al. (2018) used a CNN to refine the word embeddings on a large corpus, then the pre-trained parameters from this model are used to initialize the final CNN model. They claim that providing the network with good initialization parameters has a significant impact on the accuracy of the model. Severyn and Moschitti (2015) also focused on providing good features, obtaining Global Vector (GloVe) representations, which were then combined with n-grams and polarity scores (obtained using a lexicon). They then fed the combined features to a CNN for training and predicting the sentiment labels, and called this model *GloVe-DCNN*. The number of hashtags, emoticons, and capitalized words, for example, were also used as features in the work by Jianqiang et al. (2018). These two final words make clear the importance of representation, mainly when using CNNs that better capture the neighboring words.

There is a downside to using Twitter to create a dataset, which is that the platform's license only allows the tweets' (or users') ids to be published[1]. If a user deletes a tweet or if, for some reason, the tweet becomes unavailable, then part of the compiled dataset would not be reachable, and the results of the project will not be completely reproducible.

3.2 Sentiment Analysis in the Portuguese language

This section will present some of the work developed for SA in the Portuguese language.

One of the challenges of this work is the language restriction because the developed approaches are generally applied to English. However, there have been some projects that have addressed this constraint. There is a survey of SA for the Portuguese language (Pereira, 2021) that categorizes and describes works involving approaches to each of the tasks of SA, as well as NLP tools, lexicons, datasets, and more. This work claims that often translating texts into English and using tools developed for that language may be more effective than efforts in Portuguese, however, this is highly debatable.

Table 3.2 summarizes the works that applied SA to the Portuguese language, regarding the classification technique used, whether translation was applied, the classification result, whether it was applied to European or Brazilian Portuguese, and the data source used for the evaluation of the models.

Only Duarte (2013) and Hammes and de Freitas (2021) used translation in their works. The former translated n-grams to English and looked them up on *SentiWordNet* (Baccianella et al., 2010), a sentiment dictionary, to get their sentiment values. The latter applied automatic translation to GoEmotions, a highly unbalanced dataset that contains

[1]Content redistribution terms: `https://developer.twitter.com/en/developer-terms/agreement-and-policy`

Table 3.2: Works about SA in the Portuguese language

	Classification Technique	Translation	Result	Portuguese	Data Source for Evaluation
Carvalho and Silva (2015)	Lexicon-based (SentiLex)	No	Sentiment	European	SentiCorpus-PT and a literary piece "Os Pobres"
Balage Filho et al. (2013)	Lexicon-based (LIWC)	No	Sentiment	Brazilian	ReLi
Duarte (2013)	Lexicon-based (SentiWordNet)	Yes	Sentiment (aspect-based)	European	Twitter
Duarte et al. (2019)	Naive Bayes and SVM	No	Emotion	European	Twitter
Dosciatti et al. (2013)	SVM	No	Emotion	Brazilian	Emotion in News Portuguese dataset
Hammes and de Freitas (2021)	Fine-tuned BERT, BERTimbau	Yes	Emotion	Brazilian	Translated GoEmotions dataset

over 50.000 sentences from Reddit manually labeled in 28 classes (including *Neutral*).

Half of the works in Table 3.2 used lexicon-based approaches, while the other half used machine learning approaches. While Duarte (2013) based his classification in SentiWordNet, a tree structure was also implemented, which represented relations between each segment of the input. The sentiments found for each segment are assigned to the closest entity within the tree and are then grouped into a single label for each entity found, as well as for the whole message. It is also relevant to mention that Hammes and de Freitas (2021) fine-tuned the pre-trained BERT models for the Portuguese language (BERTimbau) and also applied a balancing algorithm to GoEmotions, which allowed for better results.

Regarding the data source for the evaluation of the models, Carvalho and Silva (2015) used SentiCorpus-PT (Carvalho et al., 2011), a social media corpus of user comments on political news articles in Portuguese, and a book by Raul Brandão. Resenha de Livros (ReLi) (Freitas et al., 2014), used by Balage Filho et al. (2013), is a corpus of book reviews in Portuguese. Dosciatti et al. (2013) used a corpus of news extracted from a Brazilian newspaper. Hammes and de Freitas (2021) translated the GoEmotions dataset, which contains over 50.000 sentences from Reddit. The remaining works, (Duarte, 2013, Duarte et al., 2019) used data extracted from Twitter for the evaluation of their models. There seems to be a tendency for using social media (Twitter, Reddit) or online newspapers as a data source.

Further details regarding each of the works mentioned in Table 3.2 are presented below.

Carvalho and Silva (2015) developed a sentiment lexicon and a lexicon-based solution for the extraction of sentiment and opinion in Portuguese texts. Each entry was selected by its ability to modify human nouns and classified by its polarity (*Positive*, *Negative*, or *Neutral*). The use of human predicates makes it a lexicon oriented by the syntactic restrictions of the predicates and not by any specific domain of knowledge.

Balage Filho et al. (2013) presents an evaluation of the Brazilian Portuguese Linguistic Inquiry and Word Count (LIWC) dictionary for SA in Brazilian Portuguese texts. The mentioned dictionary has sentiment polarity labels associated with each entry. The classification algorithm used is similar to the SO-CAL algorithm (Taboada et al., 2011), which computes the individual polarity of each word and then sums them up to form the text polarity. The results showed that this lexicon had higher *F-score* results for positive texts (in opinion and sentence classifications) than SentiLex-PT, which had higher *F-score* results for negative texts. This suggests that the LIWC dictionary performs better indicating positivity than negativity in opinion and sentence classifications.

In his Master's dissertation, Duarte (2013) proposed a SA and entity recognition tool using a tree structure, which represented relations between each segment of the input by using part of speech (e.g. noun, verb, adverb) classification. This work used Twitter messages in the Portuguese language to extract information concerning different entities. The messages were filtered and all known entities were marked and sentiment n-grams were labeled (*Positive*, *Negative*, or *Neutral*). To get the sentiment value of the n-grams, they were translated to English and looked up on *SentiWordNet* (Baccianella et al., 2010), a sentiment dictionary mentioned in section 2.3. After the identification, the sentiments found are assigned to the closest entity within the tree and are then grouped into a single label for each entity found, as well as for the whole message.

The following two works used Ekman's six basic emotions model (Ekman, 1992) as their classification target.

Duarte et al. (2019) collected texts from Twitter, and exploited emojis, often connected with emotions, to classify emotion. They also suggested that emojis could be used for self-labeling, as an alternative to manual annotation, which is a similar approach to previous words that used emoticons as features. However, the amount of conversations on Twitter where each text contains at least one emoji is low, which would limit the applications of this approach.

Dosciatti et al. (2013) presented an approach using linear multiclass SVMs and a labeled dataset consisting of news extracted from a Brazilian newspaper. This work did not consider words with an occurrence lower than four and then selected words with an information gain above 68%, reducing the dimensionality. The features used for training the SVMs were designed using Term Frequency - Inverse Document Frequency (TF-IDF) to set the weights for each word.

A transformer-based solution was proposed by Hammes and de Freitas (2021), who fine-tuned the pre-trained Bidirectional Encoder Representations from Transformers (BERT) models for the Portuguese language (BERTimbau) for the classification of 27 emotions. This solution applied automatic translation to the GoEmotions dataset, a highly unbalanced dataset (the most frequent class has 184 times more samples than the least frequent class) manually labeled in 28 classes (including *Neutral*). This work used a balancing algorithm, calculating the weights based on the number of samples of each class, which was considered the reason for the better results in comparison to those obtained using the BERT model for the original GoEmotions dataset.

3.3 Sentiment Analysis and Dialogue

This section will present some of the work developed for SA and DAR as joint tasks which mutually promote each other.

Mastodon (Cerisara et al., 2018) and DailyDialog (Li et al., 2017) are popular datasets for sentiment and dialogue classification, and are labeled for dialogue acts (e.g., *Inform*, *Question*, or *Directive*) and sentiment (*Positive*, *Negative*, *Neutral*) or emotion (e.g., *Anger*, *Joy*, or *Sadness*). Table 3.3 presents a snippet of a dialogue sample from Mastodon, that showcases the previously mentioned relation between SA and DAR, where the same sentiment is expressed when the dialogue act is Agreement.

Some approaches that perform the two classification tasks simultaneously have been developed, as shown in Table 3.4. All of the presented works used the Mastodon and DailyDialog datasets for evaluation.

Looking at Table 3.4, deep learning seems to be the common option for the joint tasks

Table 3.3: Dialogue sample from the Mastodon dataset, and their corresponding dialogue act and sentiment labels

Speaker	**Utterance**	**Dialogue Act**	**Sentiment**
User A	they are as tired of social media as i am .	*Statement*	*Negative*
User B	yes ! i don't get it . everyone i talk to about facebook–everyone - - hates it ,but none of them will take action .	*Agreement*	*Negative*

Table 3.4: Works about SA and DAR

	Classification Technique	**Result**
Qin et al. (2020a)	Deep Co-Interactive Relation Network	Dialogue act and sentiment
Li et al. (2020)	Bi-channel Dynamic Convolutions	Dialogue act and sentiment
Qin et al. (2020b)	Co-interactive Graph Attention Network	Dialogue act and sentiment

of SA and DAR. Since the models used are more complex than in previous solutions, these architectures will be explained in more detail below.

Qin et al. (2020a) proposed a Deep Co-Interactive Relation Network (DCR-Net) which adopts a co-interactive relation layer to explicitly model the contextual and mutual interaction information between both tasks. The proposed architecture uses one hierarchical encoder, consisting of a Bi-directional LSTM (BiLSTM), which captures temporal relations, and a self-attention layer, which considers the contextual information, shared by both operations. The co-interactive layer takes as input the dialogue act and the sentiment representations and outputs new representations, which consider the cross-impact on the two tasks. Before combining the information, a BiLSTM is applied for DAR, and a Multilayer Perceptron (MLP) is applied for SA to make them more task-specific. To better capture mutual information, the co-interactive layer can be stacked. Two separate decoders are used to perform the DAR and SA predictions. At the time, the DCR-Net model achieved state-of-the-art performance in Mastodon and DailyDialog.

Li et al. (2020) proposed Bi-Channel Dynamic Convolutions, which aimed at improving the joint DAR and SA tasks by fully modeling the local contexts of utterances. The proposed architecture uses a BiLSTM layer to encode the input utterance into representations, and then, in the utterance encoder layer, multi-layer bi-channel versions of Dynamic Convolution Network (DCN) and of Context-aware Dynamic Convolution Network (CDCN) are used to capture the context representations for DAR and SA, respectively. DCN is a variant of CNNs that is more flexible on mining contextual features but ignores the surroundings of the utterance, while CDCN is a variant that takes context into account, allowing for a more informative contextual representation. After the context representations are captured, a linear transformation is applied, and the model makes the predictions separately. This approach used DiaBERT (Liu and Lapata, 2019), a BERT-based encoder that takes the whole dialogue and outputs a complete representation, while BERT restricts the input to a maximum of two utterance sentences. The proposed model achieved better results than DCR-Net in Mastodon and DailyDialog.

Qin et al. (2020b) proposes a Co-Interactive Graph Attention Network (Co-GAT) to jointly perform the DAR and SA tasks. In general, previous models either consider only one source of information (contextual or mutual interaction) or employ both types using a pipeline modeling method, which results in the two kinds of information being modeled separately. In this model, a cross-utterances connection, for contextual information, and a cross-tasks connection, for mutual interaction information, are constructed and iteratively updated with each other to simultaneously model both types of information. Both connections are integrated into a unified graph architecture, and each node can be updated simultaneously with contextual and mutual interaction information. The architecture is composed of a shared hierarchical speaker-aware encoder (using a BiLSTM to capture temporal relationships, and a speaker-aware Graph Attention Network (GAT) to incorporate the speaker information), a stack of co-interactive graph layers (models the interaction process with the cross-tasks and cross-utterances connections), and two separate decoders for each prediction. The proposed model achieved better results than DCR-Net in Mastodon and DailyDialog but was not compared with the model proposed by Li et al. (2020).

A common conclusion of the referenced works is that using fine-tuned BERT improved the results obtained.

3.4 Datasets for Sentiment Analysis

This section will present dataset creation approaches, as well as some of the already available options.

A dataset is required for any kind of solution. Since the development of one or more compilations of data is a goal of this dissertation, some datasets were explored. However, it should be kept in mind that the domains of interest for this dissertation, as defined by Altice Labs (AL), are eCommerce, TV, Health Care, Finance and FinTech, and Telecommunications, which may not be represented in the datasets summarized in Table 3.5.

Table 3.5: Available datasets

	Size	**Annotation**	**Language**	**Data Source(s)**	**Domain(s)**
DailyDialog	13.118 dialogues	Dialogue act and emotion	English	Websites for the practice of English dialogues	Daily life topics (e.g., buying goods from a shop, summer vacation)
Mastodon	505 dialogues	Dialogue act and sentiment	English	Social Media (Mastodon)	General dialogues in the octodon.social Mastodon instance
Friends' Emotion Detection	12.606 utterances	Emotion	English	TV Show Friends	Daily life topics common in comedy TV shows
CORAA	400.000+ audio transcriptions	-	Brazilian Portuguese	Five audio corpora	Spontaneous speech (e.g., interviews, informal conversations)
Emotion in News	1.750 news	Emotion	Brazilian Portuguese	News extracted from Globo news website	News (international, national, politics, economics, law enforcement)
ReLi	1.600 reviews	Opinion and sentiment	Brazilian Portuguese	Book reviews posted on the Internet	Book reviews
Sentituites-PT	30.470 tweets	Sentiment	Portuguese	Tweets posted during the 2011 Portuguese elections	Politics
Wizard of Wikipedia	22.311 dialogues	-	English	WoZ framework	Wikipedia
Multi-WoZ	8.438 dialogues	Dialogue act and slots	English	WoZ framework	Restaurants, Attractions, Hotels
CamRest	680 dialogues	Dialogue act and slots	English	WoZ framework	Restaurant search
Ubuntu Dialog	Nearly 1 million dialogues	-	English	Ubuntu chat logs (2004-2015)	Technical support for Ubuntu-related problems

A brief analysis of Table 3.5 shows that only two datasets are labeled for dialogue acts and sentiment or emotion, DailyDialog (Li et al., 2017) and Mastodon (Cerisara et al., 2018) (Table 3.3 shows a snippet from a Mastodon conversation). DailyDialog is a multi-turn, human-written, dialogue dataset collected from various websites which serve for the English learner to practice English. Mastodon is a Twitter-like corpus from an alternative social network, Mastodon, which allows for reproducible experiments. Both these datasets are commonly used for model evaluation, as seen in Table 3.4.

From Table 3.5 we can also see that only three datasets do not contain dialogues, Emotion in News (Dosciatti et al., 2013), ReLi (Freitas et al., 2014), and Sentituites-PT (Carvalho et al., 2011). These datasets are in the Portuguese language and are being considered because they can be used to verify if a model trained with dialogue data could be adapted to non-dialogue contexts. For this use, the Sentituites-PT data could be more interesting, since it is larger and in Portuguese.

The remaining datasets, DailyDialog, Mastodon, Friends' Emotion Detection (Zahiri and Choi, 2018), CORAA (Junior et al., 2021), Wizard of Wikipedia, Multi-WoZ (Budzianowski et al., 2018), CamRest (Wen et al., 2016), and Ubuntu Dialog, contain dialogues, but are in English. Regarding the Friends' Emotion Detection dataset, it presents a collection of multiparty dialogues from the TV show Friends with emotion labels for each utterance. Since it contains dialogue and emotion, this could be an interesting dataset, despite being in the English language. The language restriction could be bypassed by using available Portuguese subtitles of the TV show, aligning them with the utterances, and importing the labels from the original dataset.

It is also worth noting that none of the datasets cover the desired domains, Ubuntu Dialog being the closest. The largest datasets, Wizard of Wikipedia (Dinan et al., 2018) and Ubuntu Dialog (Lowe et al., 2015), are, sadly, not annotated, as seen in Table 3.5. The CORAA dataset is not labeled, however, there is another version of this dataset, CORAA SER, composed of approximately 50 minutes of audio segments labeled in three classes (*Neutral*, *Non-Neutral Female*, *Non-Neutral Male*), based on audio features. This version of CORAA is annotated for emotions, so Neutral means there was no well-defined emotional state, and Non-Neutral was labeled considering paralinguistic elements (e.g., laughing, crying), which do not apply to our context. This dataset was introduced in the S&ER 2022 Workshop[2], which is part of the 15th edition of the International Conference on the Computational Processing of Portuguese (PROPOR 2022).

Regarding domains and data sources, there does not seem to be a trend between the presented works, except maybe for the adoption of a Wizard of Oz (WOZ) framework, since there are several options and domains from where to extract or create data.

Despite not being mentioned in Table 3.5, five of the datasets contained information regarding the average number of turns per dialogue. Multi-WoZ had the largest ratio, *13.46*, while CamRest had the smallest ratio, *5*. DailyDialog had a ratio of *7.9*, Wizard of Wikipedia had a ratio of *9.1*, and Ubuntu Dialog had a ratio of *8* turns per dialogue.

The datasets not yet presented will be summarized below, as well as the option of creating a dataset using a WOZ framework.

The CORAA corpus (Junior et al., 2021) is an Automatic Speech Recognition (ASR) dataset comprised of over 290 hours of audios in Brazilian Portuguese and their respective transcriptions. This dataset combined five existing audio corpora, ALIP (Gonçalves, 2019), C-ORAL Brasil I (Raso and Mello, 2012), NURC Recife (Oliviera Jr., 2016), SP2010 (Mendes and Oushiro, 2013), and TEDx Portuguese talks[3]. For this work, only the transcriptions would be of use, since it is not a goal to experiment with data in an audio format.

ReLi and Emotion in News were used in works mentioned in the previous sections. ReLi contains book reviews manually annotated and their polarities, and Emotion in News contains news extracted from a Brazilian newspaper and uses Ekman's emotions for classification.

SentiTuites-PT is a corpus of tweets posted by Portuguese users during the 2011 election campaign. The corpus is labeled for sentiment polarity with values -1 (*Negative*), 0 (*Neutral*), and 1 (*Positive*), regarding the entity identified in the tweet.

[2] https://sites.universidadedefortaleza.com/propor2022/?page_id=367
[3] www.ted.com

Multi-WoZ and CamRest are task-oriented datasets, which is why they include a label for slots, or dialogue states (e.g., *postcode*, *name*, *phone*), and not just dialogue act labels (e.g., *recommend*, *not found*, *greet*). Both datasets, along with Wizard of Wikipedia, were created using a WOZ framework.

In Wizard of Wikipedia, a dataset of human-to-human conversations was built using crowd-sourced workers. Several topics connected to Wikipedia[4] were discussed, and one of the humans, playing the role of the *apprentice*, has the goal of going in-depth about a chosen topic and keeping the conversation engaging, while the other human, playing the role of the *wizard*, has access to the Wikipedia information and has the goal of informing the other person about a topic, providing them with a relevant and engaging reply. CamRest and Multi-WoZ also used crowd-sourced workers to develop the conversations. In the former, the system was designed to help users find restaurants in the Cambridge area, while in the latter, the system was designed to help tourists find restaurants, attractions, and hotels.

A commonly adopted approach for the development of a dataset is through the WOZ framework (Kelley, 1984). WOZ consists of one or more *wizards*, humans who simulate part or whole of the performance of the system being designed, while interacting with users, *apprentices*, who preferably believe themselves to be using a real system. A series of interactions in this format has the potential to deliver a specification of the system's input/output behavior which can then be safely implemented, by replacing the *wizard* with a learned agent (Bernsen et al., 1994). This approach was challenging in the context of this dissertation, due to the logistical needs involved, mainly the time required to set up the framework, and the manpower required to make it work, and so was not pursued.

Gonçalo Oliveira et al. (2022) presents an interesting overview of existing dialogue datasets, containing several more options than presented in Table 3.5, but still, none of them check all the requirements for this work.

As one can verify, the options for dataset creation or application are vast, although the majority are, unfortunately, not applicable to this work. The choice of an approach would depend on the language, annotation, type of speech (dialogue or regular text), domains represented, and the goals of the work itself.

3.5 Summary

In this section, work related to several parts of this dissertation was overviewed, and can be summarized as follows:

Regarding the use of Twitter data for SA, it was verified that it is commonly used in similar problems and that it may be a suitable source for real-world data. Tweets can be collected using APIs and may require somewhat complex preprocessing. For some domains, there are already collections of tweets available.

Regarding the work applied to the Portuguese language, some possible solutions involve using a Portuguese sentiment lexicon, translating the Portuguese input and looking up sentiments in an English dictionary, collecting data in the Portuguese data, or using fine-tuned models in the Portuguese language.

Regarding the combination of SA and DAR, it was confirmed that solutions that consider these joint tasks have interesting results. It is also noteworthy that the use of a fine-tuned BERT model improved the performance for each of the mentioned approaches.

[4] `https://en.wikipedia.org/wiki/Main_Page`

Finally, there are several options for the development of a dataset, the translation of an available collection, the use of speech-to-text data (samples provided by AL), the extraction of data from Twitter, or the development of an intricate WOZ approach. However, not all of them will be usable for this work, either because of time constraints, regarding the WOZ option, or due to data incompatibilities, regarding the translation option. This last one will be explained further in section 5.1.

In the following chapter, we will present the work plan (estimated vs. real) for this dissertation.

Chapter 4

Planning

This chapter presents the expected tasks and corresponding deadlines predicted in the early stage of this dissertation and compares them with the real work plan.

The following tasks were defined:

- Data Creation: Includes the creation of two datasets, and the exploration of other alternatives;
- Data Annotation: Includes the manual annotation process and the exploration of the emoji-based alternative;
- Data Analysis: Includes the analysis of the number of turns, the number of dialogues, the average turns per dialogue, the number of samples of each label, the agreement level, and the frequencies of each label, as well as a general analysis of the data gathered (e.g., analysis of the tweets from each account);
- Models Development: Includes the development of baseline and Machine Learning (ML) approaches, and the latter includes text (e.g., Random Forest, Support Vector Machine (SVM)s) and sequence/context (e.g., Transformers) approaches;
- Models Evaluation: Includes the application of evaluation metrics to the created datasets;
- Models Optimization: Includes the comparison of different solutions and corpus available, the implementation of hyperparameters' tuning and cross-validation techniques, and the proposal of a solution that would identify recurrent problems;
- Documentation: Includes the writing of the dissertation and the scientific articles, commenting on the code developed, and writing on Altice Labs (AL)' Wiki page.

The predicted work plan, containing the mentioned tasks and their expected timelines is summarized in Figure 4.1. The actual work plan is presented in Figure 4.2.

There are plenty of differences between the original (predicted) and actual work plans. First, the timeline was extended from June to August, due to delays in the development of the proposed work and so that more results, experiences, and analyses could be performed.

Regarding the beginning of each task, the majority was accurate, with the exception of the Models Optimization task, which was postponed to the end of July, due to the extended time taken with the previous tasks.

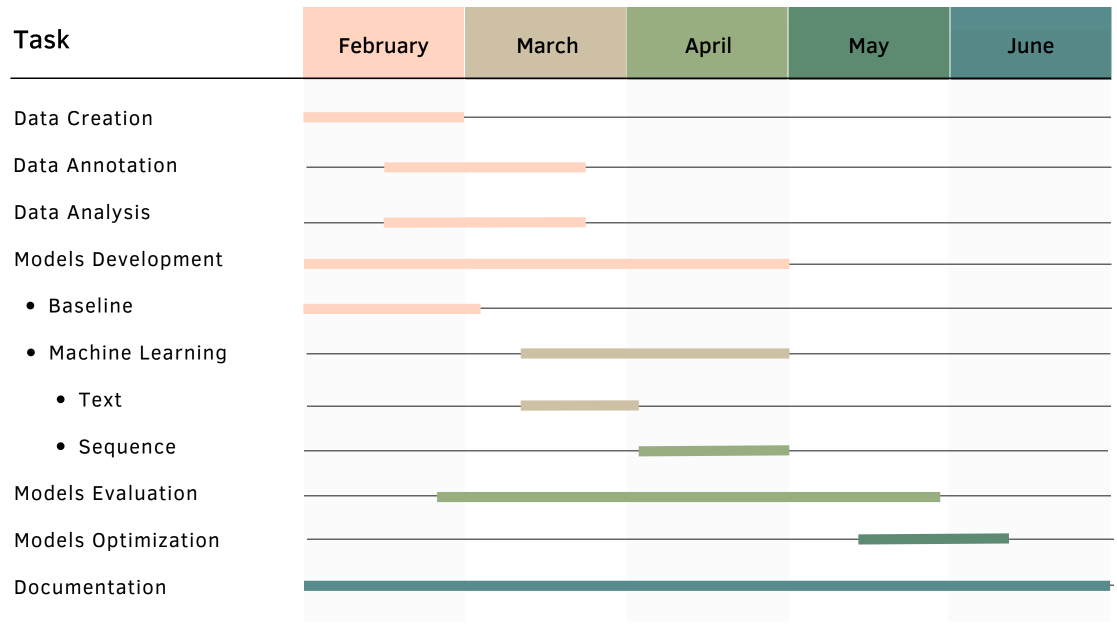


Figure 4.1: Original Work Plan

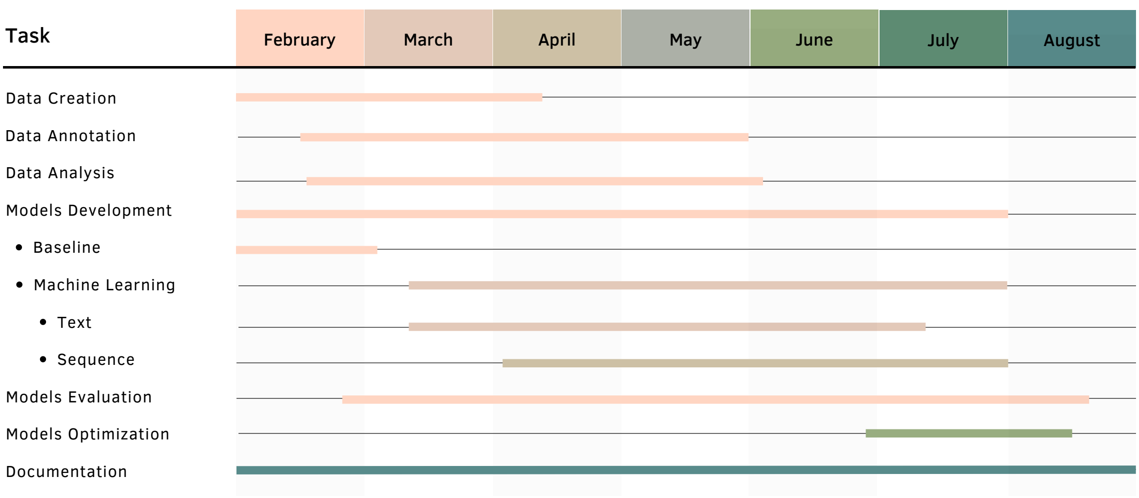


Figure 4.2: Actual Work Plan

Regarding the Data-related tasks, what took the longest was the data annotation, especially since it was dependent on the availability of third parties, but also because the choice of data would affect all the remaining work. Hence, we took some time to consider the pros and cons of each possibility (e.g., translation of the Friends' Emotion dataset, extraction from Twitter, annotation of existing datasets such as CORAA or Ubuntu Dialog, generation of data based on the AL dataset, development of a Wizard of Oz (WOZ) framework), and perform some small experiments and data exploration, which naturally was a long process, but an important one, since the amount and quality of the data can make or break the learning process of a classifier. In the Data Creation task, besides the extraction of Twitter data and the selection of the data by AL, we also spent some time exploring the Generative Pre-trained Transformer (GPT)-2 and GPT-3 text generation abilities, the possibility of adapting the Friends' Emotion dataset, and the use of the Sentituites dataset to explore whether models trained on non-dialogue data could correctly classify dialogues.

Regarding the Models Development tasks, the development of the baseline models was easily achieved in the predicted timeline, but the remaining options took longer than expected. This was mainly due to bugs and their resolution, technical issues (e.g., lost connections, unavailable extra machines), and more research about more complex approaches, such as the Few-Shot Learning (Few-SL), the Bidirectional Encoder Representations from Transformers (BERT) fine-tuning, and the sequence-specific models, Conditional Random Field (CRF) and BERT-CRF, the latter of which had to be developed in a more manual approach as to combine both individual models. Furthermore, the exploration of a broad range of different classifiers, sometimes with different requirements regarding their configuration, packages, and time complexity, was a challenging and time-consuming process.

Overall, the majority of tasks took longer than predicted, which is not surprising because it is hard to estimate the duration of such tasks, especially when developing models that were unknown or that we lacked experience with, meaning that, for some classifiers, we had to search for guides on their usage and experiment different approaches to reach the final implementation. Furthermore, additional experiences and models were set during the course of the dissertation that were not included when the original work plan was developed.

The process for completing each task (besides Documentation) will be explained in the next Chapters.

Chapter 5

Proposed Approach and Dataset Curation

In this chapter, we present the general Machine Learning (ML) approach used (Figure 5.1) and focus on the work regarding the Dataset Curation, including the analysis, creation, and annotation of datasets, along with the required preprocessing and a summary of existing options. The next steps of our approach will be presented in Chapters 6 (Feature Engineering and Model Training and Optimization) and 7 (Model Evaluation).

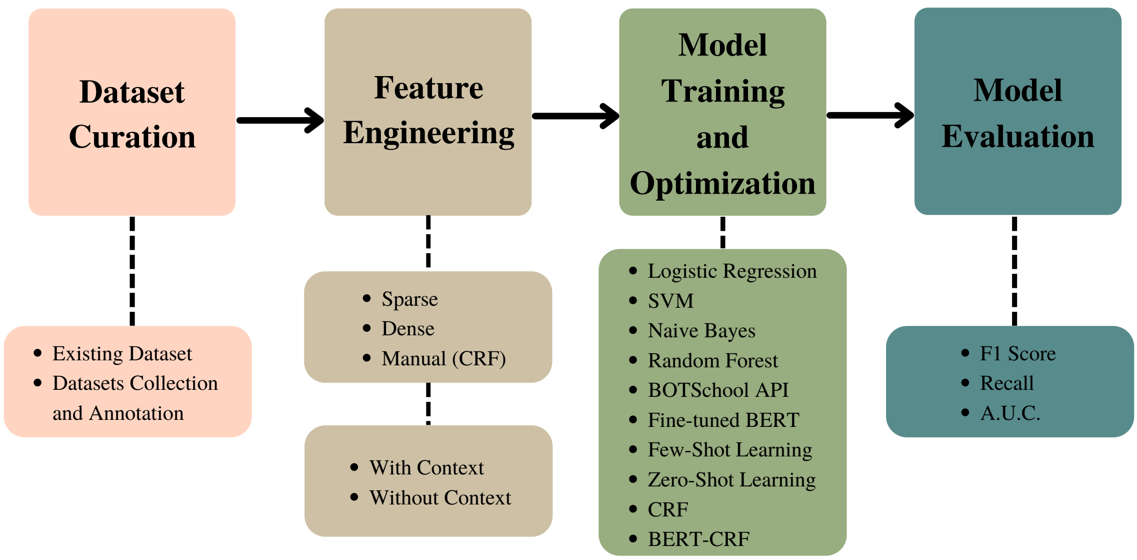


Figure 5.1: General ML Approach

Each step in the Figure is linked to a list that describes the most important aspects of each part of our approach. In the Dataset Curation, the list presents Existing Dataset and Datasets Collection and Annotation, because, as seen in Section 3.4, the available datasets were either in English, not annotated, or did not contain dialogues. Hence, there was a need to either adapt an existing dataset or create a new one.

5.1 Existing Datasets

This section focuses on the two existing datasets that were considered in the development of this work, SentiTuites (Carvalho et al., 2011) and Friends' Emotion Detection (Zahiri and Choi, 2018). The former was chosen because it contained data labeled for sentiment polarity and in the Portuguese language, and it could be used to evaluate whether models trained in non-dialogue data could perform well when classifying dialogues. The latter was chosen because it contained dialogues, was labeled (for emotion), and there was the option of translation using available subtitles in the Portuguese language.

5.1.1 SentiTuites

SentiTuites (Carvalho et al., 2011) is a corpus of tweets posted by Portuguese users during the 2011 election campaign. The corpus is labeled for sentiment polarity with values -1 (*Negative*), 0 (*Neutral*), and 1 (*Positive*). Twitter only allows the Tweet IDs to be shared, meaning that we needed to extract each tweet. Hence, some were unavailable (e.g. due to blocked accounts, deleted tweets), and from 30,470 tweets, we only gathered 13,538, composed of 1,531 positive tweets, 7,685 neutral tweets, and 4,322 negative tweets.

Examples of the data and annotations in this dataset are shown in Table 5.1.

Table 5.1: Examples of data from the SentiTuites dataset

Tweet	**Sentiment Polarity**
Passos Coelho está a mentir. Vai mesmo haver corte de salários e despedimentos. Sócrates deixou isto uma lástima.	-1
Até quando pretende a Esquerda manter Sócrates no Poder?	0
"Voluntários Sócrates"?!? consegue ser pior que o vídeo de boas vindas da ministra da educação...	-1
Passos é alto e bem parecido como Sócrates mas tem seguramente o nariz melhor, ou seja, não é tão comprido - Ângelo Correia	1

The evaluation of models trained with non-dialogue data will use this dataset.

5.1.2 Friends

The use of the Friends' Emotion Detection dataset (Zahiri and Choi, 2018) could be interesting for this work since it contains labeled dialogue, and there are publicly available Portuguese subtitles that could be aligned with the English data, and inherit the labels. This dataset is labeled with seven emotions (Neutral, Joyful, Peaceful, Powerful, Scared, Mad, Sad), which could be translated into sentiment polarity labels (Negative and Non-Negative) or kept as is, to explore Emotion Classification.

The English and Portuguese subtitles contained a timestamp, a line ID, and the line itself. However, there was no relation between both files, as the IDs and timestamps were different, and no pattern was found between the timestamps (i.e., the delay between the beginning of the dialogues was not constant). Table 5.2 shows these disparities, and it also showcases, in Line ID 8, that the same line can present a different number of utterances, making the detection of a relation between both files harder.

For these reasons, the Friends' Emotion Detection dataset was deemed unusable for this work.

Table 5.2: Example of disparities between English and Portuguese subtitles

Language	Line ID	Timestamp	Line	Timestamp Difference
English	2	00:00:52,969 - 00:00:55,137	Come on. You're going out with a guy.	00:00:06,837
Portuguese	2	00:00:59,806 - 00:01:01,933	Vais sair com um tipo.	
English	5	00:01:02,395 - 00:01:03,771	Wait. Does he eat chalk?	00:00:04,785
Portuguese	5	00:01:07,180 - 00:01:09,705	Ele tem uma corcunda e um capachinho?	
English	8	00:01:05,207 - 00:01:06,522	This is not even a date.	00:00:12,384
Portuguese	8	00:01:17,591 - 00:01:21,687	Vamos todos a ter calma. Isto nem é um encontro.	

5.2 Dataset Creation

There are several options for creating a dataset, as mentioned in chapter 3. This section focuses on the development of two datasets and their preprocessing. One of the datasets was provided by Altice Labs (AL) and is highly important since it contains examples of the real data to be evaluated by the classifiers. The other dataset required the extraction of data from Twitter, which will be explained in this section.

5.2.1 Altice Labs Dataset

AL provided us with data containing dialogues from their call-center service. These conversations are between a client and a chatbot, whose purpose is to detect the user's intention and correctly answer them. When this does not happen, it will redirect them to a human assistant.

The conversation is verbal, so a speech-to-text technique had to be employed to transcribe the utterances into text. This presents some limitations, since, sometimes, this technique cannot fully grasp what the client is saying, and the transcription does not represent what was said, causing it to lose its meaning. This can be due to noise in the call (sometimes from someone else speaking close to the caller), the caller's accent, the speed at which they speak, or mis-transcriptions if some words do not exist in the model's vocabulary.

Some examples where it is hard to understand what the client is saying follow:

- "estou a são delete", followed by "instalação do eyelashes";
- "ultrassom de saturação";
- "eu queria saber o meu primo do produto";

- "boa tarde menina menina fazer a favor dava-me o número do telefone do pão de ló delícia casa de pão de ló delícia".

The data also seems to contain an internal operation wrongly identified as part of the dialogue. An example of a conversation where this happens can be viewed on Table 5.3, in Turn IDs 5 to 15, and back again in Turn ID 21.

Table 5.3: Example of a conversation from the Altice Labs dataset where there seem to be non-dialogue parts.

Turn ID	Speaker	Utterance
1	USER	a minha box anda para a constantemente e eu não tenho sistema na televisão
2	BOT	Para que possamos ajudar com o problema na sua televisão, diga-nos: a televisão que apresenta problemas está ligada a uma MEOBOX? Ou é sem MEOBOX? Caso não pretenda apoio técnico para o sua TV, diga, voltar Atrás
3	USER	é o meu box
4	BOT	Para o ajudarmos estamos a analisar remotamente o estado da sua meo box. Este processo demora alguns segundos. Pedimos que aguarde e não desligue.
5	USER	resume
6	BOT	Para o ajudarmos estamos a analisar remotamente o estado da sua meo box. Este processo demora alguns segundos. Pedimos que aguarde e não desligue.
7	USER	resume
8	BOT	Para o ajudarmos estamos a analisar remotamente o estado da sua meo box. Este processo demora alguns segundos. Pedimos que aguarde e não desligue.
9	USER	resume
10	BOT	Vamos agora consultar remotamente o estado da ligação da sua meo box. Informamos mais uma vez este processo demora alguns segundos. (colocar 3 segundos silencio) Pedimos que aguarde e não desligue.
11	USER	resume
12	BOT	Vamos agora consultar remotamente o estado da ligação da sua meo box. Informamos mais uma vez este processo demora alguns segundos. (colocar 3 segundos silencio) Pedimos que aguarde e não desligue.
13	USER	resume
14	BOT	Vamos agora consultar remotamente o estado da ligação da sua meo box. Informamos mais uma vez este processo demora alguns segundos. (colocar 3 segundos silencio) Pedimos que aguarde e não desligue.
15	USER	resume
16	BOT	Confirme por favor: Na sua televisão aparecea informação X1, X5 ou X10?
17	USER	não eu tenho a imagem da televisão parada que não rtp1 e a imagem está parada
18	BOT	Vamos efetuar uma operação e a sua meobox vai reiniciar. Já efetuou anteriormente esta operação e continua com problemas?
19	USER	eu faço olha tem 7 vezes por dia tenho esta miséria não vou constantemente era para dar um serviço ou não estou disponível né algum problema
20	BOT	Vamos proceder à correcção da situação identificada. Este processo demora alguns segundos. Pedimos que aguarde e não desligue.
21	USER	resume
22	BOT	Efetuámos as operações necessárias para resolução da sua avaria. A sua meobox irá reiniciar. Daqui a alguns minutos verifique se está tudo a funcionar corretamente. Caso o problema persista ligue-nos novamente. Obrigada

The dataset resulting from the provided samples cannot be made public, as it contains real and private data (despite most private information having been replaced by placeholders). To overcome this obstacle and be able to share a similar dataset, we experimented with Generative Pre-trained Transformer (GPT)-2 and GPT-3 to generate fake data based of the real samples. However, it was not a simple task to evaluate the quality of the data generated, as there are no good automated metrics for this evaluation. In fact, Novikova et al. (2017) and Howcroft et al. (2020) argue that automated metrics only weakly reflect human judgements, and that human assessment is the most trusted form of evaluation in generated data. This means the process to obtain a labeled dataset using this data would require, first, an evaluation of the quality of the data, involving manual assessments of dimensions such as fluency (e.g., clarity, grammar use) and adequacy (e.g., semantic accuracy, coherence) by several participants, followed by a selection of the data with higher quality, and the annotation of this set of samples by, again, several participants. As this would be a very complex and time consuming task, as well as highly dependent on third-parties, we chose not to pursue this approach, leaving it for future work.

Tables 5.4 and 5.5 present some examples of the generated data, using GPT-2 and GPT-3, respectively.

Table 5.4: Example of data generated using GPT-2, based on the Altice Labs dataset.

Turn ID	**Speaker**	**Utterance**
1	USER	portabilidade de telemóvel
2	BOT	Vou transferir a chamada para um serviço CVP que irá processar o seu pedido.
1	USER	não tem pouca internet está mensalidade de pagamento
2	BOT	Pretende saber os dados pagamento do numero de onde está a ligar? Se pretende de outro numero diga outro numero, ou se não pretende saber dados de pagamento diga voltar Atrás
3	USER	do número que está a ligar
4	BOT	Lamento não ter percebido corretamente o que me disse. Pode por favor dizer por outras palavras o que pretende?
5	USER	fala para aqui uma informação sobre a faturação de dizer dos telemóveis
6	BOT	Vou transferir a chamada para o assistente comercial Genérico MXO
1	USER	a internet está ligada ao setor de faturação
2	BOT	Vou transferir a chamada para um serviço INT que irá processar o seu pedido.
1	USER	olha tenho aqui uma segunda via de um cartão eu vai não posso muito muito muito muito muito muito muito muito mas o que é que eu tenho aqui para aqui
2	BOT	Vou transferir a chamada para um assistente VIA que irá ajudar a tratar a situação

As we did not follow the data generation approach, we will only briefly explain how these samples were generated.

Regarding the GPT-2 approach, we used a model trained for the generation of Portuguese text, based on data from the Portuguese Wikipedia[1]. The process for generating data was as follows:

1. Load the AL dataset to base the generated data on;
2. Define the model and tokenizer, available at the Hugging Face website[2];
3. Fine-tune the model on the AL dataset;
4. Use the fine-tuned model to generate new data, according to a sampling algorithm (We will go over these methods when explaining the Few-Shot Learning (Few-SL) approach, in Section 6.3).

Regarding the GPT-3 approach, we needed access to OpenAI's Application Programming Interface (API)[3], so we signed up for a limited free trial. The process for generating data was as follows:

[1] `https://pt.wikipedia.org/wiki/`
[2] `https://huggingface.co/pierreguillou/gpt2-small-portuguese`
[3] `https://openai.com/api/`

Table 5.5: Example of data generated using GPT-3, based on the Altice Labs dataset.

Turn ID	Speaker	Utterance
1	USER	é assim eu gostava de saber se já tinha data de pagamento da meo se faz favor
2	BOT	Pretende saber os dados pagamento do numero de onde está a ligar? Se pretende de outro numero diga outro numero, ou se não pretende saber dados de pagamento diga voltar Atrás
3	USER	deste número
4	BOT	Tem a pagar ValorDivida até ao dia DataVencimento. Utilize a Entidade 21159 e a referência ReferenciaMB. Pretende receber os dados de pagamento por SMS?
5	USER	sim mas preciso de ajuda para meu
6	BOT	Será enviada uma SMS para o número do qual está a ligar. Posso ajudar em mais alguma questão?
7	USER	não é tudo obrigado
8	BOT	Obrigado por utilizar os nossos serviços.
1	USER	eu quero saber como é que eu posso fazer uma chamada para a frança faz favor
2	BOT	Lamentamos, mas não entendemos o que disse, pode repetir por outras palavras
3	USER	é uma chamada para a frança para a frança para a ponte para a frança para a ponte para a ponte para a ponte para a ponte para a ponte para a ponte para a ponte para a ponte
1	USER	avarias
2	BOT	Para o ajudarmos, diga-nos qual é o serviço para o qual precisa de apoio.
3	USER	televisão
4	BOT	Vamos transferir a sua chamada para o serviço AST

1. Set up the access to OpenAI's API;
2. Prepare the training data to fine-tune the model, organizing it into prompt-completion columns (e.g., user utterance, bot utterance). OpenAI allows access to a tool[4] that converts the samples into the required file format (JSONL).
3. Set up the GPT-3 model, defining the engine to use (davinci[5] being the most powerful option), and fine-tune it[6];
4. Use the model to generate new data, by sending an empty prompt to the Completion endpoint[7].

Going back to the AL data, it should also be noted that the chatbot speech is very mechanic and repetitive, which can affect the flow of the conversation, in the sense that context may not have as much effect as expected, because a mechanical response can "shut down" the client's reactions.

Further ahead in this Chapter, an analysis of this dataset will be presented, as well as examples of dialogues.

5.2.2 Twitter Dataset

Four reasons make Twitter a good option for the extraction of data for this work:

1. Allows the extraction of dialogues, based on the tweets and their respective replies;

[4] https://beta.openai.com/docs/guides/fine-tuning/cli-data-preparation-tool
[5] https://beta.openai.com/docs/models/gpt-3
[6] https://beta.openai.com/docs/guides/fine-tuning
[7] https://api.openai.com/v1/completions

2. Has a large amount of data, including in the Portuguese language;
3. Allows us to target the domains of interest (TV, eCommerce, Health Care, Finance & FinTech, Telecommunications);
4. Contains short informal messages, which have been confirmed by the collaborators of AL to be somewhat close to the format and language of the real call-center data.

The process of tweet extraction consists of several steps. First, a developer account is required and a project needs to be set up, preferably the academic research option, which requires approval from the Twitter team, but allows for more tweets to be extracted per month and has fewer restrictions.

Then, the following 19 user accounts were deemed related to the domains of interest and in the Portuguese language were selected: Altice Portugal (@altice_portugal), Deco Proteste (@decoproteste), Direção-Geral da Saúde (@DGSaude), FNAC Portugal (@fnacportugal), MEO Portugal (@MEOpt), Netflix Portugal (@NetflixPT), NOWO Portugal (@nowoportugal), Finanças Portugal (@pt_financas), Rádio Popular (@radiopopularPT), RTP (@rtppt), RTP Notícias (@RTPNoticias), RTP Play (@playrtp), Saúde Portugal (@saude_pt), SIC (@SIConline), Serviço Nacional de Saúde (@SNS_Portugal), TVI (@tvi), Vodafone Portugal (@VodafonePT), VOST Portugal (@VOSTPT), and Worten Portugal (@WortenPT).

However, some accounts (Finanças Portugal, Rádio Popular, RTP, Saúde Portugal, SIC, Serviço Nacional de Saúde, and TVI) did not engage with the users during the time of collection, so they were not useful for this work since the focus was on user-service dialogues. FNAC Portugal engaged with the users, but only three of the conversations contained more than two turns (three turns), so we decided to exclude this data as it did not add much value to the dataset.

In the tweet collection process, the Twitter API[8] was used. From each tweet, information related to the tweet itself is collected, along with information about the user. Despite 21 parameters being gathered, only 6 of them were in fact used:

- Tweet ID: The unique ID that identifies the posted tweet;
- Author ID: The unique ID that identifies the user who posted the tweet;
- In Reply To User ID: If the tweet is a reply, this field contains the ID of the author of the tweet it replies to;
- Conversation ID: One tweet can spark different conversation threads. The conversation ID matches the ID of the tweet that started the conversation, and is present in all related tweets;
- Text: The actual text from the tweet;
- Username: The unique screen name of the user.

All the parameters and their meanings are listed in Appendix 8.2.

Five types of search operations were developed for Twitter API requests, a mentions search, a username search, a tweet ID search, a keyword search, and a replies search. The

[8] `https://developer.twitter.com/en/docs/twitter-api`

first will operate on the tweets that mention a given account. The second will operate on the tweets from a given user. The third will return the tweet associated with the given tweet ID. The fourth option will return the tweets that contain a given keyword. The fifth option will return the replies from a given username. There is a sixth operation, the conversation search, that is used after any of the other operations, and looks up the tweets related to a given conversation ID.

We chose to focus on extracting tweet replies from the chosen accounts. This would ensure that we would get dialogues with a response from the service account, which was not granted with any other search option. Initially, we experimented with extracting tweets from users that mention the accounts of interest, but mentioning an account does not mean that you will get a response from it.

When performing a request, it is required to define the endpoint, and the parameters to be collected from the tweet (mentioned above). In a search by replies, the endpoint would be *https://api.twitter.com/2/tweets/search/all?query=**QUERY&PARAMS***, where QUERY is *from:/**USERNAME** is:reply*, and the USERNAME refers to the selected accounts' username, and where PARAMS sets the information we wish to retrieve, listed above. It is also possible to define the maximum number of results to return.

From the tweets retrieved by the search operation, we identify their conversation IDs. Then, through this unique identifier, the associated tweets are collected. This collection is then analysed to figure out which ones correspond to a dialogue between two users, which is done using the In Reply to User ID and Author ID properties, enabling a match between the users' tweets.

Both the total of tweets and the conversation tweets (between two users) are saved in .xlsx format, but only the conversation file was used in this work.

Some examples of retrieved conversations are presented in Figure 5.2. In the example, any sensitive data was replaced with placeholders (e.g., COMPANY whenever a business was mentioned, or PHONE_NUMBER whenever a phone number was given). In example 3 of the figure, the original tweet was deleted. This example was provided to demonstrate one of the problems with using Twitter data, which was that tweets available at the time of extraction may not be available later on, making it impossible to reconstruct the conversation.

This dataset is publicly available on GitHub[9] and lists the Tweet IDs and corresponding annotations (binary and multi-class), in accordance with the platform's license[10].

5.2.3 Datasets Preprocessing

Each dataset required some preprocessing. The AL and the generated datasets contained a prefix for each line, signaling whether the speaker was the user or the bot. These prefixes and the # that anticipated the bot's text were eliminated. This dataset required no further action.

Regarding the Twitter dataset, we used regular expressions[11], shown in Table 5.6, to detect some sequences that could be of interest - user tags, hashtags, URLs, and emotes.

[9] `https://github.com/NLP-CISUC/twitter_sentiment_analysis`

[10] Content redistribution terms: `https://developer.twitter.com/en/developer-terms/agreement-and-policy`

[11] `https://docs.python.org/3/library/re.html`

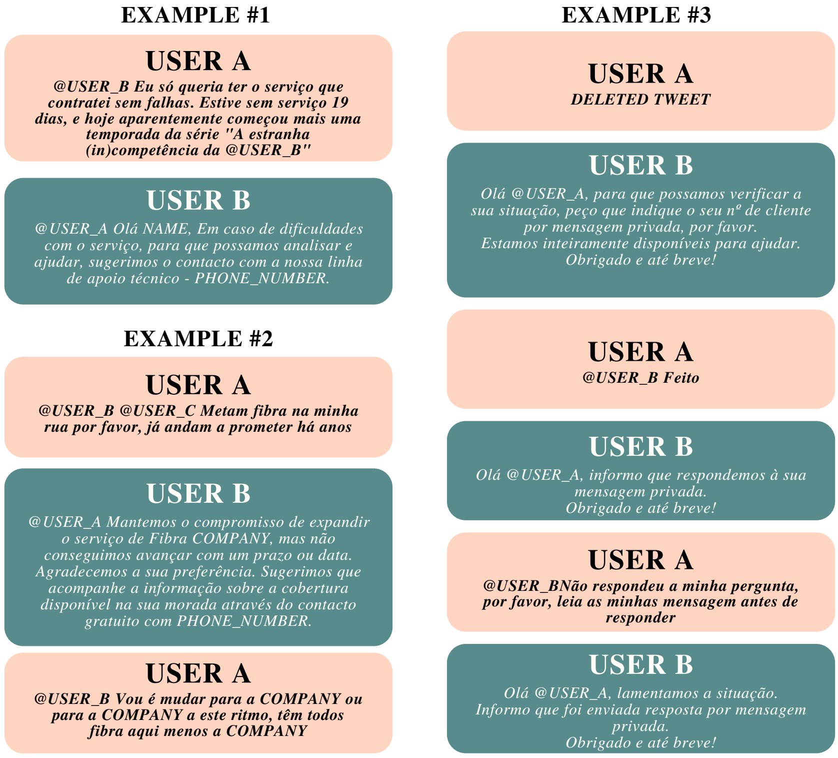


Figure 5.2: Dialogue examples collected from Twitter

Table 5.6: Regular expressions and examples for preprocessing the Twitter data

Type of Text	**Regular Expression**	**Example**
User Tag	`(\\s)?(@[\\w]+)`	@user01
Hashtag	`(\\s)?(#[\\w]+)`	#cinema
Simplified URL	`(\\s)?((http(s)?|(ftp)://)?://[^\\s]+)`	https://twitter.com/
Emote Eyes	[:;=X8]	: ; =
Emote Nose	[-oO^>'\\\\"*]{0,1}	- o >
Positive Emote	`[\\s]?\"+ Emote Eyes + Emote Nose +\"([)}DpP3]|])\"+\"\\s?`	: - D
Inverted Positive Emote	`[\\s]?\"+\"([({Cc])\"+ Emote Nose + Emote Eyes +\"\\s?`	C o :
Negative Emote	`[\\s]?\"+ Emote Eyes + Emote Nose +\"([({\\/CcSst])\"+\"\\s?`	= >S
Inverted Negative Emote	`[\\s]?\"+\"([)}\\/DSs]|])\"+ Emote Nose + Emote Eyes +\"\\s?`	} o =

The user tags and URLs were replaced by placeholders - *xUSERx* and *xURLx*. Table 5.7 presents some examples of the preprocessing mentioned (the Twitter usernames and URL were replaced with fake data for privacy's sake). The hashtags were kept unchanged since we believed they could contain relevant information, as they usually summarize the topic in discussion. We considered using emotes and emojis (detected directly) since they could be helpful for the dataset annotation - this approach will be explained ahead, in Section 5.3.

5.3 Dataset Annotation

The selected models required labeled datasets for classification. Some options for the annotation process were to base it on the use of emojis in text (Duarte et al., 2019), to use an active learning approach, to do it manually, or to do it through crowd-sourcing.

Table 5.7: Dialogues before and after preprocessing.

	Altice Labs Dataset	**Twitter Dataset**
Original Utterances	USER: casamento de chamadas	Hoje é dia de ver o último concerto de Simone de Oliveira na @canal_tv @canal depois vão ter o concerto, certo?
	BOT: #A chamada vai ser transferida para o assistente CARREGAMENTO PPP	@utilizador @canal_tv Certo! Aqui está: https://t.co/url
Preprocessed Utterances	casamento de chamadas	Hoje é dia de ver o último concerto de Simone de Oliveira na xUSERx xUSERx depois vão ter o concerto, certo?
	A chamada vai ser transferida para o assistente CARREGAMENTO PPP	xUSERx xUSERx Certo! Aqui está: xURLx

As the data extraction and development of the models took precedence and developing an active learning approach would require at least some labeled data already, we chose to focus on other parts of this work. We did not use crowd-sourcing as a paid service, but we chose acquaintances to perform this task, as we believed we could get better results by choosing people to whom we could easily explain the work and who could easily reach us for questions since sentiment is very subjective. Two approaches were experimented with, namely emoji-based annotation and manual annotation.

The exploration of the relation between Dialogue Act Recognition (DAR) and Sentiment Analysis (SA) would also require the data to be annotated for Dialogue Acts, which was not executed in order to prioritize and guarantee a decent amount of annotated samples for SA, which is the priority and objective of this work. Hence, there were no experiments performed that involved DAR, leaving this for future work.

This section will explain the approaches taken and perform an analysis of the results of the annotation process.

5.3.1 Emoji-based Annotation

The emoji-based approach could only be used in the Twitter data since the speech-to-text data does not contain emojis, which limits the application of this approach. Despite this and the fact that not all tweets contain emojis, a preliminary experience was made, with a subset of 713 tweets.

We were able to detect some emotes and emojis that could be helpful. Of the 713 tweets, 137 contained emojis or emotes, and 106 contained only emojis. There was a problem with the identification of the negative inverted emojis, such as *):* and *);*, since they were confused with the use of those symbols in listing something or just having text inside parenthesis followed by a colon. Furthermore, only about 15% of the tweets could be labeled using emojis and these labels were still subject to errors (e.g., use of irony, the emoji not reflecting the sentiment of the whole sentence). Hence, we decided not to explore this option any further and instead used manual annotation.

The emojis show in Figure 5.3 were selected based on an online emoji dictionary[12]. Some of the meanings assigned are very subjective and due to the preliminary results, we did not consider it relevant to review the dictionary.

Examples of tweets containing emotes and/or emojis can be seen in Figure 5.4, and some showcase the problems with the emote detection.

[12]`https://hotemoji.com/emoji-meanings.html`

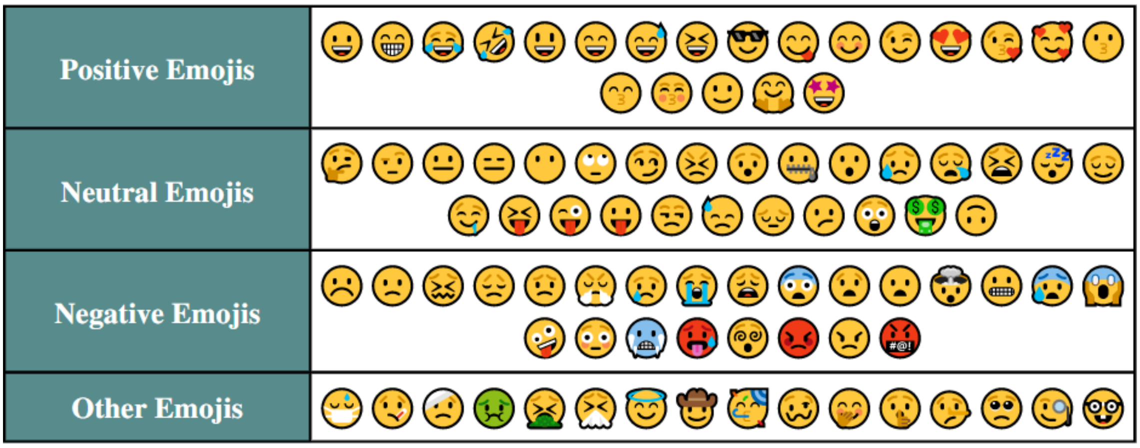


Figure 5.3: Emojis considered for emoji-based annotation

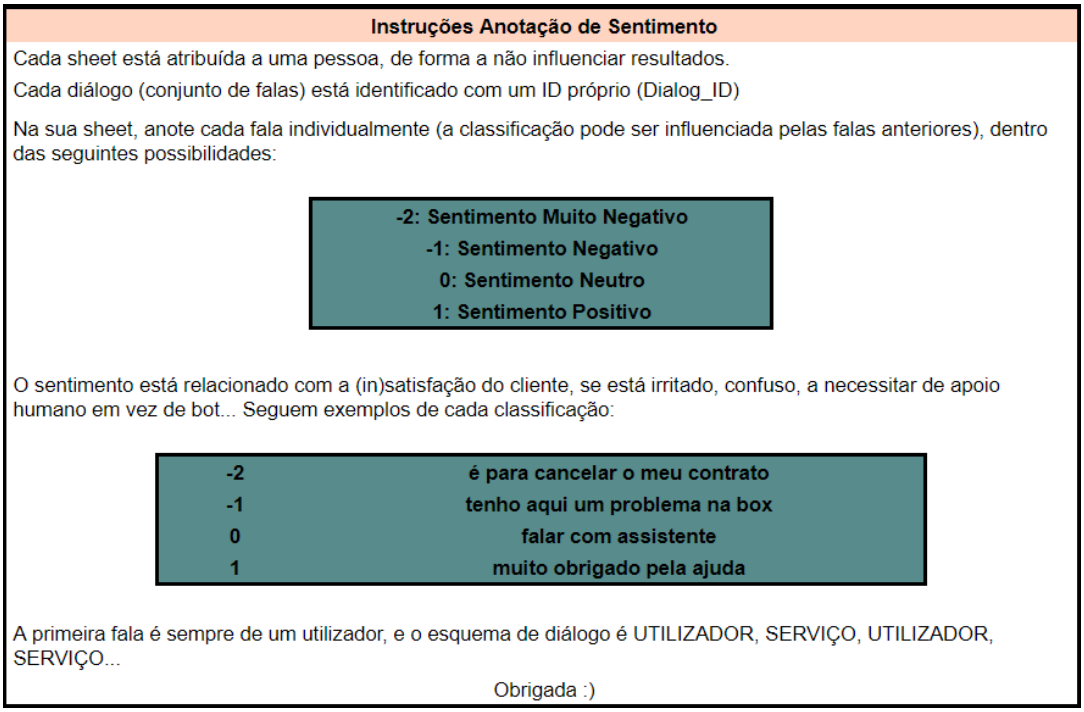
ID	Pre-processed Tweet	Tweet with Emote and Emoji Detection
1	xUSERx xUSERx xUSERx xUSERx Números. por exemplo: renda per capita, IDH, Taxa de desemprego, dentre outros. que tal fazer com o Chile também? Chile hoje (dezembro 2021): Renda per capita: US 13.230 IDH: 0,851 Taxa de desemprego: 8,4% Daqui a 4 anos a gente conversa.	xUSERx xUSERx xUSERx xUSERx Números. por exemplo: renda per capita, IDH, Taxa de desemprego, dentre outros. que tal fazer com o Chile também? Chile hoje (dezembro 2021 _EMOTE_NEG_INV_ Renda per capita: US 13.230 IDH: 0,851 Taxa de desemprego: 8,4% Daqui a 4 anos a gente conversa.
2	xUSERx xUSERx Tenho um excelente contacto na MEO :)	xUSERx xUSERx Tenho um excelente contacto na MEO _EMOTE_POS_
3	xUSERx xUSERx xUSERx xUSERx Desculpa não te ter entendido à primeira! Então a tua opinião profissional é que no que toca a quem tem cidadania portuguesa, não há dúvidas: têm direito aos testes comparticipados independentemente da sua morada ser em Portugal ou no estrangeiro?	xUSERx xUSERx xUSERx xUSERX Desculpa não te ter entendido à primeira! Então a tua opinião profissional é que no que toca a quem tem cidadania portuguesa, não há dúvida _EMOTE_NEG_INV_ têm direito aos testes comparticipados independentemente da sua morada ser em Portugal ou no estrangeiro?
4	xUSERx xUSERx xUSERx xUSERx xUSERx Isso, muito contentes, até porque a tap continua a ser nossa, a cp idem, e o salário mínimo continua a crescer por decreto 😬	xUSERx xUSERx xUSERx xUSERx xUSERx Isso, muito contentes, até porque a tap continua a ser nossa, a cp idem, e o salário mínimo continua a crescer por decreto _EMOJI_NEG_
5	xUSERx xUSERx xUSERx xUSERx xUSERx xUSERx xUSERx xUSERx xUSERx Não tive essa ideia brilhante 😂	xUSERx xUSERx xUSERx xUSERx xUSERx xUSERx xUSERx xUSERx xUSERx Não tive essa ideia brilhante _EMOJI_POS_

Figure 5.4: Examples of tweets containing emotes and/or emojis.

5.3.2 Manual Annotation

The process of manual annotation required some manpower and the development of some guidelines for assistance. A total of 14 people were involved in the annotation, distributed between 15 data blocks. For each data block, there were three annotators, and each block contained approximately 100 dialogues. Some Twitter conversations were removed if the dialogues did not suit the problem, were very repetitive, or the accounts often just replied to themselves, which is why some blocks contained less than 100 dialogues. The guidelines presented the expected labels, an example of a sentence for each label, and a description of the general goal and structure of this process. The annotators labeled the samples independently, not considering the other annotators' classifications. They were asked to classify the utterances as one of four possible sentiments: Very Negative (-2), Negative (-1), Neutral (0), and Positive (1).

Figure 5.5 presents the guidelines for the annotation of AL' data. There is only one degree for positive sentiment because it was not often that this occurred, and we did not feel that it showed up strongly enough in the data to justify having two degrees for the positive sentiment. Since the AL dataset contained situations in which the client could not get their answer quickly, and had to repeat themselves, we asked the annotators to consider this, as it builds up stress, and to decrease the sentiment in the repeated utterances.



Instruções Anotação de Sentimento

Cada sheet está atribuída a uma pessoa, de forma a não influenciar resultados.

Cada diálogo (conjunto de falas) está identificado com um ID próprio (Dialog_ID)

Na sua sheet, anote cada fala individualmente (a classificação pode ser influenciada pelas falas anteriores), dentro das seguintes possibilidades:

-2: Sentimento Muito Negativo
-1: Sentimento Negativo
0: Sentimento Neutro
1: Sentimento Positivo

O sentimento está relacionado com a (in)satisfação do cliente, se está irritado, confuso, a necessitar de apoio humano em vez de bot... Seguem exemplos de cada classificação:

-2	**é para cancelar o meu contrato**
-1	**tenho aqui um problema na box**
0	**falar com assistente**
1	**muito obrigado pela ajuda**

A primeira fala é sempre de um utilizador, e o esquema de diálogo é UTILIZADOR, SERVIÇO, UTILIZADOR, SERVIÇO...

Obrigada :)

Figure 5.5: Guidelines for the annotation of Altice Labs' data

Since there were 3 annotators for each data block, we determined the median to assign a final label to each sentence in the dialogues. We considered using the mean, the mode, and the median for this purpose but concluded that using the mean was not the best option, since the classification scale is similar to Likert scales since the data is ordinal (Very Negative, Negative, Neutral, Positive). Sullivan and Artino Jr (2013) state that an average of, for example, "never" and "rarely" has no useful meaning. Furthermore, if responses are clustered at the high and low extremes, the mean may appear to be the neutral response, which does not fairly characterize the data. In the context of this dissertation, this could happen if, for example, there is irony in an utterance, and one annotator labels it as Very Negative (-2) while the other two label it as Positive (1), obtaining the mean value of 0, a Neutral sentiment, which is false by the annotators' interpretations. Hence, the mean is not a helpful measure for this kind of data. We chose the median because annotators could all pick different labels, meaning there would be no mode. In this situation, we would have used the median, but since there were only three annotators, meaning three values for each sentence, then using the median all the time would be the same as using a combination of mode and median. This is true because if there was a mode (i.e., two or three equal labels), then the median would be the same value, as the mode value would be in the middle position.

We prioritized the real data provided by AL and the Twitter data over the generated data. This happened because the real data was essential for the evaluation of the models, and the Twitter data was more diverse than the generated data. Due to the time and manpower required, only these two datasets were annotated.

5.3.3 Annotation Analysis

An analysis of the data collected and annotated allows for the comparison of the created datasets (between themselves and existing datasets mentioned in 3.4), and provides a perception of the dimension and variety of the samples.

From the annotation process, there were four possible values for each sentiment, but they were not equally distributed due to the data's nature. Tables 5.8 and 5.9 shows the frequency of each final label, using either the mean or the median to determine this value, for each dataset.

Table 5.8: Sentiment frequency for the annotated AL dataset.

Altice Labs Dataset					
Mean			**Median**		
Sentiment	Absolute Frequency	Relative Frequency	Sentiment	Absolute Frequency	Relative Frequency
Very Negative	37	0.7%	*Very Negative*	42	0.8%
Negative	948	17.8%	*Negative*	930	17.5%
Neutral	4,299	80.9%	*Neutral*	4,311	81.2%
Positive	28	0.5%	*Positive*	29	0.5%
Total	**5,312**	**100%**	**Total**	**5,312**	**100%**

Table 5.9: Sentiment frequency for the annotated Twitter dataset.

Twitter Dataset					
Mean			**Median**		
Sentiment	Absolute Frequency	Relative Frequency	Sentiment	Absolute Frequency	Relative Frequency
Very Negative	131	12.4%	*Very Negative*	138	13.1%
Negative	400	37.9%	*Negative*	310	29.4%
Neutral	468	44.4%	*Neutral*	551	52.2%
Positive	56	5.3%	*Positive*	56	5.3%
Total	**1,055**	**100%**	**Total**	**1,055**	**100%**

Through the analysis of Tables 5.8 and 5.9, we can confirm that the data is not balanced, especially in the AL dataset, where 81.7% of the data is Non-Negative (Positive or Neutral). Due to the low frequency of *Very Negative* and *Positive* sentiments, and the main goal being to detect when the sentiment is negative, it was decided to convert this multi-class problem into a binary problem, by merging the sentiments in two final classes, *Negative* (**0**) and *Non-Negative* (**1**). This conversion makes the Twitter dataset nearly balanced, containing 57.5% of Non-Negative samples and 42.5% of Negative samples.

A visualization of the distribution of the data can be helpful to preliminarily assess how difficult it might be to separate the samples in the defined categories, using two dimensions (for example). With this in mind, we used the visualization technique mentioned in Section 2.1.2, t-distributed Stochastic Neighbor Embedding (t-SNE), a non-linear technique to project the sentence embeddings from the testing dataset (using an 80%-20% split of the data) into a two-dimensional space. The embeddings used were obtained with the Term Frequency - Inverse Document Frequency (TF-IDF) and the BERTimbau Natural Language Inference (NLI)[13] encoder techniques, and they are represented in Figure 5.6 for the Twitter dataset, and in Figure 5.7 for the AL dataset. This feature reduction technique was not used in the development of any of the classification algorithms, which is

[13] `https://huggingface.co/ricardo-filho/bert-portuguese-cased-nli-assin-assin-2`

why it serves only a visualization purpose and allows us to verify that the representation of sentences in a low dimension can be challenging as both labels seem to overlap often, especially in the AL dataset.

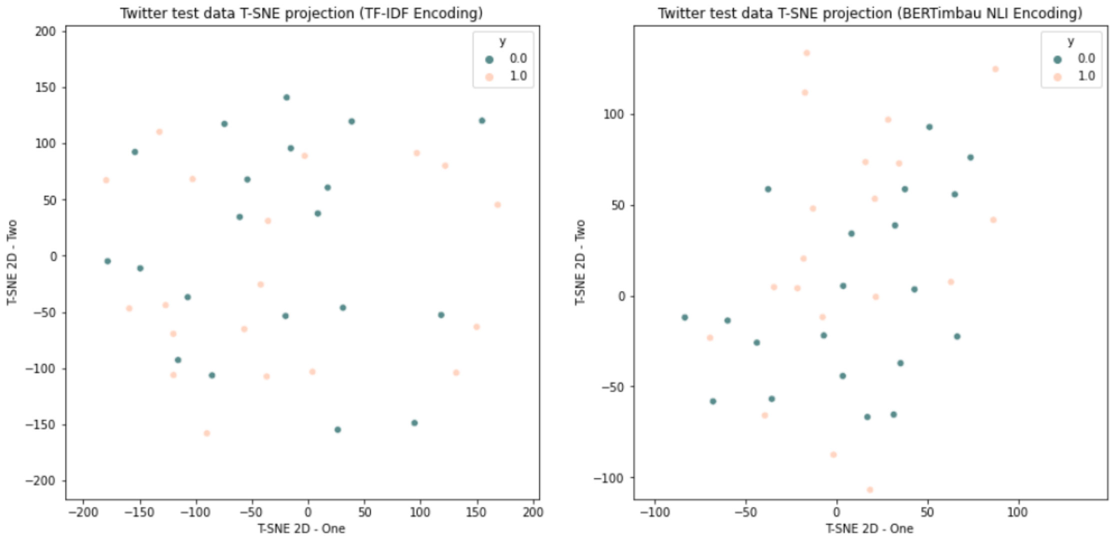


Figure 5.6: Projection of the Twitter testing dataset using t-SNE.

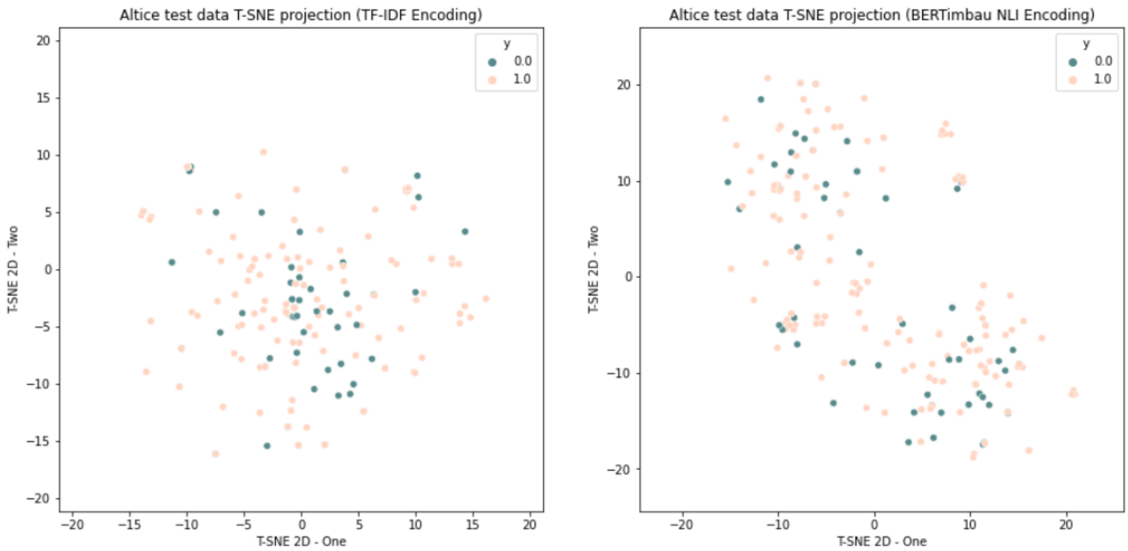


Figure 5.7: Projection of the Altice Labs testing dataset using t-SNE.

Regarding the Twitter data, it should be mentioned that the labeled dataset does not contain dialogues from all the selected accounts since we could not annotate all the extracted conversations. The final dataset contains dialogues from the NOWO Portugal, RTP Play, RTP Notícias, Vodafone Portugal, MEO Portugal, Netflix Portugal, Altice Portugal, Deco Proteste, and Direção-Geral da Saúde. Some of the Deco Proteste and Netflix Portugal's data was removed because the dialogues were very repetitive, did not always fit the domains of interest, or because the tweets in the conversation all belonged to the service account. The VOST Portugal annotated data was fully removed because it ended up relating only to weather forecasts and warnings, which did not fit any of the domains of interest.

It was important to evaluate the reliability of the annotation process. In order to accomplish this, we determined inter-rater reliability metrics (i.e., the degree of agreement between annotators) for every data block, considering each annotator's classification. Several works have used metrics for the data's reliability (Carvalho et al., 2011, Cerisara et al., 2018, Li et al., 2017), and Zapf et al. (2016) considered the following metrics the most generalized measures for agreement:

- Cohen's Kappa Coefficient (Cohen, 1960): considers only pairs of annotators, and the observed agreement is corrected for the agreement expected by chance. A score of 0 would mean a random agreement, while a score of 1 would mean a complete agreement. We omitted the Cohen's Kappa values because the pairs of annotators were not fixed and not every rater labeled every single data block;
- Fleiss' Kappa Coefficient (Fleiss, 1971): an extension of Cohen's Kappa for three raters or more, with the assumption that the raters can be randomly chosen instead of fixed, which better fits the scenario of our work;
- Krippendorff's Alpha Coefficient (Krippendorff, 1970): in contrast with the previous metrics, this value is based on the observed disagreement corrected for disagreement expected by chance. This Alpha coefficient can handle a high number of raters and ratings, and even incomplete datasets. Although we had complete datasets, we still considered it an interesting metric to use.

Values between 0.41 and 0.60 reveal moderate agreement between the annotators for the Fleiss and Cohen's metrics, while for the Krippendorff's metric this level of agreement is given by values between 0.6 and 0.79. Higher values mean a good or very good level of agreement.

Finally, for each dataset, we ended up with the characteristics shown in Table 5.10. The agreement scores were calculated for the multi-class and binary scenarios.

Table 5.10: Analysis of the two datasets used

Dataset	#Dialogues	#Turns	Avg. Turns Per Dialogue	#Negative Samples	#Non-Negative Samples	Avg. Fleiss	Binary Avg. Fleiss	Avg. Krippendorff	Binary Avg. Krippendorff
Twitter	418	1,055	2.52	448	607	0.48 ±0.13	0.67 ±0.16	0.66 ±0.16	0.67 ±0.16
Altice Labs	1,000	5,312	5.32	972	4,340	0.56 ±0.17	0.62 ±0.17	0.61 ±0.15	0.67 ±0.16

While the AL dataset is larger than the Twitter dataset, the latter is more balanced. Despite the average agreement levels being considered moderate for both datasets, it should be noted that these values refer to four different classes. Considering the binary setup, as used in the experimentation, the level of agreement increases by 19% in the Twitter data for the Fleiss metric, but only slightly (1% and 6%) in the other evaluations and dataset.

Figure 5.8 and Table 5.11 present some examples of dialogues in each annotated dataset, using the multi-class annotation.

Speaker	Turn Number	Utterance	Sentiment
USER	1	@SERVICE Estou com velocidade de upload de internet baixíssima (1MB). Tentei ligar para a linha de apoio ao cliente e falam que há anomalias que afetam todos serviços. Esta minha situação está relacionada com essa anomalia? Obrigado	-1
SERVICE	2	Olá @USER, respondemos à sua mensagem privada. Obrigado e até breve!	0
USER	1	O serviço da @SERVICE há uma semana que está uma 💩! Se antes eu andava com os dados móveis ligados em casa por falta de wi-fi, agora é a box 🤦‍♀️ agradeço que isto seja resolvido de uma vez por todas senão só perdem clientes (e eu que sempre recomendei a operadora). 🤦‍♀️	-2
SERVICE	2	@USER Olá. Pedimos que reinicie os equipamentos. Caso as dificuldades persistam, será necessário o contacto com a equipa técnica para testes em linha, ligando PHONE_NUMBER ou PHONE_NUMBER. Se tiver um nº móvel Vodafone, pode ligar gratuitamente pelo My Vodafone (https://t.co/lxyXviTSBs).	0
USER	3	@SERVICE Já fizemos tudo o que a equipa técnica nos disse e inclusive voltei a reiniciar e continua sem funcionar portanto vou ter que contactar novamente a equipa técnica! Obrigada!	-2
SERVICE	1	A cobertura vacinal das crianças até um ano atingiu 99% em 2021 e ultrapassou, até aos sete anos, a meta de 95% do Programa Nacional de Vacinação (PNV) da Direção-Geral da Saúde. Os resultados são fruto do trabalho das equipas e da adesão da população à vacinação. #DGS	0
USER	2	@SERVICE Sensacional !! Um bom trabalho gera bons resultados ! Triste ver em outros lugares, a vacinação perdendo a adesão da população, devido a falsas informações e ações de grupos antivacinas!! 👏👏👏	1

Figure 5.8: Examples of dialogues from the Twitter data

Table 5.11: Examples of dialogues from the Altice Labs data

Speaker	Turn Number	Utterance	Sentiment
USER	1	os dados para pagamento	0
SERVICE	2	Pretende saber os dados pagamento do numero de onde está a ligar? Se pretende de outro numero diga outro numero, ou se não pretende saber dados de pagamento diga voltar Atrás	0
USER	3	dados de pagamento deste número	0
SERVICE	4	Tem a pagar ValorDivida até ao dia DataVencimento. Utilize a Entidade ENTITY e a referência ReferenciaMB. Pretende receber os dados de pagamento por SMS?	0
USER	5	sim quero	0
SERVICE	6	Será enviada uma SMS para o número do qual está a ligar. Posso ajudar em mais alguma questão?	0
USER	7	não é tudo obrigado	1
SERVICE	8	Obrigado por utilizar os nossos serviços.	1
USER	1	oi eu quero saber porque é que este mês pago uma fatura tão grande 25 e 76 eu nunca paguei isto	-1
SERVICE	2	Lamentamos, mas não entendemos o que disse, pode repetir por outras palavras	-1
USER	3	outras palavras é isso é sobre a faturação deste mês como apareceu aquilo essa fatura 2576 e eu nunca paguei este valor e diz que não sei o que é que foi	-1
SERVICE	4	Pretende saber os dados pagamento do numero de onde está a ligar? Se pretende de outro numero diga outro numero, ou se não pretende saber dados de pagamento diga voltar Atrás	0
USER	5	queres saber o nosso pagamento sim porque é que eu estou a pagar isso pá caraças pá	-2
SERVICE	6	Tem a pagar ... por débito bancário até ao dia.... Posso ajudar em mais alguma questão?	0
USER	7	olha essa questão saiu que tenho aqui na fatura eu quero saber o porquê o porquê o porquê dessa fatura estar tão elevada que nunca	-2
SERVICE	8	Obrigado por utilizar os nossos serviços.	0
USER	1	não consigo estou com problemas nos meus telemóveis e não consigo estabelecer ligação para lado nenhum	-1
SERVICE	2	Vou transferir a chamada para o assistente comercial Genérico MXO	0

Chapter 6

Experimental Setup

In this chapter, we will focus on the Feature Engineering and Model Training and Optimization steps presented in Figure 5.1. The experiments performed for each classifier and the approaches taken for achieving the objectives of this work will be explained and analysed. The experimental setup will also be presented.

The exploration of different solutions to the classification problem began with lexicon-based approaches that could be used as baselines. All other experiments used Machine Learning (ML) approaches, ranging from shallow to deep classifiers (transformers-based). The models were chosen based on the approaches used in the related work (e.g., Support Vector Machine (SVM), Conditional Random Field (CRF), Fine-tuned BERT) or for being common models with different types of approaches to the problem (e.g., Logistic Regression, Random Forest).

With the exception of the lexicon-based approach, the CRF model, and the combined Bidirectional Encoder Representations from Transformers (BERT) and CRF model (BERT-CRF), every other classifier was evaluated in two ways: with and without context, since, for the objective of this work, context may be relevant in the progression of sentiment. It is important to stress that, in these scenarios, by context, we mean that the classifiers received as input a concatenation of the current sentence with, at most, the two previous sentences in the dialogue if they existed. We say at most because some models present limitations on the length of the input. In these cases, the input may present only the current sentence or the current and previous sentences. In the eventuality that the current sentence in itself is too large, it is cut until it reaches the maximum length allowed.

Furthermore, the use of context could be considered non-standard for Sentiment Analysis (SA), as, when applied to text, it usually refers to a single sentence. With this in mind, it could be interesting to see how standard SA techniques perform in a non-standard setting (the use of dialogues/context).

Besides the experiment and comparison of the performances of each classifier, we will also report the following experiences:

- The use of different representations (sparse vs dense) in the shallow classifiers and the CRF model. Note that while we defined the feature engineering as manual for the CRF model, in Figure 5.1, the chosen features include the sparse or dense representations of the utterances;

- The use of unigrams, bigrams, and trigrams in the shallow classifiers and CRF model with sparse representations;

- The use of a holdout set vs cross-validation (including hyperparameters tuning), when applicable;
- The use of more or fewer features in the CRF model;
- The use of different representation functions (i.e., ways to obtain the sentence embedding) in the BERT-CRF model;
- The use of hypotheses in Portuguese, English, and as labels or sentences, in the Zero-Shot Learning (Zero-SL) model;
- Whether models trained using non-dialogue data were able to properly classify dialogues.

The evaluation of the classifiers, using the metrics presented in section 2.2, will be presented in Chapter 7.

Regarding the experimental setup, the programming language chosen was Python, due to its simple syntax, platform independence, community support, readability, and mostly due to the great choice of libraries. It is important that the experiences performed can be reproducible, meaning that any randomness must be controlled, otherwise the data and even some of the algorithms will not be defined as they were for this work. Hence, we defined a seed state (123) to initialize the random number generator, making sure that the results are reproducible. When evaluating the models on the testing dataset, they were not retrained on the combined training and validation datasets, however this approach could possibly affect the results, and could be used in future work.

In this work, several libraries were used and they are listed in appendix 8.2. Some of them include: For data structures, pandas[1] and numpy[2]. For data visualization, seaborn[3] and matplotlib[4]. For Natural Language Processing (NLP), nltk[5] and spacy[6]. For machine learning, scikit-learn[7]. For CRF, sklearn-crfsuite[8]. For deep learning, tensorflow[9] and torch[10]. For transformers, there are several options in Hugging Face[11].

6.1 Lexicon-based Approaches

A lexicon-based approach is a traditional and quick option for SA, which is why they were considered baseline models for this work.

The SentiLex-flex (Carvalho and Silva, 2015), and the NRC Emotion (Mohammad and Turney, 2013) lexicons were used. The former is based on Portuguese texts and uses the words' inflected forms, and the latter is composed of an English Thesaurus' unigrams and bigrams translated to Portuguese. SentiLex-flex is likely to be the better option since it is based on Portuguese texts and focused on sentiments towards humans.

[1] https://pandas.pydata.org/
[2] https://numpy.org/
[3] https://seaborn.pydata.org/
[4] https://pypi.org/project/matplotlib/
[5] https://www.nltk.org/api/nltk.html
[6] https://spacy.io/
[7] https://scikit-learn.org/stable/
[8] https://sklearn-crfsuite.readthedocs.io/en/latest/
[9] https://www.tensorflow.org/
[10] https://pytorch.org/docs/stable/package.html
[11] https://huggingface.co/models

For this approach, the following steps were followed:

1. Create a dictionary based on each lexicon, containing the words and associated sentiment polarity;
2. Transform multi-class labels in binary (Non-Negative and Negative), in accordance with the datasets' annotations;
3. Find the words in each sentence that are present in the lexicon and sum their lexicon sentiment polarities;
4. Determine the final polarity of a sentence by the sentiment polarities of the composing words;
5. Analyse the results, applying the evaluation metrics defined.

As a baseline, the performance of these models was not expected to be high. The results will be presented in Chapter 7. Table 6.1 presents some examples of entries from each lexicon.

Table 6.1: Examples of entries and annotations from each lexicon

Lexicon	**Entry**	**Sentiment**
SentiLex-flex	ruinosas	-1
	entusiasmam	1
	senhor do seu nariz	0
NRC Emotion	à deriva	-1
	perspicácia	1
	permitido	0

6.2 Shallow Learning Classifiers

In this work, we used the following Shallow ML classifiers, that were presented in section 2.2.5:

- **Logistic Regression**, a simple linear classifier;
- **SVM**, a classifier that splits the data using an hyperplane. Different kernels were used, and only the one with higher performance will be presented in the results;
- **Naive Bayes**, a classifier based on the Naive Bayes theorem (equation 2.9). We chose the Multinomial Naive Bayes[12], since it is suited for text classification and handles Term Frequency - Inverse Document Frequency (TF-IDF) counts;
- **Random Forest**, an ensemble of decision trees that automatically performs feature selection.

[12] `https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html`

For each classifier, two types of representation were used, TF-IDF (Sparse) and encodings generated by BERTimbau Natural Language Inference (NLI)[13] (Dense). The former is a simple and traditional technique, while the latter is a Sentence Transformer (Reimers and Gurevych, 2019), a modification of the pre-trained BERT model developed for application with sentences. This type of transformers are fine-tuned on pairs of sentences (and labels), regarding the chance of inferring one based on the other. BERTimbau NLI is the only model of this type trained in the Portuguese language and available at Hugging Face.

For this approach, the following steps were followed:

1. Load and split the data, using a 60%-20%-20% split or cross-validation;
2. Define the model and tokenizer or TF-IDF vectorizer, according to the type of encoding used;
3. Encode the data and train the model;
4. Evaluate the model in the validation and testing datasets;
5. Analyse the results, applying the evaluation metrics defined.

6.3 Deep Learning

The following deep learning approaches, which do not consider context, were used in this work:

- **BOTSchool Application Programming Interface (API)**:

  BOTSchool[14] is a platform developed by Altice Labs (AL)[15] to create Virtual Assistants that can be used for companies to increase their customer experience and engagement, from Support, Shopping Assistant, Self-Care, Management Assistant, among others.

  AL allowed us access to their sentiment analysis API, which was built using a fine-tuned multilingual BERT model trained on 198M Tweets[16] (Barbieri et al., 2022). This approach is relevant for this work since it sets a performance mark that, if surpassed by other models, could mean there might be better options for the system created by AL. To use this model, a request is sent with the sentence, and the probabilities of it containing a negative, neutral, or positive sentiment polarity are returned.

  For this approach, the following steps were followed:

  1. Load and split the data using an 80%-20% ratio. This model is already fine-tuned in sentiment data and does not require additional training. The testing dataset contains the same data as the testing datasets from the models that require training, so the evaluation is performed on the same utterances for all classifiers;

[13] `https://huggingface.co/ricardo-filho/bert-portuguese-cased-nli-assin-assin-2`
[14] `https://botschool.ai/home`
[15] `https://www.alticelabs.com/`
[16] `https://huggingface.co/cardiffnlp/twitter-xlm-roberta-base-sentiment`

2. Define the model's endpoint (`https://api.botschool.ai/sentiment/v1/polarity?apiKey=`*API_KEY*) and request parameters. API_KEY refers to the access key provided by AL so that we could use the BOTSchool API;
3. Post requests to the API, containing one sample at a time;
4. Retrieve the probabilities of the sample containing Negative, Neutral, or Positive sentiment, and transform them into Negative and Non-Negative labels;
5. Analyse the results, applying the evaluation metrics defined.

AL has used this model in a development setting, but, so far, it has not been applied to their live services.

Regarding the training of this model, we note that the dataset used contains standard SA data, which may impact the model's performance in non-standard data, as is the case of dialogues.

- **Fine-tuned BERT**

  A fine-tuning approach consists of training a pre-trained model on the SA task, which will update the model weights. Since there is a BERT model pre-trained for the Portuguese language, BERTimbau[17] (Souza et al., 2020), there was a good chance that the use of this model fine-tuned for SA could provide good results.

  For this approach, the following steps were followed:

  1. Load and split the data, using a 60%-20%-20% split or cross-validation;
  2. Use the BERTimbau Tokenizer to generate initial embeddings for each dataset, with padding to the maximum length found in the training dataset to ensure equal length in all samples;
  3. Define the model and optimizer[18] (i.e., the optimizer is responsible for changing the weights of the model in order to minimize the loss function, in this case);
  4. Train the BERTimbau model in batches of size 16 (the model is updated at every 16 samples processed) and for two epochs (times that the model will work through the entire training dataset). At the end of each epoch, the model is evaluated using the validation dataset;
  5. Evaluate the model on the testing dataset and analyse the results, applying the evaluation metrics defined.

- **Few-Shot Learning**

  Few-Shot Learning (Few-SL) aims to train classifiers given only a few labeled examples of each class. In this work, three examples were fed to the Generative Pre-trained Transformer (GPT)-Neo[19] model.

  GPT-3[20] and GPT-J[21] were also considered, but were not applied due to limited or paid usage. Technical and time constraints did not allow us to experiment with Meta-OPT[22], which could be an interesting model. The same reasons did not allow the experimentation with the inclusion of a task description (Mi et al., 2022) in the examples selected. The experiments could have included descriptive sentences, presenting a definition, a constraint, and a prompt, such as:

[17] `https://huggingface.co/neuralmind/bert-base-portuguese-cased`
[18] `https://pytorch.org/docs/stable/generated/torch.optim.AdamW.html`
[19] `https://huggingface.co/EleutherAI/gpt-neo-2.7B`
[20] `https://openai.com/api/`
[21] `https://huggingface.co/docs/transformers/model_doc/gptj`
[22] `https://huggingface.co/facebook/opt-2.7b`

- Definition: Predict the sentiment in the input sentence;
- Constraint: Select the most suitable value from 0;1. If multiple values appear, select the latest one;
- Prompt: What is the sentiment?

Besides the choice of the model and how to describe the task, several algorithms can be used in the decoding operation (i.e., analysis and selection of the most likely tokens to generate):

- Greedy Search: Selects the word with the highest probability as the next word to generate at each timestep;
- Sampling Decoding: Randomly selects the next word according to its conditional probability distribution;
- Beam Search: Keeps the most likely N (defined by the number of beams) hypotheses at each timestep, choosing the one with the highest overall probability;
- Top-K Sampling (Fan et al., 2018): Filters the K most likely next words and redistributes the probability mass among those K words. This algorithm was adopted by GPT2;
- Top-P (Nucleus) Sampling: Filters the smallest possible set of words with a cumulative probability higher than p, and redistributes the probability mass along that set of words;
- Top-P and Top-K Sampling: A combination of the Top-P and Top-K algorithms, which can avoid low-ranked words while allowing some dynamic selection.

Some of these algorithms use a parameter called temperature, which affects how it will choose the next word to generate (from a distribution). A lower temperature value will decrease the likelihood of low probability words while increasing the likelihood of high probability words). The algorithms are further explained in Hugging Face's website (von Platen, 2020). Table 6.2 presents the execution time for a randomly selected sentence using each of the mentioned decoding algorithms. With this in mind, the Top-P and Top-K Sampling algorithm was selected for the Few-SL experiments.

Table 6.2: Exploration of Few-Shot Learning's Sampling Algorithms

Sampling Algorithm	Sampling Decoding (Temperature)	Greedy Decoding	Beam Search	Top-K Sampling	Top-P (Nucleus) Sampling	Top-P and Top-K Sampling	Top-K Sampling (with Definition, Constraint, Prompt)
Execution Time (min)	17.28	17.28	Not Determined	0.57	0.55	0.55	4.34

As mentioned in Section 2.2.5, a Few-SL approach can generate tokens that may not always match the desired classifications (0 or 1). This analysis will be included and presented in Chapter 7.

For this approach, the following steps were followed:

1. Load and split the data using an 80%-20% ratio. This model does not require training, so we only use the testing dataset;
2. Define the model and tokenizer;
3. Set three examples, according to each dataset, that contain negative and non-negative sentiment. To each example we add separation tokens (|bos| to define the start of the utterance, |eos| to define the end of the utterance, and |sentiment| to append the associated label at the end);

4. Generate and decode the answer from each input from the testing data at a time, using the chosen decoding algorithm (Top-P and Top-K). The separation tokens are also added to the input, but no associated label is appended to the |sentiment| token so that the model knows how to classify it;
5. Process the answers, verifying which ones are valid (were given a 0 or 1 label), and analyse the results on the valid data, applying the evaluation metrics defined.

For better visualization of the use of the separation tokens, we present two examples of utterances from the Twitter dataset:

1. |bos| _USER_ Já fizemos tudo o que a equipa técnica nos disse e inclusive voltei a reiniciar e continua sem funcionar portanto vou ter que contactar novamente a equipa técnica! Obrigada! |eos| |sentiment| 0
2. |bos| _USER_ por favor, não deixes de existir! (_URL_ _URL_ |eos| |sentiment| 1

- **Zero-SL**

  Zero-SL aims to perform classification without using labeled data. In this work, the BART model[23] was used. However, it was trained on data in the English language, which may add an extra layer of difficulty to the task.

  For this task, hypotheses in the Portuguese and English languages were used, as a single word (e.g. Negative, Positive) and as a sentence (e.g. The sentiment in this sentence is negative).

  For this approach, the following steps were followed:

  1. Load and split the data using an 80%-20% ratio. This model does not require training, so we only use the testing dataset;
  2. Define the model and tokenizer;
  3. To each sample in the testing dataset, add separation tokens (|bos| to define the start of the dialogue, |eos| to define the end of the dialogue, and |sentiment| at the end of the dialogue). The |bos| and |eos| tokens were not required for this approach, but were kept for coherence with the Few-SL approach;
  4. Define the hypothesis, either as a single label (e.g., Negative) or as a sentence (e.g., Esta frase tem sentimento negativo) and in Portuguese or English language;
  5. For each hypothesis and input, retrieve the probability of the hypothesis being true;
  6. Normalize the probabilities of the set of hypotheses (since the model considers semantic, we used Negative, Neutral, and Positive labels, because we believe they could bear more meaning than just using Negative and Non-Negative labels);
  7. Analyse the results, applying the evaluation metrics defined.

There are not many models trained for zero-shot classification, which is why only BART was used. We experimented with a model fine-tuned on Spanish data, SELECTRA[24], but the results were below random guessing, so they will not be presented.

[23] `https://huggingface.co/facebook/bart-large-mnli`
[24] `https://huggingface.co/Recognai/zeroshot_selectra_medium`

6.4 Classifiers with Context

Since the focus of this work is to perform SA in a dialogue context, it was important to determine if the sequential nature of the dialogue was relevant to determining its sentiment.

This section will present the CRF model and the BERT-CRF model, an approach that makes use of CRF's sequence modeling characteristics while combining them with the more powerful deep learning model, BERT.

It should be noted that the Few-SL, Zero-SL, and BOTSchool API models contain a limit on the number of tokens received, so when considering the previous sentences they may exceed this number. Hence, some samples may not contain the two previous sentences, but only one, or just the current sentence. In extreme cases, an option to cut the current sentence was added so that it would be usable by these models.

In the Few-SL and Zero-SL approaches with context, an extra token is added, |pad|, that separates between each turn in the dialogue. An example of the use of this extra token follows, from the AL dataset, where three input sequences are concatenated:

1. |bos| churrasqueira fatura |pad| Se bem percebi pretende falar sobre faturação e pagamentos. E é sobre este numero de onde me está a ligar? Caso não queira informação de faturação diga "voltar atrás" |pad| dás isto é demais |eos| |sentiment| 0

Figure 6.1 and Table 6.3 present the set of dialogues used as examples in the Few-SL approach, for both datasets. When not considering context, a sentence was chosen from these dialogues (this sentence is signaled with bold).

Dialog ID	Utterance	Sentiment
1	**xUSERx Posso terminar o contrato e mudar para outra operadora sem custos? Ja que voces nao tem capacidade de fornecer um servico basico.**	0
1	Olá xUSERx lamentamos a situação que se sinta assim, mas existe uma anomalia a decorrer em algumas zonas do país. No entanto, a nossa equipa técnica já se encontra a retificar esta situação. Estamos ao seu dispor para ajudar. Obrigado e até breve!	1
2	**xUSERx por favor, não deixes de existir! (xURLx xURLx**	1
2	xUSERx xUSERx xUSERx Seus lindos! 💙	1
3	**O serviço da xUSERx há uma semana que está uma 💩! Se antes eu andava com os dados móveis ligados em casa por falta de wi-fi, agora é a box 🙎‍♀️ agradeço que isto seja resolvido de uma vez por todas senão só perdem clientes (e eu que sempre recomendei a operadora). 🙎‍♀️**	0
3	xUSERx Olá. Pedimos que reinicie os equipamentos. Caso as dificuldades persistam, será necessário o contacto com a equipa técnica para testes em linha, ligando 16913 ou 911 691 300. Se tiver um nº móvel Vodafone, pode ligar gratuitamente pelo My Vodafone (xURLx	1
3	xUSERx Já fizemos tudo o que a equipa técnica nos disse e inclusive voltei a reiniciar e continua sem funcionar portanto vou ter que contactar novamente a equipa técnica! Obrigada!	0

Figure 6.1: Set of dialogues used as examples in the Few-Shot Learning approach, using the Twitter dataset.

The set of dialogues in Figure 6.1 was chosen because the first contains a negative example, the second contains a positive example, and the third contains a more strongly negative example.

The set of dialogues in Table 6.3 was chosen because the first contains a short and mostly neutral (non-negative) common example, the second contains a longer, strongly

Table 6.3: Set of dialogues used as examples in the Few-Shot Learning approach, using the Altice Labs dataset.

Dialogue ID	Utterance	Sentiment
1	**queria falar com o técnico**	1
1	Vamos transferir a sua chamada para um assistente. Para um melhor atendimento diga qual o assunto que pretende tratar com o assistente PPP	1
1	outra questão	1
1	Lamento não poder ajudar, vou passar a sua chamada para o assistente PPP	0
2	tarifários	1
2	Se bem percebi quer tratar de tarifários. O que pretende saber ? informações sobre o seu tarifários, conhecer outros e alterar o seu tarifário? Se não pretende falar sobre tarifários, diga "voltar atrás"	1
2	churrasqueira fatura	1
2	Se bem percebi pretende falar sobre faturação e pagamentos. E é sobre este numero de onde me está a ligar? Caso não queira informação de faturação diga "voltar atrás"	1
2	dás isto é demais	0
2	Se bem percebi pretende falar sobre faturação e pagamentos. E é sobre este numero de onde me está a ligar? Caso não queira informação de faturação diga "voltar atrás"	1
2	**facturação não quero voltar a carregar as coisa nenhuma porra que eu cada vez estou mais ligada com estes clientes da meo com estes operadores da meo f**a-se**	0
2	Lamento mas não consigo ajudar. Vou transferir a chamada para o atendimento automático	0
3	**a internet está sempre a falhar**	0
3	Vou transferir a chamada para um serviço ATF que irá processar o seu pedido.	1
3	a internet está sempre a falhar	0
3	Vamos transferir a sua chamada para o serviço AIT	1

negative example, and the third contains a short negative example with repetition.

6.4.1 CRF

A CRF is a classifier that considers context, allowing for the classification of sequences instead of single samples, and is commonly appended to deep learning classifiers such as Long Short-Term Memory (LSTM)s or BERT models as the last layer of the network (Jin et al., 2019, Souza et al., 2019).

The application of this model required some extra work, in comparison with the other classifiers, since CRF is usually used for sequence tagging operations (Lafferty et al., 2001, Patil et al., 2020, Warjri et al., 2021) which use words instead of full sentences, and needed to be adapted to consider sentences instead. We also needed to consider which features would be relevant. Some possible features were:

- **Turn/Longest Dialog**: Current turn number in comparison with the longest dialogue in the training data;
- **Number of Words**: Number of tokens in the sentence. This is obtained using Spacy[25];
- **Has Question**: Whether the sentence contains a question mark;
- **Has Exclamation**: Whether the sentence contains an exclamation mark;
- **Beginning of Speech (BOS)**: Whether the sentence is the first in the dialogue;

[25] `https://spacy.io/models/pt/`

- **Previous Labels**: The sentiment of the previous sentences, if there are any. Up to two previous sentences are considered. In training, the annotated sentiment is used, while in testing, the predicted sentiment is used;
- **Encodings:** The embeddings of the previous sentences, if there are any, and the current sentence.

´ The HasQuestion and HasExclamation features were thought to possibly have low informational value, and so the CRF model was tested with and without these features.

For this approach, the following steps were followed:

1. Load and split the data using a 60%-20%-20% ratio or cross-validation;
2. Define the model and tokenizer or TF-IDF vectorizer, according to the type of encoding used;
3. Encode the dialogues and define the features set for each sample;
4. Train the model;
5. Evaluate the model in the validation and testing datasets while updating the previous labels feature (when applicable);
6. Analyse the results, applying the evaluation metrics defined.

6.4.2 BERT-CRF

Souza et al. (2019) developed a BERT-CRF architecture used for Named-Entity Recognition (NER) in the Portuguese language and explored feature-based and fine-tuning approaches. In this work, only the fine-tuning approach was used, since it generally obtains better results.

We developed an architecture that consists of using the BERT model (i.e., BERTimbau) and feeding its output scores and embeddings to the CRF model, as well as other features. The loss generated by the CRF model will be used to update the BERT model's weights. Figure 6.2 illustrates the architecture used, as well as the features fed to the models.

The loss value was obtained using the Negative Log Likelihood Loss (NLLL), the loss function utilized by BERT models and that can be computed to work with the CRF model, so that the loss returned by the latter can maintain its meaning by the former, connecting both models. The NLLL function is based on the softmax output (a function that provides the probabilities that a certain input belongs to a certain class). The higher the probability of the predicted class, given by the softmax output, the lower the associated loss.

As seen in Figure 6.2, the BERT model feeds the CRF model with a representation for each sentence considered. Figure 6.3 represents BERT's hidden states, the embeddings resulting from the data going through BERT's 12 hidden layers. There are 13 hidden states, because the first state represents the initial input embedding, before going through any hidden layer. The embedding of the previous layer is fed to the next until it reaches the last layer, the one with the most contextual information. We only use the [CLS] token since it represents the whole sentence and not just word tokens, making the final embedding present the dimensions [number of samples x 768] instead of [number of samples x max length x 768], as seen in the figure.

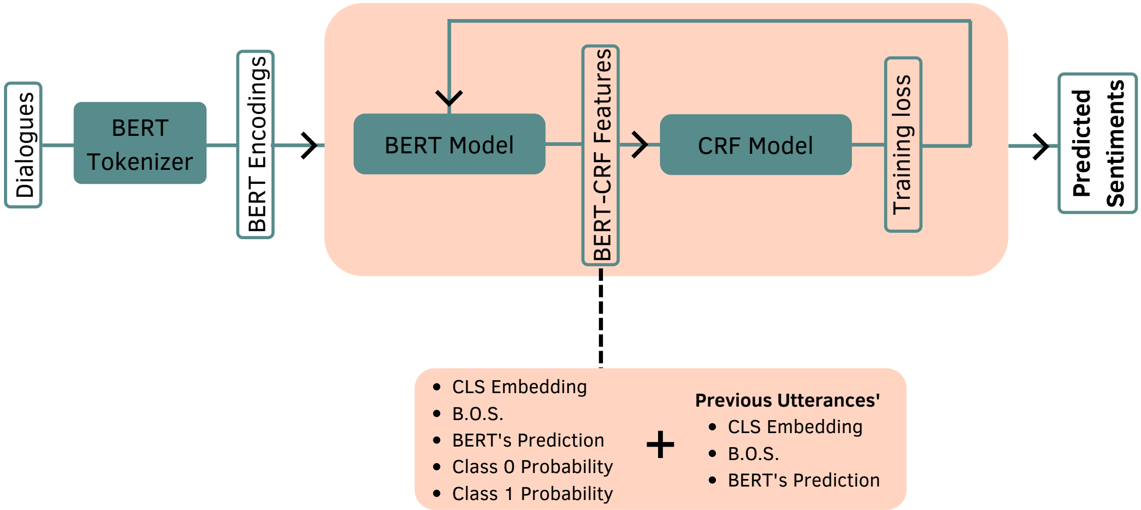


Figure 6.2: BERT-CRF model's architecture

Despite containing the most contextual information, the last layer may be too fitted for BERT's target functions, Masked Language Model (MLM) and Next Sequence Prediction (NSP), which is why we experiment with the following ways to obtain the final representation:

1. Concatenation: use the last 4 hidden layers, ending up with dimension [number of samples x (768 x 4)] instead of the regular [number of samples] x 768;
2. Mean: use the average of the last 4 hidden layers, keeping the 768 dimension;
3. Last Hidden Layer: use the last hidden layer;
4. First Hidden Layer: use the initial input embedding, containing no contextual information;

Other layer options and operations could have been used (e.g., the sum of layers, any other single layer besides first and last).

For this approach, the following steps were followed:

1. Load and split the data using a 60%-20%-20% ratio or cross-validation;
2. Define the BOS feature for each sample, the model, tokenizer, and optimizer (AdamW);
3. Use the BERTimbau Tokenizer to generate embeddings for each dataset, with padding to the maximum length found in the training dataset to ensure equal length in all samples;
4. Set the features for each sample (embeddings and BOS) and train the BERT model, using the chosen representation strategy, and batches of size 16, for 2 epochs.
5. After training the BERT model, obtain the selected CRF features, using the labels and [CLS] embeddings generated by the BERT model;
6. Train the CRF model, compute the NLLL, and feed it to the optimizer to update the BERT model's weight at the end of each epoch, and evaluate the model with the validation dataset;

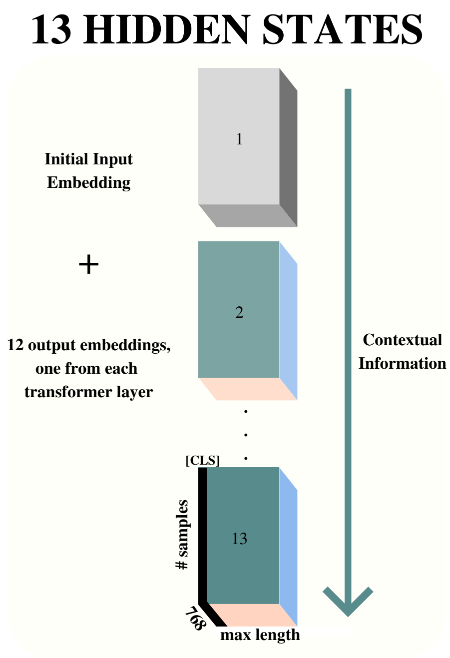


Figure 6.3: BERT's Hidden States, adapted from Neeraj (2020)

7. Evaluate the model on the testing dataset and analyse the results, applying the evaluation metrics defined.

6.5 Non-Dialogue Training

Some models were used to verify if they were able to properly classify dialogues when trained using standard, non-dialogue data (the SentiTuites dataset, mentioned in chapter 5). Only models without context were considered, since the non-dialogue data used is composed of single samples, lacking any kind of context.

The chosen models were the shallow learning classifiers. Their results, which will be presented in chapter 7, were not promising. Due to the low performance and the longer training and evaluation processes, the dense representations and the deep learning models were not used in this experiment.

It should be noted that the domain reflected in SentiTuites (Politics) is not one of our target domains, which is likely to have harmed the performance of these models.

6.6 Validation

We initially split the data into three sets, training, validation, and testing, with a 60%-20%-20% split, which grants a common ground between all models. Later, a 5-Fold Cross Validation technique was applied, which is expected to improve the performance of each model since it will select the best split for the training-validation data. The training and validation sets were used to tune the models, and the testing set was used for a final and independent evaluation of each model. When performing the split automatically, the stratification option was enabled, in order to assure that both classes were represented in each dataset. This is especially important since our datasets are unbalanced.

Regarding the cross-validation, for the free-of-context models, which use only current utterances, we used a Stratified 5-Fold cross-validator[26], which divides the data while attempting to keep each split somewhat balanced. For the models using context, a Non-stratified 5-Fold cross-validator[27] was used, since the input of these models are dialogues, and these are not labeled.

Cross-Validation can be particularly interesting when used with small datasets since it is likely that the variance in each fold is lower, while in larger datasets it is likely that each fold contains a good representation of the dataset. Hence, in smaller datasets, this approach may have a bigger impact, since it will choose the fold with the most relevant information (defined through an analysis of the best performance from each trained model).

Adding to the cross-validation, we also tuned most models' hyperparameters. This process will be explained in the next section.

6.7 Hyperparameters Tuning

A hyperparameter is a parameter used to configure the model, and that is not derived via training. There are many options for each parameter that can affect the performance of each model, therefore, we combined the cross-validation technique with hyperparameters' tuning to get better performances with the models. However, approaches such as the lexicon-based models, the AL's BOTSchool[28] API, and Zero-SL and Few-SL models were not subject to tuning, due to their lack of a training stage and the nature of each application.

The traditional models, fine-tuned BERT, and CRF, were all tuned. In the case of the BERT-CRF model, the best hyperparameters from each composing model will be used since the tuning of this model would take a very long time, and it is expected that the use of the tuned individual models will improve the performance of the whole model.

Some models required higher computational power and could not run the tuning process for the AL dataset. Hence, for these models (SVMs, Fine-tuned BERT, and BERT-CRF), we used the tuned hyperparameters found for the Twitter dataset, which may not be ideal.

The technique used for the hyperparameters tuning was Grid Search[29], which performs an exhaustive search over each possible parameter value. The model with the best parameters will be selected and applied to the testing set for final evaluation.

Due to a lack of time and resources, the Genetic Algorithm (GA) search and Population-Based Training (PBT) options for hyperparameters' tuning were not explored.

6.7.1 Shallow Classifiers

In this sub-section we will list the hyperparameters and values considered for each of the Shallow Learning Classifiers.

Regarding the Logistic Regression, we considered four hyperparameters:

[26] `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html`

[27] `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html`

[28] `https://botschool.ai/`

[29] `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html`

1. Solver: Limited-Memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS), the default algorithm, and Library for Large Linear Classification (LibLinear), good for small datasets;
2. Penalty: Regularization penalty, using L2 regularization;
3. C: Regularization parameter where smaller values specify stronger regularization;
4. Max. Iterations: Number of iterations for the solver to converge.

For the C values, we generated 20 values between the log scale of 0 and 10, and for the maximum iterations, we experimented with 100, 200, and 500. The penalty hyperparameter was set at L2, and the solvers were already described.

Regarding the SVM, we considered four hyperparameters:

1. Kernel: Defines the kernel function for the algorithm;
2. C: Regularization parameter that defines the cost of misclassification. Higher values give lower bias and higher variance;
3. Gamma: Kernel coefficient, specific to the *poly* and *rbf* kernels, that controls the shape of the margin between classes. Higher values give higher bias and lower variance;
4. Degree: Degree of the *poly* kernel function.

For the kernel, we selected linear, poly, and rbf functions. For the C values, we experimented with 0.1, 1, 10, and 100. For the Gamma values, we chose 0.001, 0.01, 0.1, and 1. For the Degree values, we set 2 and 3.

Regarding the Random Forest, we considered seven hyperparameters:

1. Criterion: The measure of the quality of a split, gini or entropy, which calculate the impurity of a node (a pure node only contains one class);
2. Estimators: The number of trees in the model;
3. Max. Features: Define the limit of features when splitting, considering the square root or the logarithm to the base 2 of the number of features;
4. Max. Depth: How deep a tree can become;
5. Min. Samples Split: The lower limit on the number of samples required to split an internal node;
6. Min. Samples Leaf: The lower limit on the number of samples required to be at a leaf node;
7. Bootstrap: Whether to use bootstrap samples when building trees. If false, the whole dataset is used to build each tree.

For the Estimators, we selected 100 and 200 trees. We defined Max. Features and Criterion as described in the list. For Max. Depth, we chose 10 and 50. For the Min.

Samples Split we experimented with 2 and 5, and for the Min. Samples Leaf with 1 and 2. We explored using and not using bootstrap in this classifier.

Regarding the Naive Bayes, there was only one hyperparameter to consider, the Alpha parameter, which was explored with values of 0 and 1. The alpha parameter represents the smoothing prior, which accounts for features not present in the training set and prevents probabilities of zero. An alpha value of 1 is called Laplace smoothing (adds 1 to all counts), otherwise it is called Lidstone smoothing (adds a value different than 1).

6.7.2 CRF

Regarding the CRF, we considered three hyperparameters:

1. Max. Iterations: Limit the iterations for the algorithm;
2. C1: Coefficient for L1 regularization (Lasso), a method to manage overfitting and perform feature selection;
3. C2: Coefficient for L2 regularization (Ridge) a method to manage overfitting and perform feature selection.

For the C1 values, we set eleven values between 0 and 2. For the C2 values, we selected ten values between 0.001 and 1. As for the Max. Iterations, we experimented with 100, 200, and 500.

We set the training algorithm to L-BFGS, which is a popular optimization algorithm that was also used in the Logistic Regression model. We also allowed all possible transitions, meaning that the model is capable of generating transition features that associate all possible label pairs, even if they do not occur in the training data.

6.7.3 Fine-tuned BERT

Regarding the fine-tuning of BERTimbau, we considered four hyperparameters:

1. Epochs: Number of times the whole dataset is passed through the model;
2. Batch Size: the size of the data sub-samples, makes the learning process faster because it does not give the model the whole data at once;
3. Learning Rate: Controls the step size (higher means faster learning but a higher chance of not reaching the optimum value);
4. Epsilon: Value to avoid division by zero when the gradient is close to zero. Larger values make the training process longer because the weight updates will be smaller.

The last two hyperparameters apply to the optimizer.

For the number of Epochs, we chose 2 and 4. For the Batch Size we had 16 and 32 but could not run experiments in the available machines using the latter. For the Learning Rate we experimented with 0.0001 and 0.00005. For the Epsilon values, we selected 0.001 and a very small value, 0.00000001.

6.7.4 BERT-CRF

As mentioned, BERT-CRF will use the best hyperparameters found for each individual model.

For the Twitter dataset, this translates into the following values for the seven total hyperparameters:

1. Batch size (BERT): 16;
2. Epochs (BERT): 2;
3. Learning Rate (BERT): 0.0001;
4. Epsilon (BERT): 0.00000001;
5. C1 (CRF): 0.408;
6. C2 (CRF): 0.556;
7. Max. Iterations (CRF): 200.

For the AL dataset, only the three CRF hyperparameters' values changed: the C1 value was set to 1.8, the C2 value to 0.67, and the number of Max. Iterations was defined as 100.

6.8 Summary

This chapter tackles the feature engineering (which, for most classifiers, consists of applying an encoding technique, TF-IDF or BERTimbau NLI, to represent each utterance), model training and optimization (when applicable, since some models do not require training data) problems.

Regarding the first problem, we used three main approaches, as represented in Figure 5.1: sparse, when using the TF-IDF technique, dense, when using the BERTimbau NLI approach, and manual, which we define as a manually selected set of features (e.g., number of words, BOS), including a sparse or dense representation of the utterances involved. This last option is only used for the CRF and BERT-CRF classifiers. When the context is involved but not included in the original models, we concatenate the current and previous utterances to compare the performances in both scenarios (with and without context).

Since representation has an impact on the performance of a model, we experimented with the use of unigrams vs. multi grams in the models using TF-IDF, the definition of a smaller set of features for the CRF model, the use of different representation functions in the BERT-CRF model, and setting the Zero-SL hypotheses as labels or sentences.

Regarding model training and optimization, we developed solutions based on the ten classifiers listed in Figure 5.1. The datasets were split 60%-20%-20% for training, validation, and testing, when the models required training data, and 80%-20% when they did not, as is the case of the BOTSchool API, Few-SL, and Zero-SL classifiers. The first group of classifiers was optimized, using cross-validation and grid-search. We kept the results from the non-optimized models as they all share the same training and testing data, providing us with a solid base for comparison of their performances.

An experiment regarding the performance of models trained in standard SA data and their application to dialogues was also defined, and it uses the Sentituites dataset as a source for non-dialogue samples.

The results obtained from each experiment and classifier will be presented and discussed in the next chapter.

Chapter 7

Experimental Results

In this chapter, we will focus on the Model Evaluation step presented in Figure 5.1. We will present the results from the evaluation of each of the developed classifiers, as well as the experiences mentioned in Chapter 6, including the comparison of different n-grams, kernels, representations, and hyperparameters.

It should be mentioned that the number of samples in each dataset is different based on whether we consider context or not. This happens because without context we perform the data split by utterance, because we use a method[1] that splits the data automatically, while with context we perform it by dialogue and each contains a varying number of utterances. However, we could also have organized the data in dialogues, split it like that, and then extract the individual sentences. Since we were not considering the context in those approaches, we did not think this was important enough to experiment with, since the only difference would be that each sub-dataset would contain full dialogues. The data split is used when defining a testing dataset, and when performing cross-validation or defining a fixed validation set (however, we did not explore the validation scores). Table 7.1 presents the number of samples of each label, in the testing dataset, in both scenarios, with and without context. It is also relevant to mention that the testing dataset contains the same data (dialogues or utterances) for all models since the data split is performed equally across all of these experiments, which creates a common base between each of them. When not performing cross-validation, the training data is also the same for every model (that requires training).

Table 7.1: Number of samples of each class present in the testing dataset, with and without using context.

	Altice Labs Testing Dataset (Without Context)	**Twitter Testing Dataset (Without Context)**	**Altice Labs Testing Dataset (With Context)**	**Twitter Testing Dataset (With Context)**
Non-Negative Samples	867	120	891	139
Negative Samples	196	91	191	79
Total Samples	**1,063**	**211**	**1,082**	**217**

The evaluation results present six metrics, Accuracy, Precision, Recall, Specificity, F1-Score, and Area Under the ROC Curve (AUC). However, not all of them share the same relevance, and it is important to define which metrics are more important for this dissertation's problem.

With this in mind, and considering we are approaching it from a binary perspective

[1] `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html`

(i.e., a client's sentiment is either negative or non-negative), one could affirm that the loss of a client would be worse than the pointless allocation of a human to a client (i.e., when the client does not require human intervention), since losing a client would reduce the company's profit and negatively affect their reputation, as the client could complain about the service, officially or not. What this analysis tells us is that a false negative situation (i.e., classifying a client's sentiment as non-negative when it is actually negative) should be minimized as much as possible, even if it might slightly increase the number of False Positive (FP) (i.e., this would translate into calling a human to handle a client with no negative sentiment). With this in mind, it could be argued that the most relevant metric would be the recall. However, the f1-score can be equally relevant, since it balances both situations mentioned (the loss of a client due to a False Negative (FN) scenario and the pointless allocation of a human to a client due to a FP scenario), by considering both the recall and the precision metrics.

Despite considering recall and f1-score as the most relevant metrics, we will also keep an eye on the AUC, as it is a common performance metric since it evaluates the classifier's ability to distinguish between classes.

In the last section of this chapter, we will present a summary of the analysis of the experimental results.

7.1 Lexicon-based Approaches

In this section, we present the results using the lexicon-based models. These classifiers do not produce probabilities of a sample belonging to each class, so the metric AUC is not applicable. The remaining metrics' scores are presented in Tables 7.2 and 7.3.

Table 7.2: Evaluation of the lexicon-based approaches using the Altice Labs dataset.

Altice Labs Dataset					
Lexicon	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**
NRC Emotion Lexicon	0.71	0.17	0.15	0.84	0.16
SentiLex-flex	0.23	0.19	0.95	0.06	0.32

Table 7.3: Evaluation of the lexicon-based approaches using the Twitter dataset.

Twitter Dataset					
Lexicon	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**
NRC Emotion Lexicon	0.59	0.64	0.1	0.96	0.17
SentiLex-flex	0.63	0.54	0.92	0.4	0.68

Using the Altice Labs (AL) data (the focus of this work), these models perform worse than random-chance for most metrics, making it a bad option. These values also confirm why it is important to consider the F1-Score metric, and not just the Recall metric since, in the SentiLex results, we get very high recall scores, which could make us wrongly assume this was a good model.

Using the Twitter data, the results are not as low, but we still cannot claim that these are good models.

By analysing the Recall and Specificity scores, we can conclude that using the SentiLex lexicon, the models seem to classify most utterances as Non-Negative (high recall and low specificity), while when using the NRC Emotion lexicon, the models seem to classify most utterances as Negative (low recall and high specificity), especially in the AL dataset.

7.2 Shallow Learning Classifiers

This section presents the results for the shallow learning classifiers using sparse and dense representations. We also analyse the effect of the n-grams choice in the sparse representation models. Regarding the Support Vector Machine (SVM) model, we chose the RBF kernel since it performed better than the other kernels (the evaluation results of each kernel, with and without context, can be seen in appendix 8.2).

7.2.1 Sparse Representations

A sparse representation is obtained by using the Term Frequency - Inverse Document Frequency (TF-IDF) technique. The results presented in Tables 7.4 and 7.5 use unigrams, but we will explore the effect of the choice of n-grams further ahead.

Table 7.4: Evaluation of the shallow classifiers, using TF-IDF encoding, with the AL dataset.

Altice Labs Dataset						
Model	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Logistic Regression	0.91	0.84	0.63	0.97	0.72	0.92
RBF SVM	0.91	0.84	0.66	0.97	0.74	0.9
Naive Bayes	0.91	0.82	0.64	0.97	0.72	0.92
Random Forest	0.91	0.78	0.7	0.96	0.74	0.92

Table 7.5: Evaluation of the shallow classifiers, using TF-IDF encoding, with the Twitter dataset.

Twitter Dataset						
Model	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Logistic Regression	0.84	0.89	0.73	0.93	0.8	0.94
RBF SVM	0.86	0.85	0.81	0.89	0.83	0.93
Naive Bayes	0.82	0.88	0.67	0.93	0.76	0.94
Random Forest	0.83	0.86	0.73	0.91	0.79	0.9

The Twitter dataset presents better results than the AL dataset, overall, although the greatest difference is in the Recall scores. This means that these shallow models with sparse representations are better at classifying negative sentiment in the Twitter dataset.

Despite the scores being generally good, the recall (and so the f1-score as well) are mostly below 80%, so while the results are much better than the lexicon-based results, there is room for improvement.

The best shallow model with sparse representations seems to be the SVM for the Twitter dataset with a slightly lower AUC but higher recall and f1-score than other models, and the Random Forest for the AL dataset, achieving the top score on all three important metrics.

Regarding the experiment on the use of unigrams, bigrams, and trigrams in the shallow classifiers with sparse representations, the first presented the best results, and this is the n-gram setting to be used in all following experiments. The full results can be viewed in Appendix 8.2.

Focusing on another experiment, related to the effect of using models trained in standard Sentiment Analysis (SA) data instead of dialogues, a comparison of the F1 Scores obtained when training shallow learning classifiers (with sparse representations) in the Sentituites dataset (80% split for training) and evaluating the performance on the remaining 20% of this dataset, and the Twitter and AL datasets can be seen in Figure 7.1. The results regarding the other evaluation metrics are presented in Appendix 8.2.

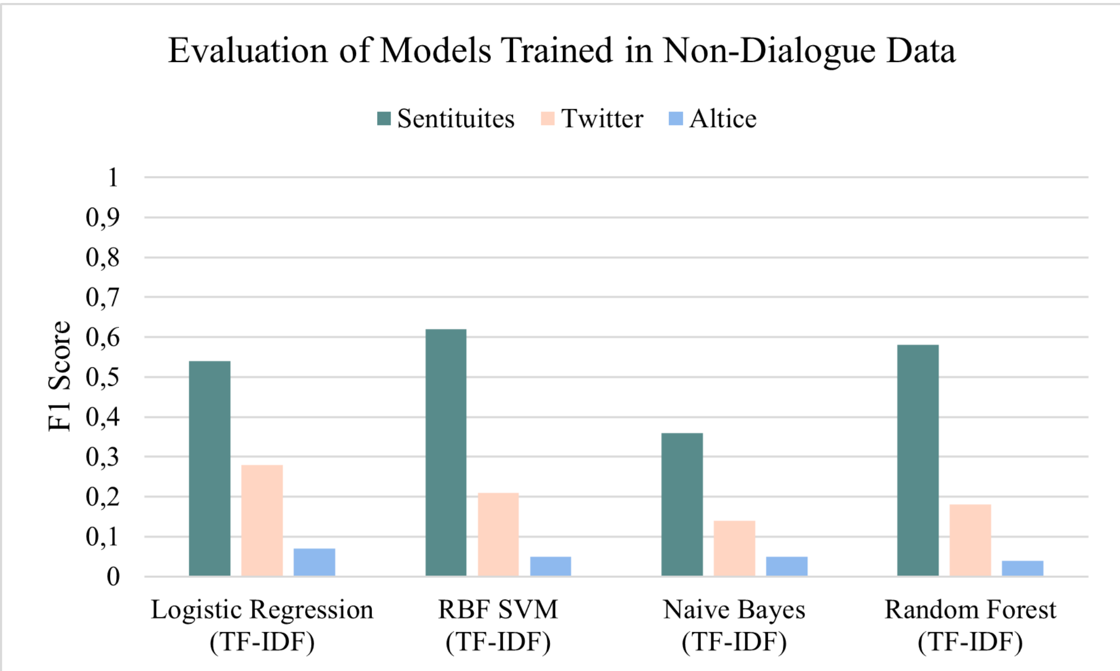


Figure 7.1: Evaluation of the Models Trained in Non-Dialogue Data

The F1 Scores of the models trained in the Sentituites dataset reveal that the performance is not good even when evaluated in the same type of data (under random guessing using the Naive Bayes classifier). This could be due to overfitting in the training process, since the metrics in Appendix 8.2 show low Recall scores and very high Specificity scores, suggesting that the classifiers mostly make Non-Negative predictions, resulting in a high number of True Negative (TN) and FN. Since this experiment was not a priority, despite being interesting, we did not attempt to further analyse and overcome the overfitting hypothesis.

The decrease in performance for all models is striking, especially in the AL dataset, which differs the most from the Sentituites type of data. The difference in the domains used for training and evaluation is likely to have harmed the performance of these classifiers, but this may not be the only reason for such a significant decrease in the evaluation scores. These results suggest that it may be important to train the classifiers in dialogues, as standard SA data may not be sufficient for a good performance in SA in the context of dialogues.

Every model evaluated in our datasets performs below random guessing for both the Recall and F1 scores, and hence this approach should not be considered for the purpose of this dissertation.

7.2.2 Dense Representations

A dense representation is obtained by using the BERTimbau Natural Language Inference (NLI) Tokenizer. The results are presented in Tables 7.6 and 7.7.

Table 7.6: Evaluation of the shallow classifiers, using BERTimbau NLI encoding, with the AL dataset

Altice Labs Dataset						
Model	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Logistic Regression	0.93	0.8	0.83	0.95	0.81	0.94
RBF SVM	0.93	0.82	0.8	0.96	0.81	0.94
Naive Bayes	0,9	0.76	0.69	0.95	0.72	0.9
Random Forest	0.92	0.82	0.76	0,96	0.79	0.95

Table 7.7: Evaluation of the shallow classifiers, using BERTimbau NLI encoding, with the Twitter dataset

Twitter Dataset						
Model	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Logistic Regression	0.92	0.91	0.9	0.93	0.9	0.96
RBF SVM	0.91	0.93	0.86	0.95	0.89	0.97
Naive Bayes	0.86	0.84	0.84	0.88	0.84	0.92
Random Forest	0.88	0.88	0.84	0.92	0.86	0.96

Better performances are achieved in the Twitter dataset than in the AL dataset, overall, although the greatest difference is in the Recall scores. This means that these shallow models with dense representations are better at classifying negative sentiment in the Twitter dataset, as were the models with sparse representation. As seen in the data projections earlier, in Figures 5.7 and 5.6, this may also be due to the Twitter data being more easily separable (although not easy), as we could see in the projection that the AL data seemed to overlap more, although it also contained more samples. Hence, there is a possibility that the Twitter dataset is just easier to classify.

The best shallow model with dense representations seems to be the Logistic Regression for both datasets, with a slightly lower AUC but higher recall and f1-score than the other models.

From the analysis of Tables 7.4, 7.5, 7.6, and 7.7, we can affirm that the use of dense representations, in the form of BERTimbau NLI embeddings, provides the models with better features, which is reflected in the higher performance for all shallow models. Naive Bayes was the only model not to show a notable improvement using these embeddings, and just on the AL data, but still, the recall value slightly increased. The higher Specificity score in both approaches is justified by the larger presence of Non-Negative samples in the datasets.

7.3 Deep Learning Classifiers

This section presents the results for the deep learning classifiers, in Tables 7.8 and 7.9.

Regarding the Zero-Shot Learning (Zero-SL) model, we will present the results obtained using the labels in Portuguese (Negativo, Neutro, Positivo) as hypothesis, since this approach obtained higher results overall (slightly lower on the Twitter dataset, but notably higher on the AL dataset, compared to the other Zero-SL approaches). The scores of these other approaches can be viewed in Appendix 8.2.

Regarding the Few-Shot Learning (Few-SL) model, the AUC score will not be presented (defined as Not Applicable (N.A.)), since this approach generates sequences, not providing us with the required probabilities for each label. We will present the results obtained without using descriptions (Definition, Constraint, Prompt) and using the Generative Pre-trained Transformer (GPT)-Neo model, and will also analyse the number of valid classifications.

Table 7.8: Evaluation of the deep learning classifiers without context, with the AL dataset

Altice Labs Dataset						
Model	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
BOTSchool API	0.85	0.57	0.73	0.88	0.64	0.87
Fine-tuned BERTimbau	0.95	0.89	0.85	0.97	0.87	0.97
Few-Shot Learning	0	0	0	0	0	N.A.
Zero-Shot Learning	0.8	0.46	0.76	0.81	0.57	0.85

It is notable that the Few-SL and Zero-SL are not good options for this work. The latter's metric scores could be compared to the lexicon-based scores, seen in Tables 7.2 and 7.3, and the former's scores can not be considered, because for the Twitter dataset only 0.01% of the classifications were valid, while for the AL dataset no classification was valid. The decoding algorithm used was likely not the best option (although it is not the worst, since it combines the Top-P and Top-K algorithms), but it was chosen because it was rather quick to run, as seen previously in Table 6.2. Due to a lack of time and resources, we did not experiment with other decoding algorithms or models (e.g. Meta-OPT).

Going back to the Zero-SL model, it performs considerably better than the Few-SL approach but, for the Twitter dataset, it performs worse than the SentiLex lexicon model. However, it is interesting how a model not trained in the Portuguese language, managed

Table 7.9: Evaluation of the deep learning classifiers without context, with the Twitter dataset

Twitter Dataset						
Model	Accuracy	Precision	Recall	Specificity	F1 Score	A.U.C.
BOTSchool API	0.75	0.72	0.7	0.79	0.71	0.75
Fine-tuned BERTimbau	0.93	0.97	0.88	0.97	0.92	0.99
Few-Shot Learning	0.67	1	0.5	1	0.67	N.A.
Zero-Shot Learning	0.58	0.44	0.63	0.55	0.52	0.62

to obtain better results using hypotheses in Portuguese over English.

The best deep learning model seems to be the Fine-tuned BERTimbau for both datasets, although in the Twitter dataset it rivals the results obtained using the Logistic Regression with dense embeddings. The BOTSchool Application Programming Interface (API) performs worse than the shallow learning classifiers with dense embeddings, and in the AL dataset it is comparable to the performance of these classifiers with sparse embeddings. As such, there seem to be better options than the actual choice for SA in AL, and this is an important contribution of this work.

7.4 Classifiers with Context

This section presents the results for the classifiers using context, either by concatenating previous sentences in the input or by being models able to process context (Conditional Random Field (CRF) and Bidirectional Encoder Representations from Transformers (BERT)-CRF). Since all previous models are considered, along with two more others, there are too many results to present in a table, and so we graphically present the results for each model (Figures 7.2 and 7.3), considering the most important metrics. The table with the full results is in Appendix 8.2. A graphical comparison of the results with and without context can be seen in Figures 7.4 and 7.5. We chose the F1-score metric since it presents a balanced result between the recall and precision metrics. In this comparison we will not include the CRF and BERT-CRF models since they were not tested without context.

Regarding the Zero-SL and Few-SL models, the configuration is the same as for the results presented in the previous section (Portuguese labels as hypotheses and no use of description, respectively). However, the latter will not be included in the comparison of the results with and without context, due to the extremely low percentage of valid results for the Few-SL model without context.

Regarding the CRF model, the results presented use the features described in section 6.4, since there was no significant difference between using fewer or more features. For the TF-IDF embedding, the CRF model is using unigrams, which achieved the best results between unigrams, bigrams, and trigrams. The results of other n-grams and features are in Appendix 8.2.

Regarding the BERT-CRF model, we are presenting the results obtained using the last hidden layer as the representation function. The results using the remaining options can be seen in Appendix 8.2.

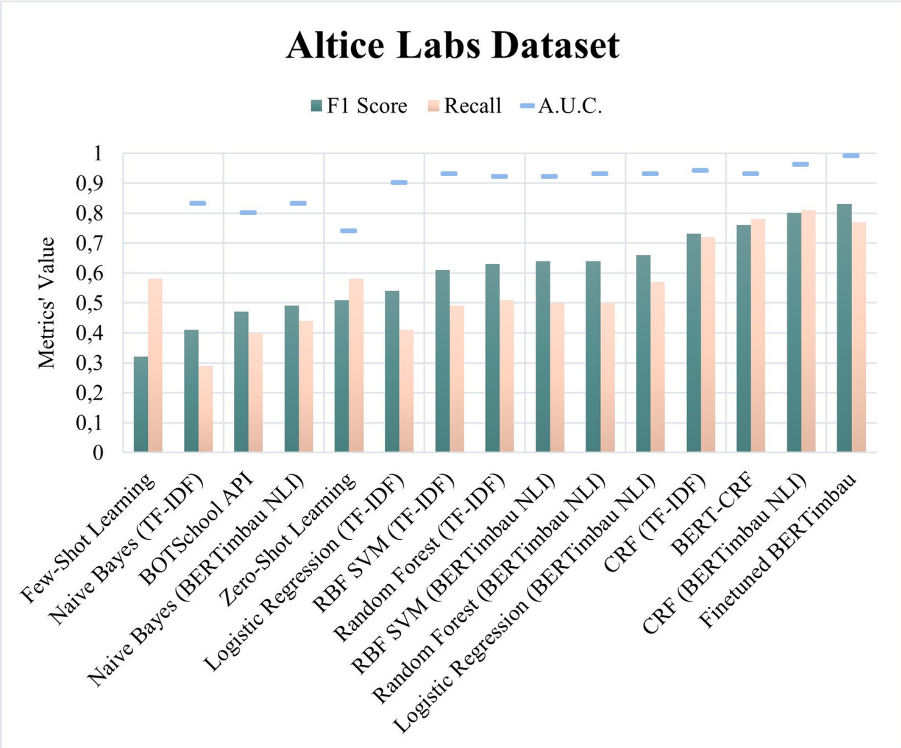


Figure 7.2: Evaluation of the classifiers with context, using the Altice Labs dataset

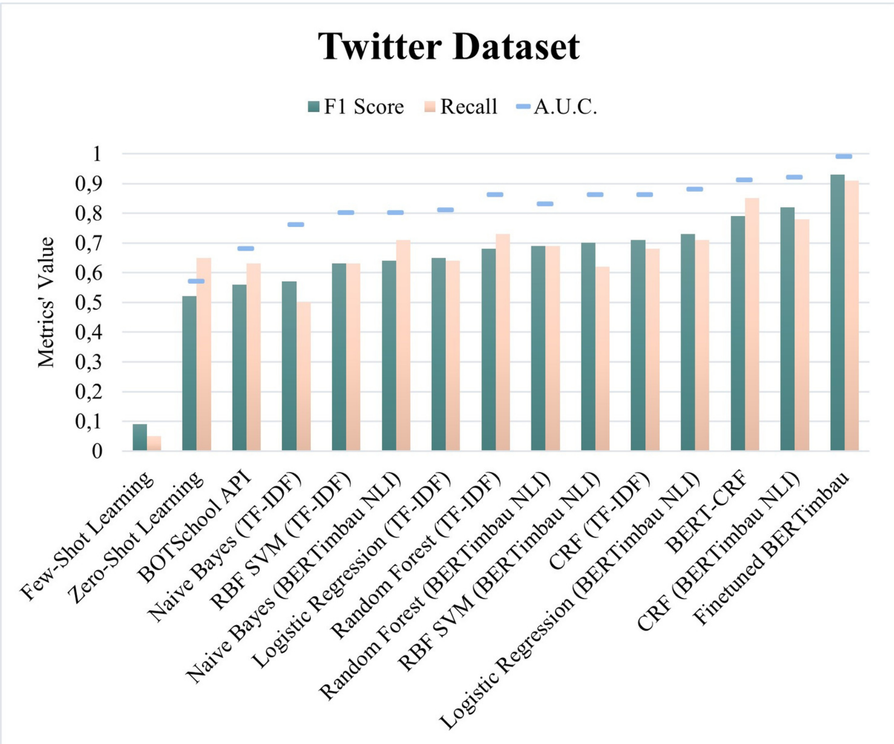


Figure 7.3: Evaluation of the classifiers with context, using the Twitter dataset

Notably, the highest scores come from the models that originally consider context, CRF and BERT-CRF, and the fine-tuning BERTimbau model for both datasets. The BOTSchool API, and Naive Bayes with sparse representations are the common models among the 5 lowest performing classifiers for both datasets. This is interesting because it suggests that the solution being explored by AL does not have to consider the client's previous utterances.

Regarding the use of sparse and dense representations in the CRF model, the use of the BERTimbau NLI embeddings seems to improve the performance, in accordance with the results from the shallow learning classifiers. Interestingly, with context, the Random Forest model with sparse representations outperforms most other classifiers, especially in the Twitter dataset where it takes the fourth top position. In the AL dataset, only the CRF models and the Logistic Regression with dense representations perform better than this model (considering only models that compare dense and sparse representations).

Similarly to the conclusions reached earlier, it is notable that the Few-SL and Zero-SL are not good options for this work, with scores around 60% or lower for most metrics (the exception being the AUC metric of the Zero-SL model with the AL data). Regarding the Few-SL, we can declare that the model performs better when considering context, but still, only 47% and 73% of the classifications were valid for the Twitter and AL datasets, respectively. This means that the AL dataset is better for text generation, probably because there is less variance in the samples. Yan et al. (2018) alerted to the fact that in short texts, such as tweets, abbreviations are commonly used, and the sentence is normally informal, with poor grammar and misspellings, which could make it harder for a Few-SL model to generalize and gather sufficient semantic information, and possibly justify the significantly lower results for the Twitter dataset. We should also keep in mind that the model used is multilingual (predominantly English), which is likely to negatively impact the results.

An interesting note on the shallow classifiers with sparse representations is that when considering context, the use of multi grams, particularly trigrams, improved the performance of these classifiers. These results can be seen in Appendix 8.2.

We will now compare the performance of the classifiers with and without considering the context.

The performance decrease when using context is notable for most classifiers, except for the fine-tuned BERTimbau model, which shows that context does not improve the performance of these classifiers, for the datasets used. The Naive Bayes and BOTSchool API are the models with the highest decrease in performance (as evaluated by the F1-score metric).

While in the AL dataset, there are no improvements at all by considering context (the concatenation of the current and past utterances), in comparison to the results obtained without considering it, in the Twitter dataset, despite most classifiers performing better without context, it is more often that the difference between both approaches returns similar results. In fact, we should note that, the Fine-tuned BERTimbau model remains the classifier with the highest performance, and when considering context, in the Twitter dataset, its score is further increased, making this, so far, the best model for SA in this dataset.

Regarding the other CRF model experiments, about the use of different n-grams and other features, the classifiers applying only unigrams versus bigrams and trigrams performed better on all important metrics on both datasets (the use of trigrams presented the worst results on both datasets). The experiment regarding the use of fewer features showed no significant differences in both datasets, as mentioned earlier. The full results for both experiments are presented in Appendix 8.2.

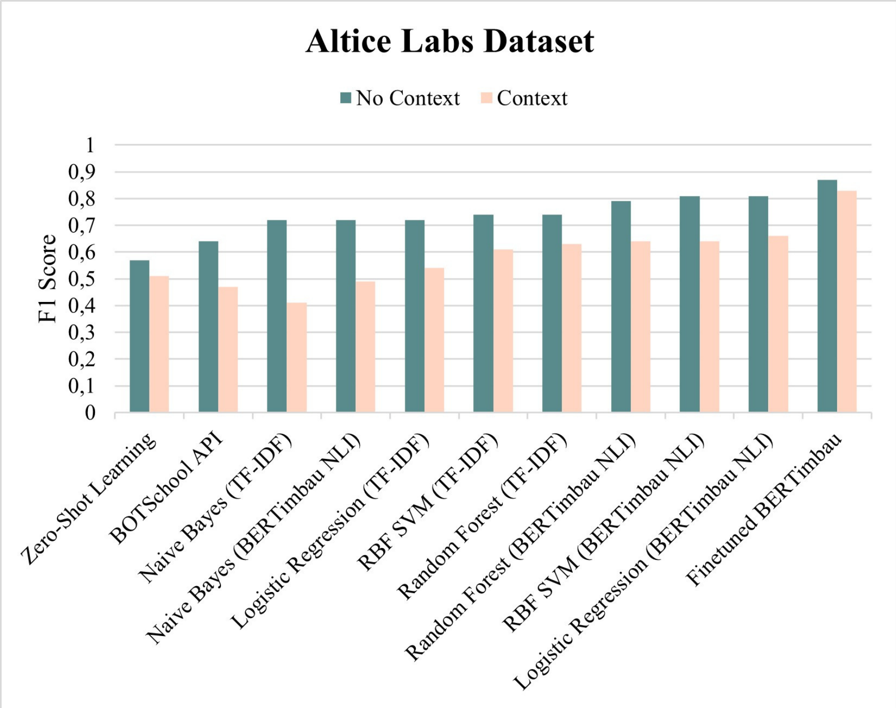


Figure 7.4: Comparison of the F1 Score between Classifiers with and without Context, using the Altice Labs dataset

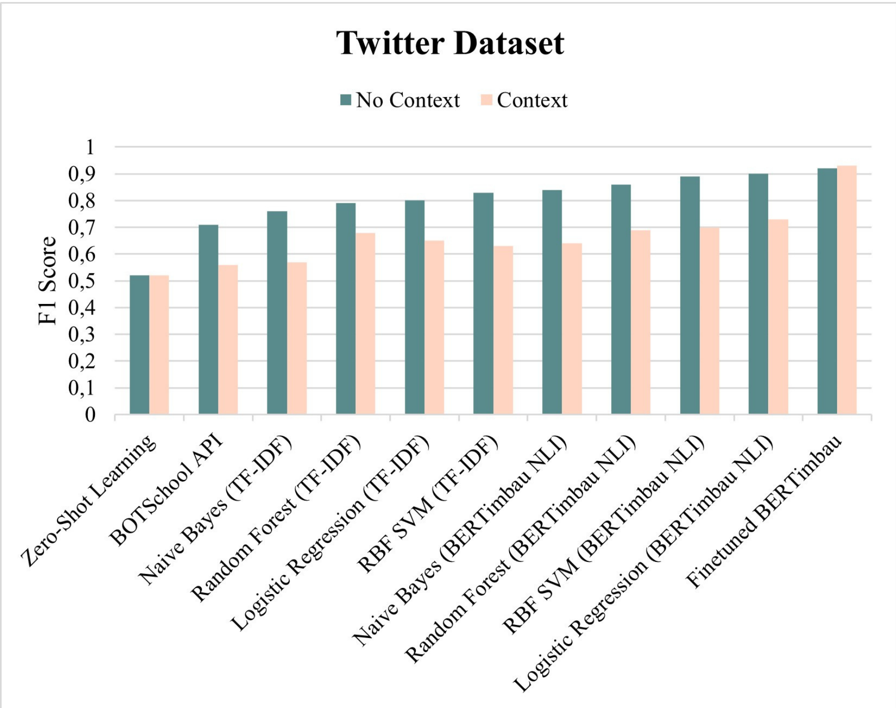


Figure 7.5: Comparison of the F1 Score between Classifiers with and without Context, using the Twitter dataset

7.5 Cross-Validation and Hyperparameters' Tuning

This section presents the results for classifiers that could be optimized: all except the "ready-to-use" models, BOTSchool API, Few-SL and Zero-SL, and the lexicon-based approaches. Regarding the SVM models, we will only present the RBF kernel, as it was the one selected previously.

The cross-validation and grid search processes are time-consuming and of heavy computation. Hence, not all classifiers, namely SVMs, and fine-tunedBERT could go through these in the available time. As an alternative to this, and since most classifiers were able to perform these operations in the smaller Twitter dataset, we decided to use the best parameters found for this dataset in the AL dataset and verify if the results improved in comparison with the not-optimized classifiers. We automatically select the best model, according to the F1-Score.

The batch size hyperparameter considered for tuning BERTimbau could not be experimented with, because the available machines did not handle a batch size higher than 16.

The BERT-CRF model has issues handling the AL dataset even without optimization. Because of this and the tuning process being especially heavy in this scenario, since we would need to tune two models at the same time, for this classifier we will use the best hyperparameters found for each individual model and we will verify if there are improvements in its performance.

Figures 7.6 and 7.7 present a comparison between the performances of each classifier and its tuned version with highest performance for each datasets. For ease of visualization, we will only consider the F1 score. The full results are presented in Appendix 8.2. Regarding the fine-tuned BERTimbau model, we only presented the top performance model in this Appendix, but the full tuning experience (using the Twitter dataset) can be viewed in Appendix 8.2.

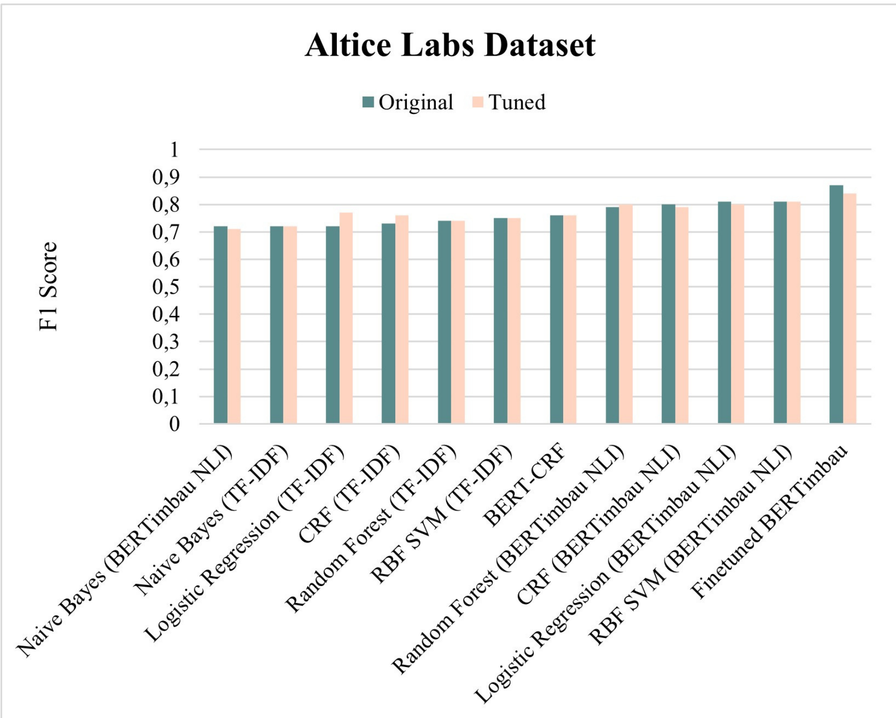


Figure 7.6: Comparison between the F1-Scores of the tuned and original Classifiers, using the Altice Labs dataset.

From both figures, we can verify that while the optimization process increases the scores of most models, there was no large variation between both performances.

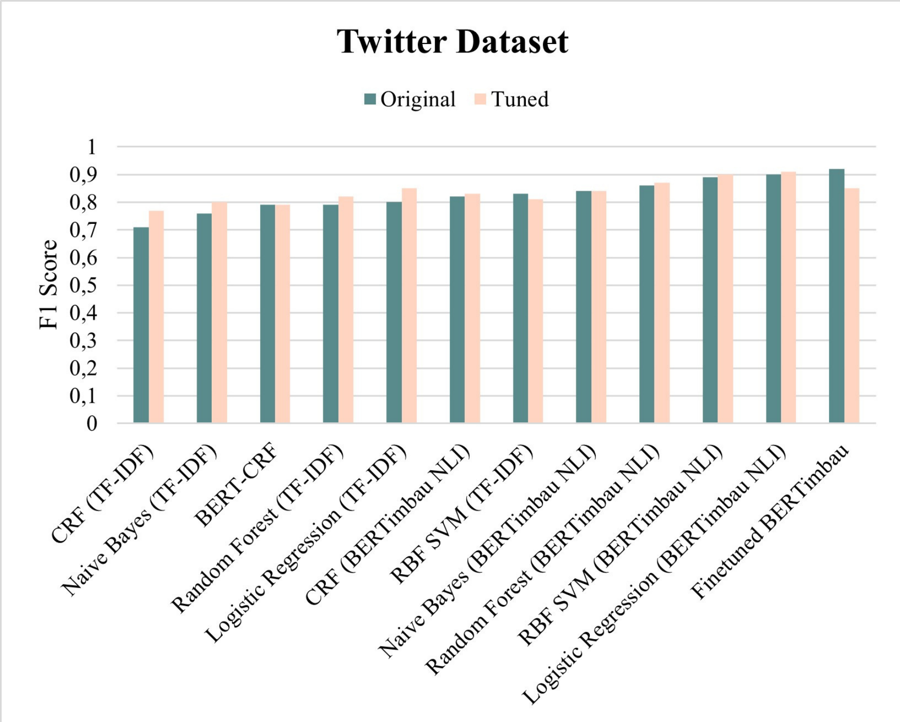


Figure 7.7: Comparison between the F1-Scores of the tuned and original Classifiers, using the Twitter dataset.

We will now list the hyperparameters' values used in the original and tuned classifiers:

- **Logistic Regression (TF-IDF)**:

Table 7.10: Comparison between the hyperparameters' values of the original and tuned Logistic Regression (TF-IDF)

Dataset	Classifier	C	Max. Iterations	Penalty	Solver
Twitter	Tuned	3.36	100	l2	LibLinear
	Original	1	100	l2	L-BFGS
Altice Labs	Tuned	11.29	100	l2	L-BFGS
	Original	1	100	l2	L-BFGS

  Looking at Table 7.10, the only common value between both datasets was the Max. Iterations, as the Penalty was a constant.

  The tuning process improved the performance of the classifiers.

- **Logistic Regression (BERTimbauNLI)**:

  Looking at Table 7.11, we can see that the only different value between both datasets was the Max. Iterations, in opposition to what happened with the Logistic Regression (TF-IDF). In fact, the original hyperparameters for the Twitter dataset are the same as the tuned hyperparameters, meaning the only difference between those approaches is the Cross-Validation.

  The performance of the tuned classifiers did not change much from the original classifiers, there was only a 0.01 shift in the F1 Score.

- **RBF SVM (TF-IDF)**:

Table 7.11: Comparison between the hyperparameters' values of the original and tuned Logistic Regression (BERTimbau NLI)

Dataset	Classifier	C	Max. Iterations	Penalty	Solver
Twitter	Tuned	1	100	l2	L-BFGS
	Original	1	100	l2	L-BFGS
Altice Labs	Tuned	1	200	l2	L-BFGS
	Original	1	100	l2	L-BFGS

The hyperparameters obtained with the Twitter dataset were used with the AL dataset, as the SVM training was too computationally expensive to go through the cross-validation and tuning operations.

The hyperparameters' values used were:

- C: 100 in the tuned classifiers, and 1 in the original classifiers;
- Gamma: 0.01 in the tuned classifiers, and "scale"[2] in the original classifiers.

There was no change in the performance with the AL dataset. With the Twitter dataset, the tuned classifier performed slightly worse (-0.02 in the F1 score).

- **RBF SVM (BERTimbau NLI)**:

  The hyperparameters obtained with the Twitter dataset were used with the AL dataset, as the RBF SVM training was too computationally expensive to go through the cross-validation and tuning operations.

  The hyperparameters' values used were:

  - C: 10 in the tuned classifiers, and 1 in the original classifiers
  - Gamma: 0.01 in the tuned classifiers, and "scale" in the original classifiers.

  There was no change in the performance with the AL dataset. With the Twitter dataset, the tuned classifier performed slightly better (+0.01 in the F1 score).

- **Naive Bayes (TF-IDF)**:

  For both datasets and both the tuned and original classifiers, the value of Alpha was set to 1, meaning the classifiers used the Laplace smoothing.

  There was no change in the performance with the AL dataset. With the Twitter dataset, the tuned classifier performed slightly better (+0.04 in the F1 score). The difference in performance, in this case, is only due to the application of Cross-Validation.

- **Naive Bayes (BERTimbau NLI)**:

  For both datasets and both the tuned and original classifiers, the value of Alpha was set to 1, meaning the classifiers used the Laplace smoothing.

  There was no change in the performance with the Twitter dataset. With the AL dataset, the tuned classifier performed slightly worse (-0.01 in the F1 score). The difference in performance, in this case, is only due to the application of Cross-Validation.

- **Random Forest (TF-IDF)**:

[2] `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html`

Table 7.12: Comparison between the hyperparameters' values of the original and tuned Random Forest (TF-IDF)

Dataset	Classifier	Criterion	Max. Depth	Max. Features	Min. Samples Leaf	Min. Samples Split	Estimators	Bootstrap
Twitter	Tuned	Gini	50	log2	1	5	200	True
	Original	Gini	None	sqrt	1	2	100	True
Altice Labs	Tuned	Gini	50	sqrt	2	2	100	False
	Original	Gini	None	sqrt	1	2	100	True

Looking at Table 7.12, the only common values between both datasets were the Criterion and the Max. Depth.

There was no change in the performance with the AL dataset. With the Twitter dataset, the tuned classifiers performed slightly better (+0.03 in the F1 score).

- **Random Forest (BERTimbau NLI)**:

Table 7.13: Comparison between the hyperparameters' values of the original and tuned Random Forest (BERTimbau NLI)

Dataset	Classifier	Criterion	Max. Depth	Max. Features	Min. Samples Leaf	Min. Samples Split	Estimators	Bootstrap
Twitter	Tuned	Entropy	50	log2	1	2	100	False
	Original	Gini	None	sqrt	1	2	100	True
Altice Labs	Tuned	Entropy	10	sqrt	2	5	100	False
	Original	Gini	None	sqrt	1	2	100	True

Looking at Table 7.13, the only common values between both datasets were the Criterion, the Bootstrap usage, and the Estimators. It is interesting to note that using sparse encoding, the classifiers performed better with the Gini criterion, while for dense encoding, they performed better with the Entropy criterion.

The performances with both datasets improved with the tuning process, even if only by 0.01 in the F1 scores.

- **CRF (TF-IDF)**:

Table 7.14: Comparison between the hyperparameters' values of the original and tuned CRF (TF-IDF)

Dataset	Classifier	C1	C2	Max. Iterations
Twitter	Tuned	0.21	0.22	200
	Original	0	0	Unlimited
Altice Labs	Tuned	0.2	0.33	100
	Original	0	0	Unlimited

Looking at Table 7.14, we can see that there are no common hyperparameters' values between both datasets, although the C1 value is very similar.

In these classifiers, the performances with both datasets were improved with the tuning process, by 0.06 (largest difference so far) and 0.03 in the Twitter and AL datasets, respectively.

- **CRF (BERTimbau NLI)**:

Looking at Table 7.15, we can see that, again, there are no common hyperparameters' values between both datasets. Interestingly, the Twitter dataset seems to perform better with more iterations, as opposed to the AL dataset.

Table 7.15: Comparison between the hyperparameters' values of the original and tuned CRF (BERTimbau NLI)

Dataset	**Classifier**	**C1**	**C2**	**Max. Iterations**
Twitter	Tuned	0.41	0.56	200
	Original	0	0	Unlimited
Altice Labs	Tuned	1.8	0.67	100
	Original	0	0	Unlimited

The performance of the tuned classifiers did not change much from the original classifiers, there was only a 0.01 shift in the F1 Score.

- **Fine-tuned BERTimbau**:

  The hyperparameters obtained with the Twitter dataset were used with the AL dataset, as the tuning process was too computationally expensive to go through the cross-validation and tuning operations.

  The hyperparameters' values used were:

  - Batch Size: 16; (tuned)
  - Epochs: 2;
  - Learning Rate: 0.0001;
  - Epsilon: 0.00000001.

  The tuned classifiers performed worse than the originals, with a 0.07 and a 0.03 decrease using the Twitter dataset and the AL dataset, respectively. So far, a 0.07 shift is the largest difference in performance between the tuned and original classifiers.

- **BERT-CRF**:

  The hyperparameters obtained with the individual models (Fine-tuned BERTimbau and CRF) were used for this combination, as the tuning process was too computationally expensive to go through the cross-validation and tuning operations.

  The values were already presented in Section 6.7, so they will not be repeated.

  There was no change in the performance of the tuned and original classifiers, regarding the F1 Score.

The results presented regard only the no-context approach (for the classifiers that do not consider context, originally). This is because context, for the most part, did not improve the performance of the classifiers, and the tuning process is computationally expensive.

7.6 Analysis of the Experimental Results

The lexicon-based classifiers (Tables 7.2 and 7.3) defined baseline results for the task of SA in the context of dialogue. Most models performed worse than random guessing (i.e., results lower than 50%), with SentiLex-flex being the exception when applied to the Twitter dataset. Both lexicon-based classifiers achieve higher F1 Scores when applied to this dataset.

All the shallow learning classifiers with sparse representations (Tables 7.4 and 7.5) perform better than the baseline models, but most do not achieve Recall and F1 Scores higher than 80%. For the AL dataset and Twitter dataset, Random Forest and SVM are the best models, respectively. There is a small difference between the performance of Random Forest and SVM in the AL dataset, so we could say that the best shallow learning classifier with sparse representations is the SVM model, overall. Every classifier performs better when applied to the Twitter dataset.

When the shallow learning models use a dense representation (Tables 7.6 and 7.7) they all achieve higher performances than their sparse representation equivalents. The Logistic Regression classifier achieves higher results in both datasets, followed by the SVM classifier. Most models present Recall and F1 Scores higher than 80%, with maximum values of 90%. Every classifier performs better when applied to the Twitter dataset.

From the deep learning models (Tables 7.8 and 7.9), only the Fine-tuned BERTimbau classifier performed better than the shallow learning models with dense representations. The BOTSchool API achieves higher Recall and AUC scores in the AL dataset than the Twitter dataset, contrary to the current tendency. Zero-SL also performs better when applied to the AL dataset, possibly due to how messages are written on Twitter, as already discussed. Focusing on the BOTSchool API, its performance is somewhat similar to that of the shallow learning classifiers with sparse representations. The former achieves higher Recall scores than the latter classifiers, but lower AUC and F1 scores.

When context is considered, the CRF, BERT-CRF, and Fine-tuned BERTimbau perform better than the remaining classifiers. The first achieves higher Recall scores than the third in the AL dataset, but lower AUC and F1 scores. In the Twitter dataset, the Fine-tuned BERTimbau is the best classifier, with a Recall, AUC and F1 scores higher than 90%. When applied to the AL dataset, this classifier achieves Recall and F1 Scores of around 80%, so the difference in performance between datasets is significant.

Context has decreased the performances of most classifiers, with the sole exception of the Fine-tuned BERTimbau model, when applied to the Twitter dataset. The BOTSchool API is one of the models in which the performance decreases the most when considering context (-27% in the AL dataset, and -24% in the Twitter dataset).

The optimization (when applicable) of the classifiers did not improve their performance significantly, the largest improvement being 6% in the CRF model with sparse representation, applied to the Twitter dataset. In the AL dataset, the Fine-tuned BERTimbau performed better than the remaining tuned models, while in the Twitter dataset, the Logistic Regression performed better.

Finally, we can affirm that, in these datasets, the shallow learning classifiers with dense representations, the Fine-tuned BERTimbau, and the CRF and BERT-CRF classifiers all performed better than the solution being considered by AL, BOTSchool API. The high decrease (especially in the AL dataset) when considering the previous utterances in this solution makes this one of the worst models to use with context. This could be related to the classifier using standard SA data instead of dialogues, which could make it harder to analyse context.

Regarding the remaining experiments, introduced in Chapter 6, the experimental results suggest that:

- The classifiers benefit from the use of BERTimbau NLI's dense representations;
- The use of multi grams instead of unigrams is only beneficial in the shallow learning

classifiers, and when considering context;

- The choice of fewer features (removing HasQuestion and HasExclamation) in the CRF model did not have a considerable impact on the model's performance;
- The BERT-CRF model performed better when using the representation obtained in the last hidden state;
- The Zero-SL model performed better when using labels as hypotheses instead of sentences;
- The models trained using non-dialogue data (sentituites) perform significantly worse than when trained using dialogues.

This last item, and the number of solutions that performed better than the BOTSchool API (which was also trained in standard SA data), suggest that it is important to train the models in dialogues to perform SA in this type of data.

This concludes the experimentation and evaluation of all models developed during the course of this dissertation. The conclusions derived from this work will be presented in the next chapter.

Chapter 8

Conclusions and Future Work

Sentiment Analysis (SA) is a domain that still faces many challenges, mainly because there are many different languages, dialects, and words that are continuously being invented and dropped, or their meanings keep evolving, making it a very dynamic field. Besides this, the perception of sentiments and emotions can be very subjective, which can make the classification process complex.

In this Chapter, we will present the conclusions taken based on the work and analysis developed, as well as set the possible future work.

8.1 Conclusions

Going over the objectives and contributions mentioned in Chapter 1, we can say that we achieved the goals set for this dissertation and contributed to the SA field.

Keeping the first objective in mind, we created, annotated, and analysed two datasets (referred to as the Altice Labs (AL) dataset and the Twitter dataset) containing dialogues in the Portuguese language. It was highly important to provide the models with samples similar to the ones found in real data, which, for this problem, would mean dialogues in Portuguese and related to the domains of eCommerce, TV, Health Care, Finance and FinTech, and Telecommunications. The use of Twitter for data extraction checks all these requirements, which made it a good option for the creation of data.

The two datasets were manually labeled, with the contribution of 14 annotators. The evaluation of the agreement level between them for each block of data revealed that the average agreement was just of moderate level. This could be expected, as we mentioned that the classification of sentiment can be very subjective. However, this is likely to have harmed the models trained on these datasets, because similar utterances were classified differently, making it harder to detect and generalize the sentiment.

The Twitter dataset was made available[1] in accordance with Twitter's Content Redistribution Terms, which allow the publishing of the Twitter IDs only.

Regarding the second objective, we experimented with several models and techniques for SA and Natural Language Processing (NLP), which is confirmed by the high number of classifiers, with varying complexity, developed and evaluated during the course of this dissertation. From the comparison of the performances of each classifier, we could derive

[1] `https://github.com/NLP-CISUC/twitter_sentiment_analysis`

the following conclusions:

- The analysis of the related work for this dissertation made it clear that the use of Bidirectional Encoder Representations from Transformers (BERT) or fine-tuned BERT models seemed to perform better than solutions that did not include these approaches. In fact, we confirmed that the fine-tuned BERTimbau classifier was the model with better performance, for both datasets;

- Keeping in mind the importance of BERT, we also saw improvements from using the dense representations (obtained with BERTimbau Natural Language Inference (NLI)) over the sparse representations (obtained with Term Frequency - Inverse Document Frequency (TF-IDF)). An interesting conclusion regarding the use of this dense representation was that the evaluation of the classifiers with context (Figures 7.2 and 7.3), considering the three most important metrics, revealed that the Conditional Random Field (CRF) model with the BERTimbau NLI representation, performs similarly to the fine-tuned BERTimbau, enforcing the importance of semantically meaningful embeddings for NLP. In fact, in all the comparisons performed, the Fine-tuned BERTimbau model is usually followed by a classifier with this type of representation;

- Most models with TF-IDF encoding did not benefit from the use of multi-grams over unigrams, except when context (the concatenation of the current and previous sentences) was involved, possibly due to the use of similar expressions or words over the dialogue;

- The results from using models trained in non-dialogue data and applying them to our datasets, as well as the number of options that perform better than the BOTSchool Application Programming Interface (API) solution, also trained in this type of data, suggest that to achieve a good performance in the classification of SA in dialogue, it is important that the classifiers are trained in this non-standard type of data. These results further emphasize the importance of the creation of our datasets, as they allow for better performance when compared to the two existing options tested (Sentituites and BOTSchool API);

- An analysis regarding the use of context revealed that, for these datasets, the inclusion of the previous utterances actually harmed the performance of the models, contrary to what could have been expected. An exception is the Fine-tuned BERTimbau model, which actually performed slightly better when considering context (as seen in Figure 7.5), however, this only happened in the Twitter dataset. In the AL dataset, we can argue that a possibility for context not being helpful in the classification of sentiment in dialogues, is that the conversations are with a chatbot, making them less emotionally rich, especially when the user knows they are not speaking with another person, as is the case. A possibility for these results in the Twitter dataset, is that the dialogues are mostly small (an average of 2.52 turns per dialogue), which could make it so that the conversation does not develop enough for context to be beneficial to the classification task;

- Overall, and despite the difference in the size of both datasets, the models trained on the Twitter dataset generally perform better than those trained on the AL dataset. This may be due to the quality of the data samples, since the latter dataset suffers from some limitations due to the underlying speech-to-text conversion, making some utterances hard to understand even for people (confirmed during the annotation process);

- The tuning process did not have a large impact on the performance of the classifiers, the biggest difference being an increase of 0.06 in the F1 Score for the CRF model with sparse representation. However, this could be due to the choice of values and the use of grid search, which may not be the best option for the tuning of hyper-parameters. Without further experimentation, we cannot confirm that the models would not benefit from the tuning process, but this did not happen (with significant improvements) for the values considered.

Regarding the third objective, there are several options that, for our datasets, perform better than the current solution in use by AL. Considering the F1 Score metric, these options include a Logistic Regression model with dense representations, a Random Forest model with sparse representations, a Fine-tuned BERTimbau model, and the solutions that originally consider context, especially a CRF model with dense representations, and a BERT-CRF model. Any of these options could improve the way AL is able to communicate with its clients, by better identifying the expression of negative sentiment in the dialogues.

Regarding the fourth objective, two scientific articles were written during the course of this dissertation: one was submitted to the IberSPEECH 2022 Conference[2], and the other to the RECPAD Conference[3]. The notifications of acceptance will only be sent by the end of September, so at the time of the submission of this dissertation, these results are not yet known.

This analysis allows us to confirm that the objectives set for this work were achieved, and the approach followed allowed for several contributions to the SA field:

- Creation of two labeled datasets for SA in Portuguese dialogues. One of which is publicly available;
- Presentation of solutions with some degree of innovation;
- Extensive experimentation and evaluation of different models and techniques for SA in dialogues, which led to most of the conclusions presented above;
- Identification of models that perform better than AL' current solution. If they choose to integrate any of the models in their system, it would be important to define a negativity threshold that, if surpassed, would activate the fallback system and notify the company that a client required personalized attention;
- Submission of two scientific articles into NLP-related conferences.

Nevertheless, there are still several possible tasks for future work, which will be presented in the following section.

8.2 Future Work

This section presents the tasks considered for future work, in relation to the data (creation and annotation) and the models (development, optimization, evaluation), as well as general possible improvements and experiments.

[2]http://iberspeech2022.ugr.es/
[3]https://recpad2022.ipleiria.pt/

Regarding the creation of data, the use of a Wizard of Oz (WOZ) framework could generate interesting data and even more fitting to our interests than the Twitter data.

Regarding the annotation of data, there are five possible options, where three of them mostly aim at improving the moderate annotation agreement obtained for the created datasets.

- The use of an active learning approach (Settles, 2009) (e.g., applying uncertainty sampling, where the active learner queries the instances about which it is least certain how to label, while labeling the others) could be beneficial to either improve the current annotations or to enlarge the created datasets, by annotation the remaining data;

- The annotators involved in this work could gather to discuss and review the annotations with a lower agreement;

- The data generated using the Generative Pre-trained Transformer (GPT)-2 model could be evaluated, selected, and annotated. The AL dataset cannot be publicly shared, but it could be useful in the annotation process of the generated data since the samples would be very similar. Hence, the similarity between the annotated and non-annotated utterances could be computed (e.g., using cosine similarity) to perform a match of the labels, when the similarity metric was high;

- The datasets could be labeled for emotion instead of sentiment polarity, which could allow for interesting comparisons between both approaches. Another option would be to adapt existing datasets with emotion annotations;

- The analysis of the related work studied for this dissertation showed us that the joint tasks of SA and Dialogue Act Recognition (DAR) can provide better results than SA (or DAR) alone. However, we could not confirm this since we had no annotated data for DAR. Hence, the annotation of dialogue acts present in our datasets could allow for the development of classifiers with these joint tasks, and possibly better performances than those achieved with just SA in mind.

Regarding the development of classifiers and their optimization, there are four possible options:

- The application of feature reduction techniques in the classifiers that are computationally heavy and do not handle large datasets too well (e.g., BERT-CRF, Fine-tuned BERTimbau). We could use the t-distributed Stochastic Neighbor Embedding (t-SNE) technique presented for visualization purposes, or others such as the Principal Component Analysis (PCA), a common feature reduction approach, so that the models could run more smoothly;

- The Few-Shot Learning (Few-SL) and Zero-Shot Learning (Zero-SL) classifiers could be explored with different base models, decoding algorithms and descriptive sentences (in the case of the former), and trying to find other semantically meaningful sentences (in the case of the latter). If good results were achieved, the models could be used to classify the remaining unlabeled data. Since the low Twitter results may be derived from the use of abbreviations and misspellings, the former could be converted to their original forms, and a spell checker can be applied to correct the latter;

- An intriguing approach would be to use Ordinal Regression (Saad and Yang, 2019) in a multi-class setting. What this would do is perform a better analysis of the errors, since it would compute a higher loss if a Very Negative sample was classified as Positive than it would if it were classified as Negative, which could provide interesting results in the scenario of this dissertation;

- Since there were few improvements using a grid search approach, it could be beneficial to explore the Genetic Algorithm (GA) search or the Population-Based Training (PBT) approaches.

Regarding the evaluation of the classifiers' performances, it could be useful to test how well the models trained on Twitter data could classify the AL data. This would also let us know if we succeeded in trying to extract data similar to the real data.

Regarding some of the problems faced with the AL data, we mentioned that sometimes the client would repeat the same utterance. This was considered in the annotation process, where the annotators were told to decrease the sentiment when this was detected. However, more work could be done that could potentially improve the solutions. This repetition could happen because there are specific topics that do not get correctly replied to. To tackle this, an analysis of the wrongly classified utterances could be performed, applying a clustering algorithm to it (the use of BERTimbau NLI could be interesting for this purpose since sentence transformers would extract semantically meaningful embeddings). From the clustered data we could possibly extract some common topics, and then we could train the model with data related to that topic, hopefully improving the interactions with the clients.

Generally speaking, the development of a tool capable of analyzing dialogues and classifying the sentiment in their utterances could be interesting and a good contribution to the SA field.

Finally, another option that could be interesting, but not as related to the topic at hand, would be an aspect-based approach, which would allow us to understand if there are any specific aspects related to AL that users react to in a more negative (or positive) way than to other aspects, such as *price* or *fibra*.

Throughout this chapter, we could see that a lot of work was done, resulting in several experiments and conclusions, but that many other approaches could have been used. This is great news for the SA and NLP fields because it means that there is still a lot to explore (and hopefully enhance or invent) in these topics.

References

Adibi, F., Majidi, B., and Eshghi, M. (2018). Personalized Advertisement in the Video Games Using Deep Social Network Sentiment Analysis. In *2018 2nd National and 1st International Digital Games Research Conference: Trends, Technologies, and Applications (DGRC)*, pages 104–108. IEEE.

Aggarwal, C. C. (2014a). An Introduction to Data Classification. In *Data Classification: Algorithms and Applications*, pages 1–36. CRC Press.

Aggarwal, C. C. (2014b). Instance-Based Learning: A Survey. In *Data Classification: Algorithms and Applications*, pages 157–186. CRC Press.

Almeida, R. J. A. (2018). LeIA - Léxico para Inferência Adaptada. `https://github.com/rafjaa/LeIA`.

Altice Portugal (2020). History: Historical evolution of Altice Portugal. `https://https://www.telecom.pt/en-us/a-pt/pages/historia.aspx`. Accessed: 2021-12-23.

Amin, A., Hossain, I., Akther, A., and Alam, K. M. (2019). Bengali vader: A sentiment analysis approach using modified vader. In *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, pages 1–6. IEEE.

Asani, E., Vahdat-Nejad, H., and Sadri, J. (2021). Restaurant recommender system based on sentiment analysis. *Machine Learning with Applications*, 6:100114.

Baccianella, S., Esuli, A., and Sebastiani, F. (2010). Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10*.

Balage Filho, P., Pardo, T. A. S., and Aluísio, S. (2013). An evaluation of the Brazilian Portuguese LIWC dictionary for sentiment analysis. In *Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology*.

Barbieri, F., Espinosa-Anke, L., and Camacho-Collados, J. (2022). XLM-T: Multilingual Language Models in Twitter for Sentiment Analysis and Beyond. *Proceedings of the LREC, Marseille, France*, pages 20–25.

Bernsen, N. O., Dybkjær, H., and Dybkjær, L. (1994). Wizard of OZ prototyping: How and when. *Proc. CCI Working Papers Cognit. Sci./HCI, Roskilde, Denmark*.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Brown, P. F., Cocke, J., Della Pietra, S. A., Della Pietra, V. J., Jelinek, F., Lafferty, J., Mercer, R. L., and Roossin, P. S. (1990). A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Budzianowski, P., Wen, T.-H., Tseng, B.-H., Casanueva, I., Ultes, S., Ramadan, O., and Gasic, M. (2018). MultiWOZ – A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Burkov, A. (2020). *Machine Learning Engineering.* True Positive Incorporated.

Carvalho, P., Sarmento, L., Teixeira, J., and Silva, M. J. (2011). Liars and saviors in a sentiment annotated corpus of comments to political debates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 564–568.

Carvalho, P. and Silva, M. (2015). SentiLex-PT: Principais características e potencialidades. *Oslo Studies in Language*, 7(1).

Cerisara, C., Jafaritazehjani, S., Oluokun, A., and Le, H. T. (2018). Multi-task dialog act and sentiment recognition on Mastodon. *CoRR*, abs/1807.05013.

Chang, M.-W., Ratinov, L.-A., Roth, D., and Srikumar, V. (2008). Importance of Semantic Representation: Dataless Classification. In *Aaai*, volume 2, pages 830–835.

Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46.

Costa, C., Aparicio, M., and Aparicio, J. (2021). Sentiment Analysis of Portuguese Political Parties Communication. In *The 39th ACM International Conference on Design of Communication*, pages 63–69.

Cowen, A. S. and Keltner, D. (2017). Self-report captures 27 distinct categories of emotion bridged by continuous gradients. *Proceedings of the National Academy of Sciences*, 114(38):E7900–E7909.

Dale, R. (2016). The return of the chatbots. *Natural Language Engineering*, 22(5):811–817.

Deng, H., Sun, Y., Chang, Y., and Han, J. (2014). Probabilistic models for classification. In *Data Classification: Algorithms and Applications*, pages 65–86. CRC Press.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805.*

Dinan, E., Roller, S., Shuster, K., Fan, A., Auli, M., and Weston, J. (2018). Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241.*

Domingos, P. (2015). *The master algorithm: How the quest for the ultimate learning machine will remake our world.* Basic Books.

Dosciatti, M. M., Ferreira, L., and Paraiso, E. (2013). Identificando emoçoes em textos em Português do Brasil usando máquina de vetores de suporte em soluçao multiclasse. *ENIAC-Encontro Nacional de Inteligência Artificial e Computacional. Fortaleza, Brasil.*

Duarte, E. S. (2013). Sentiment analysis on Twitter for the Portuguese language. Master's thesis, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa.

Duarte, L., Macedo, L., and Gonçalo Oliveira, H. (2019). Exploring Emojis for Emotion Recognition in Portuguese Text. In *Proceedings of the 19th EPIA Conference on Artificial Intelligence*, volume 11805 of *LNCS/LNAI*, pages 719–730. Springer.

Einsenstein, J. (2018). *Natural Language Processing*. MIT press.

Ekman, P. (1992). An argument for basic emotions. *Cognition and Emotion*, 6(3-4):169–200.

Fan, A., Lewis, M., and Dauphin, Y. (2018). Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*.

Firth, J. (1957). A Synopsis of Linguistic Theory 1930-1955. In *Studies in Linguistic Analysis*. Philological Society, Oxford. reprinted in Palmer, F. (ed. 1968) Selected Papers of J. R. Firth, Longman, Harlow.

Fleiss, J. L. (1971). Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Freitas, C., Motta, E., Milidiú, R., and César, J. (2014). *Sparkling Vampire... lol! Annotating Opinions in a Book Review Corpus*, pages 128–146. Cambridge Scholars Publishing.

Freund, Y., Schapire, R., and Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612.

Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media.

Ghosh, A., Li, G., Veale, T., Rosso, P., Shutova, E., Barnden, J., and Reyes, A. (2015). Semeval-2015 Task 11: Sentiment Analysis of Figurative Language in Twitter. In *International Workshop on Semantic Evaluation*, pages 470–478.

Gonçalo Oliveira, H., Ferreira, P., Martins, D., Silva, C., and Alves, A. (2022). A Brief Survey of Textual Dialogue Corpora. In *Proceedings of the 13th Language Resources and Evaluation Conference*, LREC 2022, pages 1264–1274, Marseille, France. ELRA.

Gonçalves, S. C. L. (2019). Projeto ALIP (Amostra Linguística do Interior Paulista) e banco de dados Iboruna: 10 anos de contribuição com a descrição do Português Brasileiro. *Estudos Linguísticos (São Paulo. 1978)*, 48(1):276–297.

Hammes, L. O. A. and de Freitas, L. A. (2021). Utilizando BERTimbau para a Classificação de Emoções em Português. In *Anais do XIII Simpósio Brasileiro de Tecnologia da Informação e da Linguagem Humana*, pages 56–63. SBC.

Howcroft, D. M., Belz, A., Clinciu, M.-A., Gkatzia, D., Hasan, S. A., Mahamood, S., Mille, S., Van Miltenburg, E., Santhanam, S., and Rieser, V. (2020). Twenty years of confusion in human evaluation: NLG needs evaluation sheets and standardised definitions. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 169–182.

Hu, M. and Liu, B. (2004). Mining and Summarizing Customer Reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 168–177, New York, NY, USA. Association for Computing Machinery.

Hutto, C. and Gilbert, E. (2014). VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text. *Proceedings of the International AAAI Conference on Web and Social Media*, 8(1):216–225.

Jaderberg, M., Dalibard, V., Osindero, S., Czarnecki, W. M., Donahue, J., Razavi, A., Vinyals, O., Green, T., Dunning, I., Simonyan, K., et al. (2017). Population based training of neural networks. *arXiv preprint arXiv:1711.09846*.

Jianqiang, Z., Xiaolin, G., and Xuejun, Z. (2018). Deep convolution neural networks for twitter sentiment analysis. *IEEE Access*, 6:23253–23260.

Jin, Y., Xie, J., Guo, W., Luo, C., Wu, D., and Wang, R. (2019). LSTM-CRF neural network with gated self attention for Chinese NER. *IEEE Access*, 7:136694–136703.

Junior, A. C., Casanova, E., Soares, A., de Oliveira, F. S., Oliveira, L., Junior, R. C. F., da Silva, D. P. P., Fayet, F. G., Carlotto, B. B., Gris, L. R. S., et al. (2021). CORAA: a large corpus of spontaneous and prepared speech manually validated for speech recognition in Brazilian Portuguese. *arXiv preprint arXiv:2110.15731*.

Jurafsky, D. and Martin, J. H. (2021). *Speech and Language Processing*. Pearson Prentice Hall/Pearson education international, 2nd edition. (3rd edition draft, 2021, at `https://web.stanford.edu/~jurafsky/slp3/`).

Kelley, J. F. (1984). An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems (TOIS)*, 2(1):26–41.

Kotsiantis, S. B. (2007). Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160(1):3–24.

Kouloumpis, E., Wilson, T., and Moore, J. (2021). Twitter Sentiment Analysis: The Good the Bad and the OMG! *Proceedings of the International AAAI Conference on Web and Social Media*, 5(1):538–541.

Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., and Brown, D. (2019). Text classification algorithms: A survey. *Information*, 10(4):150.

Krippendorff, K. (1970). Estimating the reliability, systematic error and random error of interval data. *Educational and Psychological Measurement*, 30(1):61–70.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Lamsal, R. (2020). Coronavirus (COVID-19) Tweets Dataset. `https://dx.doi.org/10.21227/781w-ef42`.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. *CoRR*, abs/1910.13461.

Li, J., Fei, H., and Ji, D. (2020). Modeling Local Contexts for Joint Dialogue Act Recognition and Sentiment Classification with Bi-channel Dynamic Convolutions. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 616–626, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Li, Y., Su, H., Shen, X., Li, W., Cao, Z., and Niu, S. (2017). DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset. *CoRR*, abs/1710.03957.

Liu, Y. and Lapata, M. (2019). Text Summarization with Pretrained Encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Lowe, R., Pow, N., Serban, I. V., and Pineau, J. (2015). The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294.

Madanian, S., Airehrour, D., Samsuri, N. A., and Cherrington, M. (2021). Twitter Sentiment Analysis in Covid-19 Pandemic. In *2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 0399–0405. IEEE.

Mandrekar, J. N. (2010). Receiver operating characteristic curve in diagnostic test assessment. *Journal of Thoracic Oncology*, 5(9):1315–1316.

Manning, C. D., Schütze, H., and Raghavan, P. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Mendes, R. B. and Oushiro, L. (2013). Mapping paulistano portuguese: the sp2010 project. In Mello, H., Pettorino, M., and Raso, T., editors, *Proceedings of the VIIth GSCP International Conference*. Firenze University Press.

Mi, F., Wang, Y., and Li, Y. (2022). CINS: Comprehensive Instruction for Few-Shot Learning in Task-Oriented Dialog Systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11076–11084.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

Miller, E. G., Matsakis, N. E., and Viola, P. A. (2000). Learning from one example through shared densities on transforms. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 1, pages 464–471. IEEE.

Mohammad, S., Kiritchenko, S., Sobhani, P., Zhu, X., and Cherry, C. (2016). Semeval-2016 Task 6: Detecting Stance in Tweets. In *International Workshop on Semantic Evaluation*, pages 31–41.

Mohammad, S. M. and Turney, P. D. (2013). Crowdsourcing a Word-Emotion Association Lexicon. *Computational Intelligence*, 29(3):436–465.

Neeraj, T. (2020). Feature-based Approach with BERT. *trishalaneeraj.github.io*.

Nielsen, F. Å. (2011). A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. *arXiv preprint arXiv:1103.2903*.

Novikova, J., Dusek, O., Curry, A. C., and Rieser, V. (2017). Why We Need New Evaluation Metrics for NLG. In *2017 Conference on Empirical Methods in Natural Language Processing*, pages 2231–2242. Association for Computational Linguistics.

Oliviera Jr., M. (2016). NURC Digital Um protocolo para a digitalização, anotação, arquivamento e disseminação do material do Projeto da Norma Urbana Linguística Culta (NURC). *CHIMERA: Revista de Corpus de Lenguas Romances y Estudios Lingüísticos*, 3(2):149–174.

Otter, D. W., Medina, J. R., and Kalita, J. K. (2021). A Survey of the Usages of Deep Learning for Natural Language Processing. *IEEE Transactions on Neural Networks and Learning Systems*, 32(2):604–624.

Pak, A. and Paroubek, P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10, pages 1320–1326.

Patil, N., Patil, A., and Pawar, B. (2020). Named entity recognition using conditional random fields. *Procedia Computer Science*, 167:1181–1188.

Pearl, L. and Steyvers, M. (2010). Identifying Emotions, Intentions, and Attitudes in Text Using a Game with a Purpose. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, CAAGET '10, page 71–79, USA. Association for Computational Linguistics.

Pennington, J., Socher, R., and Manning, C. (2014). GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Pereira, D. A. (2021). A survey of sentiment analysis in the Portuguese language. *Artificial Intelligence Review*, 54(2):1087–1115.

Plutchik, R. (1984). Emotions: A general psychoevolutionary theory. *Approaches to emotion*, 1984:197–219.

Pontiki, M., Galanis, D., Papageorgiou, H., Androutsopoulos, I., Manandhar, S., Al-Smadi, M., Al-Ayyoub, M., Zhao, Y., Qin, B., De Clercq, O., et al. (2016). SemEval-2016 Task 5: Aspect Based Sentiment Analysis. In *International Workshop on Semantic Evaluation*, pages 19–30.

Qin, L., Che, W., Li, Y., Ni, M., and Liu, T. (2020a). DCR-Net: A Deep Co-Interactive Relation Network for Joint Dialog Act Recognition and Sentiment Classification. *CoRR*, abs/2008.06914.

Qin, L., Li, Z., Che, W., Ni, M., and Liu, T. (2020b). Co-GAT: A Co-Interactive Graph Attention Network for Joint Dialog Act Recognition and Sentiment Classification. *CoRR*, abs/2012.13260.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training. Preprint available at `https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf`.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Rambocas, M. and Gama, J. (2013). Marketing Research: The Role Of Sentiment Analysis. FEP Working Papers 489, Universidade do Porto, Faculdade de Economia do Porto.

Raso, T. and Mello, H. (2012). *C-oral - Brasil I: Corpus de Referência do Português Brasileiro Falado Informal*. Editora UFMG.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Rinaldi, A., Oseguera, O., Tuazon, J., and Cruz, A. C. (2017). End-to-end dialogue with sentiment analysis features. In *International Conference on Human-Computer Interaction*, pages 480–487. Springer.

Ritter, A., Cherry, C., and Dolan, W. B. (2010). Unsupervised modeling of Twitter conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 172–180.

Rosenthal, S., Farra, N., and Nakov, P. (2017). SemEval-2017 Task 4: Sentiment Analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, Vancouver, Canada. Association for Computational Linguistics.

Rosenthal, S., Nakov, P., Kiritchenko, S., Mohammad, S. M., Ritter, A., and Stoyanov, V. (2015). SemEval-2015 Task 10: Sentiment Analysis in Twitter. In *Proc. of the 9th International Workshop on Semantic Evaluation*, pages 451–463.

Roy, A., Guria, S., Halder, S., Banerjee, S., and Mandal, S. (2020). *Summarizing Opinions with Sentiment Analysis from Multiple Reviews on Travel Destinations*, pages 310–320. IGI Global.

Russell, J. A. (1980). A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161.

Saad, S. E. and Yang, J. (2019). Twitter Sentiment Analysis Based on Ordinal Regression. *IEEE Access*, 7:163677–163685.

Settles, B. (2009). Active learning literature survey. In *CS Technical Reports*. University of Wisconsin-Madison Department of Computer Sciences.

Severyn, A. and Moschitti, A. (2015). Twitter Sentiment Analysis with Deep Convolutional Neural Networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, page 959–962, New York, NY, USA. Association for Computing Machinery.

Sharma, P. and Moh, T.-S. (2016). Prediction of Indian election using sentiment analysis on Hindi Twitter. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 1966–1971.

Souza, F., Nogueira, R., and Lotufo, R. (2019). Portuguese named entity recognition using BERT-CRF. *arXiv preprint arXiv:1909.10649*.

Souza, F., Nogueira, R., and Lotufo, R. (2020). BERTimbau: Pretrained BERT Models for Brazilian Portuguese. In *Proceedings of the Brazilian Conference on Intelligent Systems (BRACIS 2020)*, volume 12319 of *LNCS*, pages 403–417. Springer.

Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Ess-Dykema, C. V., and Meteer, M. (2000). Dialogue Act Modeling for Automatic Tagging and Recognition of Conversational Speech. *Computational Linguistics*, 26(3):339–373.

Strapparava, C. and Mihalcea, R. (2007). Semeval-2007 Task 14: Affective Text. In *International Workshop on Semantic Evaluation*, pages 70–74.

Sullivan, G. M. and Artino Jr, A. R. (2013). Analyzing and interpreting data from Likert-type scales. *Journal of graduate medical education*, 5(4):541–542.

Sutton, C., McCallum, A., et al. (2012). An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373.

Taboada, M., Brooke, J., Tofiloski, M., Voll, K., and Stede, M. (2011). Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics*, 37(2):267–307.

Van der Maaten, L. and Hinton, G. (2008). Visualizing Data Using t-SNE. *Journal of Machine Learning Research*, 9(11).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. (2017). Graph attention networks. *arXiv preprint arXiv:1710.10903.*

von Platen, P. (2020). How to generate text: Using different decoding methods for language generation with Transformers.

Warjri, S., Pakray, P., Lyngdoh, S. A., and Maji, A. K. (2021). Part-of-speech (pos) tagging using conditional random field (crf) model for khasi corpora. *International Journal of Speech Technology*, 24(4):853–864.

Watt, J., Borhani, R., and Katsaggelos, A. K. (2020). *Machine learning refined: Foundations, algorithms, and applications*. Cambridge University Press.

Wen, T.-H., Gasic, M., Mrkšić, N., Barahona, L. M. R., Su, P.-H., Ultes, S., Vandyke, D., and Young, S. (2016). Conditional Generation and Snapshot Learning in Neural Dialogue Systems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2153–2162.

Wiebe, J. and Riloff, E. (2011). Finding Mutual Benefit between Subjectivity Analysis and Information Extraction. *IEEE Transactions on Affective Computing*, 2(4):175–191.

Williams, A., Nangia, N., and Bowman, S. (2018). A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82.

Xu, F. F., Alon, U., Neubig, G., and Hellendoorn, V. J. (2022). A systematic evaluation of large language models of code. In *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, pages 1–10.

Yan, L., Zheng, Y., and Cao, J. (2018). Few-shot learning for short text classification. *Multimedia Tools and Applications*, 77(22):29799–29810.

Zahiri, S. and Choi, J. D. (2018). Emotion Detection on TV Show Transcripts with Sequence-based Convolutional Neural Networks. In *Proceedings of the AAAI Workshop on Affective Content Analysis*, AFFCON'18, pages 44–51.

Zapf, A., Castell, S., Morawietz, L., and Karch, A. (2016). Measuring inter-rater reliability for nominal data–which coefficients and confidence intervals are appropriate? *BMC medical research methodology*, 16(1):1–10.

Zhu, X. and Goldberg, A. B. (2009). Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130.

Appendices

Appendix A

List of the information gathered about each tweet:

- *Conversation ID*: One tweet can spark different conversation threads. The conversation ID matches the ID of the tweet that started the conversation, and is present in all related tweets;
- *Author ID*: The unique ID that identifies the user who posted the tweet;
- *Tweet ID*: The unique ID that identifies the posted tweet;
- *Created At*: The date and time at which the tweet was posted;
- *Retweet Count*: The number of times the tweet was retweeted;
- *Reply Count*: The number of times the tweet was replied to;
- *Like Count:* The number of times someone liked the tweet;
- *Quote Count*: The number of times someone quoted the tweet;
- *Language*: The language of the tweet, if detected by Twitter;
- *Text*: The actual text from the tweet;
- *Referenced Type*: List containing the type (e.g., quoted, replied to) of the tweets this tweet refers to;
- *Referenced ID*: List containing the IDs of the tweets this tweet refers to;
- *In Reply To User ID*: If the tweet is a reply, this field contains the ID of the author of the tweet it replies to;
- *Verified*: If the user is verified by Twitter;
- *Username*: The unique screen name of the user;
- *Name*: Name of the user, as defined in their profile;
- *Followers Count*: Number of followers of the user;
- *Following Count*: Number of accounts that the user is following;
- *Tweet Count*: Number of tweets posted by the user;
- *Listed Count*: Number of lists that the user is a part of;
- *User Created At*: The time and date at which the user's account was created.

Appendix B

Full metric scores of the classifiers using context. They are split in categories according to the type of algorithm used. The second category, Deep Learning with "Context", receives as input the current and previous sentences only, which is why context is referred to that way. Table 1 presents these scores for the Altice Labs (AL) dataset, and Table 2 for the Twitter dataset.

Table 1: Evaluation of the classifiers with context using the AL dataset, grouped by categories

Altice Labs Dataset						
Model	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
CRF-based Classifiers						
CRF (BERTimbau NLI)	0.93	0.79	0.81	0.95	0.8	0.96
CRF (TF-IDF)	0.91	0.75	0.72	0.95	0.73	0.94
BERT-CRF	0.92	0.74	0.78	0.94	0.76	0.93
Deep Learning with "Context"						
Few-Shot Learning	0.67	0.22	0.58	0.69	0.32	N.A.
Zero-Shot Learning	0.8	0.45	0.58	0.85	0.51	0.74
Botschool API	0.84	0.58	0.4	0.94	0.47	0.8
Finetuned BERTimbau	0.93	0.90	0.77	0.98	0.83	0.96
Shallow Learning with BERTimbau NLI						
Logistic Regression	0.89	0.77	0.57	0.96	0.66	0.93
RBF SVM	0.9	0.87	0.5	0.98	0.64	0.92
Naive Bayes	0.84	0.55	0.44	0.92	0.49	0.83
Random Forest	0.9	0.88	0.5	0.99	0.64	0.93
Shallow Learning with TF-IDF						
Logistic Regression	0.88	0.79	0.41	0.98	0.54	0.9
RBF SVM	0.89	0.82	0.49	0.98	0.61	0.93
Naive Bayes	0.85	0.72	0.29	0.98	0.41	0.83
Random Forest	0.9	0.84	0.51	0.98	0.63	0.92

Table 2: Evaluation of the classifiers with context using the Twitter dataset, grouped by categories

Twitter Dataset						
Model	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
CRF-based Classifiers						
CRF (BERTimbau NLI)	0.88	0.87	0.78	0.94	0.82	0.92
CRF (TF-IDF)	0.8	0.74	0.68	0.86	0.71	0.86
BERT-CRF	0.86	0.74	0.85	0.87	0.79	0.91
Deep Learning with "Context"						
Few-Shot Learning	0.75	0.5	0.05	0.98	0.09	N.A.
Zero-Shot Learning	0.56	0.43	0.65	0.51	0.52	0.57
Botschool API	0.65	0.51	0.63	0.66	0.56	0.68
Finetuned BERTimbau	0.93	0.95	0.91	0.96	0.93	0.99
Shallow Learning with BERTimbau NLI						
Logistic Regression	0.81	0.75	0.71	0.87	0.73	0.88
RBF SVM	0.81	0.81	0.62	0.92	0.7	0.86
Naive Bayes	0.72	0.59	0.71	0.73	0.64	0.8
Random Forest	0.78	0.69	0.69	0.83	0.69	0.83
Shallow Learning with TF-IDF						
Logistic Regression	0.76	0.67	0.64	0.82	0.65	0.81
RBF SVM	0.74	0.64	0.63	0.81	0.63	0.8
Naive Bayes	0.73	0.66	0.5	0.86	0.57	0.76
Random Forest	0.76	0.64	0.73	0.77	0.68	0.86

Appendix C

Packages used throughout the development stage of this dissertation:

Table 3: List of packages used in the development of this work, and their descriptions.

Package	Description	Version
TwitterAPI	Allows access to Twitter	2.7.5
requests	Allows HTTP requests to be sent	2.27.1
json	Allows writing and reading .json files	2.0.9
os	Allows access to operating system functionalities (e.g., setting environment variables)	3.8.13
time	Allows the measure of execution time	3.8.13
pandas	Allows access to more data structures (e.g., dataframes)	1.4.2
numpy	Supports multi-dimensional data structures (e.g., tensors)	1.22.3
torch	Allows tensor computation and an auto-gradient system for deep neural networks	1.11.0
nltk	Allows access to Natural Language Processing techniques (e.g., tokenization)	3.7
sklearn	Allows access to machine learning models and techniques (e.g., cross-validation, SVMs)	1.1.1
matplotlib	Allows the visualization of data	3.5.1
seaborn	Allows statistical data visualization	0.11.2
pickle	Allows easy storage of many data types	4.0
transformers	Allows access to pretrained transformer models	4.19.2
tensorflow	Allows high performance numerical computation	2.8.0
gpt_2_simple	Allows easy access to GPT-2's functionalities	0.8.1
spacy	Allows access to Natural Language Processing techniques (e.g., lemmatization)	3.3.0
krippendorff	Allows fast computation of Krippendorff's alpha agreement measure	0.5.1
statsmodels	Allows fast computation of Fleiss' kappa agreement measure	0.13.2
pygsheets	Allows access to google spreadsheets	2.0.5
random	Allows access to random-number generators	3.8.13
sentence_transformers	Allows access to state-of-the-art sentence embeddings	2.2.0
sklearn-crfsuite	Allows access to a sklearn-compatible CRF classifier	0.3.6
tqdm	Allows access to progress bars and functions	4.64.0
math	Allows access to mathematical functions (e.g., ceil)	3.8.13
openai	Allows access to GPT-3	0.18.0
nlpcloud	Allows access to GPT-J	1.0.25

Appendix D

Conditional Random Field (CRF) experiment, using more ("full" - list described in Section 6.4) or fewer (without HasQuestion and HasExclamation) features. Table 4 presents the metric scores using the AL dataset, and Table 5 using the Twitter dataset.

Table 4: Comparison between the use of more or fewer features in the CRF models, using the AL dataset.

Altice Labs Dataset							
Representation	**Features**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Sparse	Full	0.91	0.75	0.72	0.95	0.73	0.94
Sparse	Fewer	0.92	0.8	0.7	0.96	0.75	0.94
Dense	Full	0.93	0.79	0.81	0.95	0.8	0.96
Dense	Fewer	0.93	0.79	0.82	0.95	0.8	0.96

Table 5: Comparison between the use of more or fewer features in the CRF models, using the Twitter dataset.

Twitter Dataset							
Representation	**Features**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Sparse	Full	0.8	0.74	0.68	0.86	0.71	0.86
Sparse	Fewer	0.81	0.79	0.63	0.91	0.7	0.85
Dense	Full	0.88	0.87	0.78	0.94	0.82	0.92
Dense	Fewer	0.89	0.89	0.81	0.94	0.85	0.91

CRF experiment, using different N-gram ranges (unigrams, bigrams, and trigrams). Table 6 presents the metric scores using the AL dataset, and Table 7 using the Twitter dataset.

Table 6: Comparison between the use of N-grams in the CRF models, using the AL dataset.

Altice Labs Dataset						
N-Grams	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Unigram	0.91	0.75	0.72	0.95	0.73	0.94
Bigram	0.9	0.81	0.57	0.97	0.67	0.9
Trigram	0.86	0.63	0.55	0.93	0.59	0.88

Table 7: Comparison between the use of N-grams in the CRF models, using the Twitter dataset.

Twitter Dataset						
N-Grams	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Unigram	0.8	0.74	0.68	0.86	0.71	0.86
Bigram	0.73	0.61	0.67	0.76	0.64	0.72
Trigram	0.71	0.6	0.58	0.78	0.59	0.7

Appendix E

Support Vector Machine (SVM) experiment, using each kernel function (linear, poly, rbf) with Term Frequency - Inverse Document Frequency (TF-IDF) encoding. Table 8 presents the metrics scores using the AL dataset, and Table 9 using the Twitter dataset.

Table 8: Comparison of the results obtained with each tested kernel function for SVMs with TF-IDF encoding, using the AL dataset.

Altice Labs Dataset						
Kernel	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Linear	0.91	0.79	0.68	0.96	0.73	0.9
Poly	0.91	0.83	0.63	0.97	0.72	0.88
RBF	0.91	0.84	0.66	0.97	0.74	0.9

Table 9: Comparison of the results obtained with each tested kernel function for SVMs with TF-IDF encoding, using the Twitter dataset.

Twitter Dataset						
Kernel	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Linear	0.82	0.9	0.76	0.93	0.82	0.91
Poly	0.82	0.76	0.85	0.8	0.8	0.89
RBF	0.86	0.85	0.81	0.89	0.83	0.93

SVM experiment, using each kernel function (linear, poly, rbf) with BERTimbau Natural Language Inference (NLI) encoding. Table 10 presents the metrics scores using the AL dataset, and Table 11 using the Twitter dataset.

Table 10: Comparison of the results obtained with each tested kernel function for SVMs with BERTimbau NLI encoding, using the AL dataset.

Twitter Dataset						
Kernel	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Linear	0.92	0.77	0.81	0.94	0.79	0.91
Poly	0.93	0.81	0.8	0.96	0.8	0.94
RBF	0.93	0.82	0.8	0.96	0.81	0.94

Table 11: Comparison of the results obtained with each tested kernel function for SVMs with BERTimbau NLI encoding, using the Twitter dataset.

Twitter Dataset						
Kernel	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Linear	0.91	0.94	0.84	0.96	0.89	0.96
Poly	0.91	0.94	0.84	0.96	0.89	0.96
RBF	0.91	0.93	0.86	0.95	0.89	0.97

SVM experiment, using each kernel function (linear, poly, rbf) and considering "context". Table 12 presents the metrics scores using the AL dataset, and Table 13 using the Twitter dataset.

Table 12: Comparison of the results obtained with each tested kernel function for SVMs, using "context" and the AL dataset.

Altice Labs Dataset							
Representation	**Kernel**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Sparse	*Linear*	0.89	0.82	0.46	0.98	0.59	0.9
Sparse	*Poly*	0.88	0.83	0.42	0.98	0.56	0.92
Sparse	*RBF*	0.89	0.82	0.49	0.98	0.61	0.92
Sparse	*Linear*	0.91	0.78	0.66	0.96	0.72	0.93
Sparse	*Poly*	0.9	0.9	0.51	0.99	0.65	0.92
Sparse	*RBF*	0.9	0.87	0.5	0.98	0.64	0.92

Table 13: Comparison of the results obtained with each tested kernel function for SVMs, using "context" and the Twitter dataset.

Twitter Dataset							
Representation	**Kernel**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Sparse	*Linear*	0.73	0.63	0.63	0.79	0.63	0.8
Sparse	*Poly*	0.64	0.5	0.68	0.61	0.58	0.76
Sparse	*RBF*	0.74	0.64	0.63	0.81	0.63	0.8
Sparse	*Linear*	0.77	0.68	0.71	0.81	0.69	0.84
Sparse	*Poly*	0.81	0.8	0.63	0.91	0.7	0.87
Sparse	*RBF*	0.81	0.81	0.62	0.92	0.7	0.86

Appendix F

Shallow classifiers' results for N-gram range tested (unigram, bigram, trigram), using TF-IDF encoding. Table 14 presents the metric scores using the AL dataset, and Table 15 using the Twitter dataset.

Table 14: Comparison of the results obtained with each tested N-gram range for shallow classifiers with sparse encoding, using the AL dataset.

Altice Labs Dataset							
Model	**N-Grams**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Logistic Regression	Unigrams	0.91	0.84	0.63	0.97	0.72	0.92
	Bigrams	0.89	0.84	0.49	0.98	0.62	0.88
	Trigrams	0.88	0.85	0.43	0.98	0.57	0.86
Linear SVM	Unigrams	0.91	0.79	0.68	0.96	0.73	0.9
	Bigrams	0.89	0.82	0.54	0.97	0.65	0.86
	Trigrams	0.88	0.85	0.45	0.98	0.59	0.77
Poly SVM	Unigrams	0.91	0.83	0.63	0.97	0.72	0.88
	Bigrams	0.89	0.82	0.53	0.97	0.64	0.81
	Trigrams	0.89	0.84	0.47	0.98	0.6	0.78
RBF SVM	Unigrams	0.91	0.84	0.66	0.97	0.74	0.9
	Bigrams	0.89	0.82	0.55	0.97	0.66	0.73
	Trigrams	0.89	0.84	0.47	0.98	0.6	0.76
Naive Bayes	Unigrams	0.91	0.82	0.64	0.97	0.72	0.92
	Bigrams	0.88	0.79	0.51	0.97	0.62	0.88
	Trigrams	0.88	0.79	0.47	0.97	0.59	0.85
Random Forest	Unigrams	0.91	0.78	0.7	0.96	0.74	0.92
	Bigrams	0.9	0.81	0.56	0.97	0.66	0.88
	Trigrams	0.89	0.83	0.49	0.98	0.62	0.86

Shallow classifiers' results for N-gram range tested (unigram, bigram, trigram), using TF-IDF encoding and considering "context". Table 16 presents the metric scores using the

Table 15: Comparison of the results obtained with each tested N-gram range for shallow classifiers with sparse encoding, using the Twitter dataset.

Twitter Dataset							
Model	**N-Grams**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Logistic Regression	Unigrams	0.84	0.89	0.73	0.93	0.8	0.94
	Bigrams	0.71	0.91	0.35	0.98	0.51	0.88
	Trigrams	0.62	0.81	0.14	0.98	0.24	0.82
Linear SVM	Unigrams	0.86	0.9	0.76	0.93	0.82	0.91
	Bigrams	0.81	0.75	0.84	0.79	0.79	0.88
	Trigrams	0.78	0.66	0.98	0.62	0.62	0.82
Poly SVM	Unigrams	0.82	0.76	0.85	0.8	0.8	0.89
	Bigrams	0.8	0.74	0.8	0.79	0.79	0.86
	Trigrams	0.61	0.72	0.14	0.96	0.96	0.81
RBF SVM	Unigrams	0.86	0.85	0.81	0.89	0.89	0.93
	Bigrams	0.7	0.8	0.41	0.92	0.92	0.59
	Trigrams	0.59	0.62	0.14	0.93	0.93	0.34
Naive Bayes	Unigrams	0.82	0.88	0.67	0.93	0.93	0.94
	Bigrams	0.7	0.89	0.34	0.97	0.97	0.88
	Trigrams	0.62	0.81	0.14	0.98	0.98	0.82
Random Forest	Unigrams	0.83	0.86	0.73	0.91	0.91	0.9
	Bigrams	0.64	0.94	0.19	0.99	0.99	0.61
	Trigrams	0.62	0.92	0.13	0.99	0.99	0.82

AL dataset, and Table 17 using the Twitter dataset.

Table 16: Comparison of the results obtained with each tested N-gram range for shallow classifiers with sparse encoding, using the AL dataset and considering "context".

Altice Labs Dataset							
Model	**N-Grams**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Logistic Regression	Unigrams	0.88	0.79	0.41	0.98	0.54	0.9
	Bigrams	0.88	0.8	0.4	0.98	0.53	0.91
	Trigrams	0.89	0.77	0.57	0.96	0.66	0.93
Linear SVM	Unigrams	0.89	0.82	0.46	0.98	0.59	0.9
	Bigrams	0.9	0.82	0.59	0.97	0.69	0.92
	Trigrams	0.91	0.78	0.66	0.86	0.72	0.93
Poly SVM	Unigrams	0.88	0.83	0.42	0.98	0.56	0.92
	Bigrams	0.88	0.82	0.39	0.98	0.53	0.91
	Trigrams	0.9	0.9	0.51	0.99	0.65	0.92
RBF SVM	Unigrams	0.89	0.82	0.49	0.98	0.61	0.93
	Bigrams	0.89	0.83	0.48	0.98	0.61	0.93
	Trigrams	0.9	0.87	0.5	0.98	0.64	0.92
Naive Bayes	Unigrams	0.85	0.72	0.29	0.98	0.41	0.83
	Bigrams	0.88	0.77	0.45	0.97	0.57	0.84
	Trigrams	0.84	0.55	0.44	0.92	0.49	0.83
Random Forest	Unigrams	0.9	0.84	0.51	0.98	0.63	0.92
	Bigrams	0.89	0.8	0.53	0.97	0.64	0.92
	Trigrams	0.9	0.88	0.5	0.99	0.64	0.93

Table 17: Comparison of the results obtained with each tested N-gram range for shallow classifiers with sparse encoding, using the Twitter dataset and considering "context".

Twitter Dataset							
Model	**N-Grams**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Logistic Regression	Unigrams	0.76	0.67	0.64	0.82	0.65	0.81
	Bigrams	0.76	0.7	0.55	0.87	0.62	0.78
	Trigrams	0.81	0.75	0.71	0.87	0.73	0.88
Linear SVM	Unigrams	073	0.63	0.63	0.79	0.63	0.8
	Bigrams	0.75	0.68	0.58	0.85	0.63	0.79
	Trigrams	0.77	0.68	0.71	0.81	0.69	0.84
Poly SVM	Unigrams	0.64	0.5	0.68	0.61	0.58	0.76
	Bigrams	0.63	0.49	0.68	0.6	0.57	0.7
	Trigrams	0.81	0.8	0.63	0.91	0.7	0.87
RBF SVM	Unigrams	0.74	0.64	0.63	0.81	0.63	0.8
	Bigrams	0.74	0.65	0.59	0.82	0.62	0.77
	Trigrams	0.81	0.81	0.62	0.92	0.7	0.86
Naive Bayes	Unigrams	0.73	0.66	0.5	0.86	0.57	0.76
	Bigrams	0.76	0.77	0.46	0.92	0.58	0.73
	Trigrams	0.72	0.55	0.71	0.73	0.64	0.8
Random Forest	Unigrams	0.76	0.64	0.73	0.77	0.68	0.86
	Bigrams	0.72	0.64	0.54	0.83	0.59	0.81
	Trigrams	0.78	0.69	0.69	0.83	0.69	0.83

Appendix G

Zero-Shot Learning (Zero-SL) results, using different languages (Portuguese vs. English) and different hypotheses settings (as labels vs. as sentences). Table 18 presents the metrics scores using the AL dataset, and Table 19 using the Twitter dataset.

Table 18: Comparison of the results obtained with different languages and sentence vs. label hypotheses in the Zero-Shot Learning model, using the AL dataset

Altice Labs Dataset							
Language	**Hypothesis**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Portuguese	Label	0.8	0.46	0.76	0.81	0.57	0.85
	Sentence	0.33	0.14	0.52	0.29	0.22	0.52
English	Label	0.49	0.25	0.92	0.4	0.39	0.76
	Sentence	0.78	0.35	0.26	0.9	0.3	0.64

Table 19: Comparison of the results obtained with different languages and sentence vs. label hypotheses in the Zero-Shot Learning model, using the Twitter dataset

Twitter Dataset							
Language	**Hypothesis**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Portuguese	Label	0.58	0.44	0.63	0.55	0.52	0.62
	Sentence	0.51	0.38	0.59	0.46	0.46	0.58
English	Label	0.57	0.45	0.81	0.44	0.44	0.59
	Sentence	0.59	0.37	0.22	0.79	0.79	0.5

Zero-SL results, using different languages (Portuguese vs. English) and different hypotheses settings (as labels vs. as sentences), while considering "context". Table 20 presents the metrics scores using the AL dataset, and Table 21 using the Twitter dataset.

Table 20: Comparison of the results obtained with different languages and sentence vs. label hypotheses in the Zero-Shot Learning model, using "context" and the AL dataset.

Altice Labs Dataset							
Language	**Hypothesis**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Portuguese	Label	0.8	0.45	0.58	0.85	0.51	0.74
	Sentence	0.26	0.14	0.64	0.18	0.23	0.49
English	Label	0.43	0.21	0.83	0.34	0.34	0.63
	Sentence	0.65	0.22	0.39	0.7	0.28	0.58

Table 21: Comparison of the results obtained with different languages and sentence vs. label hypotheses in the Zero-Shot Learning model, using "context" and the Twitter dataset.

Twitter Dataset							
Language	**Hypothesis**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Portuguese	Label	0.56	0.43	0.65	0.51	0.52	0.57
	Sentence	0.46	0.32	0.46	0.45	0.38	0.49
English	Label	0.56	0.43	0.65	0.51	0.52	0.57
	Sentence	0.56	0.36	0.29	0.71	0.32	0.48

Appendix H

Bidirectional Encoder Representations from Transformers (BERT)-CRF's results using different representation functions (ways to retrieve the sentence's embedding) using the Twitter dataset (Table 22):

Table 22: Comparison of the results obtained with different representation functions in the BERT-CRF model, using the Twitter dataset.

Twitter Dataset						
Representation Function	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Concatenation	0.84	0.72	0.82	0.85	0.77	0.91
Mean	0.86	0.77	0.82	0.88	0.79	0.91
Last	0.86	0.74	0.85	0.87	0.79	0.91
First	0.57	0.46	0.41	0.67	0.43	0.60

Appendix I

Results for each Machine Learning (ML) classifier analysed and their tuned version. The bold values mean that, for that model, we had to use the hyperparameters found for the Twitter dataset and apply them to the AL dataset. Table 23 presents the metrics scores using the AL dataset, and Table 24 using the Twitter dataset.

Table 23: Comparison between the performances of the tuned and original classifiers, using the AL dataset.

Altice Labs Dataset								
Representation	**Model**	**Version**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Sparse	*Logistic Regression*	Original	0.91	0.84	0.63	0.97	0.72	0.92
		Tuned	0.92	0.81	0.73	0.96	0.77	0.94
Dense	*Logistic Regression*	Original	0.93	0.8	0.83	0.95	0.81	0.94
		Tuned	0.93	0.79	0.82	0.95	0.8	0.94
Sparse	*Linear SVM*	Original	0.91	0.79	0.68	0.96	0.73	0.9
		Tuned	0.92	0.81	0.71	0.96	0.76	0.92
Dense	*Linear SVM*	Original	0.92	0.77	0.81	0.94	0.79	0.91
		Tuned	**0.93**	**0.81**	**0.82**	**0.96**	**0.81**	**0.93**
Sparse	*Poly SVM*	Original	0.91	0.83	0.63	0.97	0.72	0.88
		Tuned	**0.91**	**0.8**	**0.65**	**0.96**	**0.72**	**0.89**
Dense	*Poly SVM*	Original	0.93	0.81	0.8	0.96	0.8	0.94
		Tuned	**0.93**	**0.82**	**0.8**	**0.96**	**0.81**	**0.94**
Sparse	*RBF SVM*	Original	0.91	0.84	0.66	0.97	0.74	0.9
		Tuned	**0.91**	**0.77**	**0.73**	**0.95**	**0.75**	**0.89**
Dense	*RBF SVM*	Original	0.93	0.82	0.8	0.96	0.81	0.94
		Tuned	**0.93**	**0.81**	**0.81**	**0.96**	**0.81**	**0.95**
Sparse	*Naive Bayes*	Original	0.91	0.82	0.64	0.97	0.72	0.92
		Tuned	0.91	0.81	0.65	0.97	0.72	0.92
Dense	*Naive Bayes*	Original	0.9	0.76	0.69	0.95	0.72	0.9
		Tuned	0.9	0.74	0.69	0.94	0.71	0.9
Sparse	*Random Forest*	Original	0.91	0.78	0.7	0.96	0.74	0.92
		Tuned	0.91	0.82	0.68	0.97	0.74	0.92
Dense	*Random Forest*	Original	0.92	0.82	0.76	0.96	0.79	0.95
		Tuned	0.93	0.84	0.77	0.97	0.8	0.96
Dense (BERTimbau)	*Finetuned BERTimbau*	Original	0.95	0.89	0.85	0.97	0.87	0.97
		Tuned	**0.94**	**0.92**	**0.78**	**0.98**	**0.84**	**0.96**
Sparse	*CRF*	Original	0.91	0.75	0.72	0.95	0.73	0.94
		Tuned	0.91	0.75	0.77	0.94	0.76	0.95
Dense	*CRF*	Original	0.93	0.79	0.81	0.95	0.8	0.96
		Tuned	0.93	0.79	0.8	0.96	0.79	0.96
Dense (BERTimbau)	*BERT-CRF*	Original	0.92	0.74	0.78	0.94	0.76	0.93
		Tuned	**0.92**	**0.71**	**0.81**	**0.94**	**0.76**	**0.94**

Table 24: Comparison between the performances of the tuned and original classifiers, using the Twitter dataset.

Twitter Dataset								
Representation	**Model**	**Version**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Sparse	*Logistic Regression*	Original	0.84	0.89	0.73	0.93	0.8	0.94
		Tuned	0.88	0.92	0.79	0.95	0.85	0.94
Dense	*Logistic Regression*	Original	0.92	0.91	0.9	0.93	0.9	0.96
		Tuned	0.92	0.92	0.9	0.94	0.91	0.97
Sparse	*Linear SVM*	Original	0.86	0.9	0.76	0.93	0.82	0.91
		Tuned	0.86	0.88	0.78	0.92	0.83	0.92
Dense	*Linear SVM*	Original	0.91	0.88	0.91	0.91	0.89	0.95
		Tuned	0.92	0.9	0.92	0.92	0.91	0.97
Sparse	*Poly SVM*	Original	0.82	0.76	0.85	0.8	0.8	0.89
		Tuned	0.86	0.85	0.82	0.89	0.83	0.92
Dense	*Poly SVM*	Original	0.91	0.94	0.84	0.96	0.89	0.96
		Tuned	0.91	0.94	0.85	0.96	0.89	0.97
Sparse	*RBF SVM*	Original	0.86	0.85	0.81	0.89	0.83	0.93
		Tuned	0.85	0.88	0.75	0.92	0.81	0.91
Dense	*RBF SVM*	Original	0.91	0.93	0.86	0.95	0.89	0.97
		Tuned	0.92	0.91	0.9	0.93	0.9	0.97
Sparse	*Naive Bayes*	Original	0.82	0.88	0.67	0.93	0.76	0.94
		Tuned	0.84	0.88	0.74	0.92	0.8	0.94
Dense	*Naive Bayes*	Original	0.86	0.84	0.84	0.88	0.84	0.92
		Tuned	0.86	0.84	0.84	0.88	0.84	0.92
Sparse	*Random Forest*	Original	0.83	0.86	0.73	0.91	0.79	0.9
		Tuned	0.86	0.93	0.74	0.96	0.82	0.94
Dense	*Random Forest*	Original	0.88	0.88	0.84	0.92	0.86	0.96
		Tuned	0.89	0.89	0.85	0.92	0.87	0.95
Dense (BERTimbau)	*Finetuned BERTimbau*	Original	0.93	0.97	0.88	0.97	0.92	1
		Tuned	0.92	0.91	0.9	0.93	0.9	0.97
Sparse	*CRF*	Original	0.8	0.74	0.68	0.86	0.71	0.86
		Tuned	0.83	0.76	0.78	0.86	0.77	0.91
Dense	*CRF*	Original	0.88	0.87	0.78	0.94	0.82	0.92
		Tuned	0.88	0.88	0.78	0.94	0.83	0.92
Dense (BERTimbau)	*BERT-CRF*	Original	0.86	0.74	0.85	0.87	0.79	0.91
		Tuned	0.86	0.78	0.81	0.88	0.79	0.91

Results for each ML classifier analysed and their tuned version, considering context. The bold values mean that, for that model, we had to use the hyperparameters found for the Twitter dataset and apply them to the AL dataset. Table 25 presents the metrics scores using the AL dataset, and Table 26 using the Twitter dataset.

Table 25: Comparison between the performances of the tuned and original classifiers, using "context" and the AL dataset.

Altice Labs Dataset								
Representation	**Model**	**Version**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Sparse	*Logistic*	Original	0.88	0.79	0.41	0.98	0.54	0.9
	Regression	Tuned	0.89	0.8	0.51	0.97	0.62	0.91
Dense	*Logistic*	Original	0.89	0.77	0.57	0.96	0.66	0.93
	Regression	Tuned	0.91	0.8	0.64	0.97	0.71	0.94
Sparse	*Linear*	Original	0.89	0.82	0.46	0.98	0.59	0.9
	SVM	Tuned	**0.89**	**0.82**	**0.46**	**0.98**	**0.59**	**0.9**
Dense	*Linear*	Original	0.91	0.78	0.66	0.96	0.72	0.93
	SVM	Tuned	**0.91**	**0.78**	**0.66**	**0.96**	**0.72**	**0.93**
Sparse	*Poly*	Original	0.88	0.83	0.42	0.98	0.56	0.92
	SVM	Tuned	**0.88**	**0.83**	**0.42**	**0.98**	**0.56**	**0.92**
Dense	*Poly*	Original	0.9	0.9	0.51	0.99	0.65	0.92
	SVM	Tuned	**0.9**	**0.9**	**0.51**	**0.99**	**0.65**	**0.92**
Sparse	*RBF*	Original	0.89	0.82	0.49	0.98	0.61	0.93
	SVM	Tuned	**0.89**	**0.82**	**0.49**	**0.98**	**0.61**	**0.93**
Dense	*RBF*	Original	0.9	0.87	0.5	0.98	0.64	0.92
	SVM	Tuned	**0.9**	**0.87**	**0.5**	**0.98**	**0.64**	**0.92**
Sparse	*Naive*	Original	0.85	0.72	0.29	0.98	0.41	0.83
	Bayes	Tuned	0.86	0.71	0.31	0.97	0.43	0.83
Dense	*Naive*	Original	0.84	0.55	0.44	0.92	0.49	0.83
	Bayes	Tuned	0.84	0.55	0.43	0.92	0.48	0.82
Sparse	*Random*	Original	0.9	0.84	0.51	0.98	0.63	0.92
	Forest	Tuned	0.91	0.86	0.57	0.98	0.69	0.91
Dense	*Random*	Original	0.9	0.88	0.5	0.99	0.64	0.93
	Forest	Tuned	0.9	0.87	0.52	0.98	0.65	0.93

Table 26: Comparison between the performances of the tuned and original classifiers, using "context" and the Twitter dataset.

Twitter Dataset								
Representation	**Model**	**Version**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Sparse	*Logistic*	Original	0.76	0.67	0.64	0.82	0.65	0.81
	Regression	Tuned	0.78	0.72	0.64	0.86	0.68	0.82
Dense	*Logistic*	Original	0.81	0.75	0.71	0.87	0.73	0.88
	Regression	Tuned	0.85	0.84	0.73	0.92	0.78	0.89
Sparse	*Linear*	Original	0.73	0.63	0.63	0.79	0.63	0.8
	SVM	Tuned	0.75	0.65	0.63	0.81	0.64	0.82
Dense	*Linear*	Original	0.77	0.68	0.71	0.81	0.69	0.84
	SVM	Tuned	0.82	0.78	0.69	0.89	0.73	0.89
Sparse	*Poly*	Original	0.64	0.5	0.68	0.61	0.58	0.76
	SVM	Tuned	0.71	0.58	0.68	0.72	0.63	0.79
Dense	*Poly*	Original	0.81	0.8	0.63	0.91	0.7	0.87
	SVM	Tuned	0.83	0.79	0.72	0.89	0.75	0.9
Sparse	*RBF*	Original	0.74	0.64	0.63	0.81	0.63	0.8
	SVM	Tuned	0.77	0.75	0.54	0.9	0.63	0.8
Dense	*RBF*	Original	0.81	0.81	0.62	0.92	0.7	0.86
	SVM	Tuned	0.84	0.82	0.72	0.91	0.77	0.89
Sparse	*Naive*	Original	0.73	0.66	0.5	0.86	0.57	0.76
	Bayes	Tuned	0.75	0.7	0.51	0.88	0.59	0.78
Dense	*Naive*	Original	0.72	0.59	0.71	0.73	0.64	0.8
	Bayes	Tuned	0.73	0.6	0.71	0.74	0.65	0.8
Sparse	*Random*	Original	0.76	0.64	0.73	0.77	0.68	0.86
	Forest	Tuned	0.77	0.7	0.65	0.84	0.67	0.86
Dense	*Random*	Original	0.78	0.69	0.69	0.83	0.69	0.83
	Forest	Tuned	0.8	0.75	0.65	0.88	0.7	0.85

Appendix J

Results of the Finetuned BERTimbau tuning process for the Twitter dataset. The set of hyperparameters and folds that we considered the best for each experience is presented in bold (Table 27).

Table 27: Comparison of the results obtained using Cross-Validation and Grid Search Tuning for the Finetuned BERTimbau model.

Twitter Dataset									
Learning Rate	**Epsilon**	**Epochs**	**Fold**	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
0.0001	0.00000001	4	0	0.86	0.93	0.79	0.94	0.85	0.99
			1	**0.89**	**0.84**	**0.9**	**0.88**	**0.87**	**0.96**
			2	0.89	0.91	0.85	0.93	0.88	0.95
			3	0.89	0.82	0.9	0.88	0.86	0.95
			4	0.9	0.88	0.89	0.91	0.88	0.96
0.0001	0.00000001	2	0	0.87	0.85	0.85	0.88	0.85	0.99
			1	0.9	0.9	0.86	0.92	0.88	0.99
			2	**0.92**	**0.91**	**0.9**	**0.93**	**0.9**	**0.97**
			3	0.63	0.15	0.93	0.61	0.26	0.93
			4	0.86	0.7	0.96	0.81	0.81	0.96
0.0001	0.001	2	0	0.86	0.73	0.93	0.82	0.82	0.99
			1	0.83	0.67	0.92	0.79	0.78	0.97
			2	**0.89**	**0.8**	**0.94**	**0.86**	**0.86**	**0.99**
			3	0.87	0.76	0.93	0.84	0.84	0.97
			4	0.83	0.68	0.9	0.8	0.77	0.99
0.0001	0.001	4	0	0.9	0.9	0.87	0.92	0.88	0.99
			1	0.91	0.86	0.93	0.9	0.89	0.97
			2	0.89	0.85	0.9	0.89	0.87	0.96
			3	0.89	0.9	0.85	0.92	0.87	0.96
			4	**0.9**	**0.87**	**0.9**	**0.9**	**0.88**	**0.96**
0.00005	0.001	2	0	**0.88**	**0.76**	**0.95**	**0.84**	**0.84**	**0.99**
			1	0.85	0.74	0.91	0.82	0.82	0.95
			2	0.87	0.77	0.92	0.84	0.84	0.95
			3	0.85	0.68	0.95	0.8	0.79	0.95
			4	0.87	0.76	0.92	0.84	0.83	0.99
0.00005	0.001	4	0	0.91	0.86	0.92	0.9	0.89	0.99
			1	0.89	0.82	0.9	0.88	0.86	0.96
			2	0.91	0.91	0.88	0.93	0.89	0.99
			3	0.89	0.9	0.85	0.92	0.87	0.97
			4	**0.88**	**0.76**	**0.96**	**0.84**	**0.85**	**0.99**
0.00005	0.00000001	2	0	**0.89**	**0.79**	**0.95**	**0.86**	**0.86**	**0.96**
			1	0.85	0.69	0.95	0.81	0.8	0.99
			2	0.88	0.78	0.92	0.85	0.84	0.96
			3	0.81	0.58	0.95	0.75	0.72	0.99
			4	0.83	0.62	0.98	0.77	0.76	0.99
0.00005	0.00000001	4	0	0.9	0.95	0.84	0.95	0.89	0.96
			1	0.89	0.9	0.85	0.92	0.87	0.97
			2	0.89	0.92	0.84	0.94	0.88	0.96
			3	0.87	0.84	0.85	0.88	0.84	0.99
			4	**0.9**	**0.8**	**0.95**	**0.87**	**0.87**	**0.99**

Appendix K

Full metric scores of the experiments regarding dialogue classification when the models are not trained in this type of data. Tables 28, 29, and 30 present the evaluation of the shallow learning models with sparse representations when tested on the AL dataset, the Twitter dataset, and the Sentituites dataset, respectively.

Table 28: Evaluation of the classifiers trained on the Sentituites data and tested on the AL data.

Altice Labs Dataset						
Model	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Logistic Regression	0.7	0.08	0.07	0.84	0.07	0.43
RBF SVM	0.8	0.19	0.03	0.97	0.05	0.43
Naive Bayes	0.81	0.25	0.03	0.98	0.05	0.45
Random Forest	0.81	0.24	0.02	0.99	0.04	0.56

Table 29: Evaluation of the classifiers trained on the Sentituites data and tested on the Twitter data.

Twitter Dataset						
Model	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Logistic Regression	0.59	0.59	0.18	0.91	0.28	0.53
RBF SVM	0.6	0.73	0.12	0.97	0.21	0.54
Naive Bayes	0.59	0.7	0.08	0.98	0.14	0.53
Random Forest	0.56	0.48	0.11	0.91	0.18	0.58

Table 30: Evaluation of the classifiers trained and tested on the Sentituites data.

Sentituites Dataset						
Model	**Accuracy**	**Precision**	**Recall**	**Specificity**	**F1 Score**	**A.U.C.**
Logistic Regression	0.79	0.9	0.39	0.98	0.54	0.83
RBF SVM	0.82	0.93	0.47	0.98	0.62	0.88
Naive Bayes	0.75	0.94	0.22	0.99	0.36	0.83
Random Forest	0.8	0.91	0.43	0.98	0.58	0.85