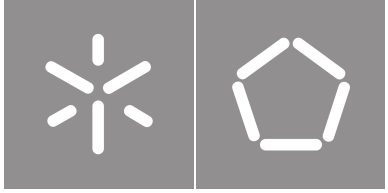
Universidade do Minho
Escola de Engenharia

António Manuel Almeida Pinto Bandeira Santos

Natural Language Processing Applied to Human Resources

October, 2023



Universidade do Minho
Escola de Engenharia

António Manuel Almeida Pinto Bandeira Santos

Natural Language Processing Applied to Human Resources

Master Thesis
Master in Informatics Engineering

Work developed under the supervision of:
José João Antunes Guimarães Dias Almeida
Luís Filipe Costa Cunha

October, 2023

Acknowledgements

The conclusion of this dissertation marks the end of a stage in my life that spanned the last 18 years. This stage, and everyone involved in it, has profoundly shaped my life, and I wouldn't have had it any other way.

Firstly, I would like to express my gratitude to Daniel Oliveira and everyone at Konkconsulting for providing the means and support for the completion of this document. I would also like to thank Prof. José João Almeida and Filipe Cunha for supervising and assisting in making this thesis the best it could be. Special thanks go to my godfather, Jorge Simões, for generously offering his time, insight, and expertise to ensure the proper refinement of my thesis. Additionally, I'm grateful to ChatGPT for rectifying the broken English present in this document.

I want to extend my appreciation to my entire family for supporting me throughout this journey, even when I diverged from the path and became the first person in two generations of my family not to pursue medicine. I'd like to offer a special thanks to my parents, grandparents, and my sister for playing a pivotal role in shaping the person I am today and ensuring my success in life.

I would also like to acknowledge all my friends and close people in my life, 'À equipa,' for providing me with great times and also keeping me on track during moments of heavy procrastination. To Né and Sara, who stood by my side throughout the last academic years, making this journey much brighter. Additionally, I would like to express my gratitude to Sara for being the person she is and being my mental safe haven this last year, always being there for me whenever I needed support, truly a beacon of happiness in my life.

Last but not least, I want to thank my furry little friend, Jagu, for simply existing and being the best cat any owner could dream of.

Every single one of you who had an impact in my life indirectly had a part in the inception of this dissertation and for that i would like to thank you all. **Thank you!**

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the Universidade do Minho.

Universidade do Minho, Braga, October 2023

António Manuel Almeida Pinto Bandeira Santos

Abstract

Natural Language Processing Applied to Human Resources

As technology continues to advance, many companies are seeking ways to integrate Artificial Intelligence (AI) into their operations in order to optimize their workflow and improve efficiency. One area that could greatly benefit from AI solutions is the Human Resources (HR) department. This dissertation explores the use of AI and Natural Language Processing (NLP) to extract emotions from text-based communications, such as emails and instant messages between HR representatives and employees. By providing insight into the states of mind being expressed in text, this technology has the potential to improve communication and understanding between the two parties, which in turn may lead to better conflict resolution, increased employee engagement, and improved productivity. The study will examine the use of various machine learning algorithms ranging from Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) to current state-of-the-art architectures such as transformers like BERT and XLNet to extract better insights from text and to identify the emotions contained in messages. The results of this thesis contribute to the development of an AI solution capable of identifying one out of seven emotions in both general and specialized conversations with an accuracy of 75.38%, having the capacity of enhancing the efficiency of HR departments by facilitating communication and understanding with employees.

Keywords: natural language processing (nlp), emotion recognition, learning, neural network, human resources, employee retention

Resumo

Processamento de Linguagem Natural Applicado a Recursos Humanos

À medida que a tecnologia avança, muitas empresas procuram formas de integrar a Inteligência Artificial (IA) nas suas operações, a fim de optimizar o seu fluxo de trabalho e melhorar a eficiência. Uma área que pode beneficiar da Inteligência Artificial é o departamento de Recursos Humanos (RH). Este artigo explora o uso de IA e Processamento de Linguagem Natural (PNL) para extrair emoções em comunicações baseadas em texto, tais como e-mails e mensagens entre representantes de RH e empregados. Ao fornecer informações sobre o estado de espírito expresso no texto, esta tecnologia tem o potencial de melhorar a comunicação e a compreensão entre as duas partes, o que por sua vez pode levar a uma melhor resolução de conflitos, a um maior envolvimento dos funcionários e a uma maior produtividade. O estudo examinará a utilização de vários algoritmos de aprendizagem de máquinas, tais como Convolutional Neural Network (CNN) e Recurrent Neural Network (RNN) para extrair melhores percepções do texto e para identificar as emoções contidas nas mensagens. Os resultados desta tese contribuem para o desenvolvimento de uma solução de IA capaz de identificar uma de entre sete possíveis emoções em conversas de contexto profissional tal como conversas gerais com uma precisão de 75.38%, tendo a capacidade de melhorar o fluxo de trabalho dos departamentos de RH, facilitando a comunicação e a compreensão com os funcionários.

Palavras-chave: processamento de linguagem natural (pln), extração de emoções, aprendizagem profunda, redes neuronais, recursos humanos, retenção de colaboradores

Contents

List of Figures

List of Tables

List of Listings

Acronyms

AI Artificial Intelligence ix, 2
API Application Programming Interface 3

BERT Bidirecional Encoder Representations from Transformers ix, 18
BiGRU Bidirectional Gated Recurrent Unit 17
BiLSTM Bidirectional Long-Short Term Memory 16

CBET Cleaned Balanced Emotional Tweets 13
CNN Convolutional Neural Network ix, 16

DCNN Deep Convolutional Neural Network 13
distilBERT Distilled BERT 18, 19

EARL Emotion Annotation and Representation Language 8
EWE Emotion Word Embeddings 16

GloVe Global Vectors for Word Representation 14, 16
GPT Generative Pre-trained Transformer 2
GRU Gated Recurrent Unit 16

HR Human Resources ix, 1, 2, 10, 17

IEMOCAP Interactive Emotional Dyadic Motion Capture 10
ISEAR International Survey on Emotion Antecedents and Reactions 11, 18

LSTM Long Short-Term Memory 13

MELD Multimodal EmotionLines Dataset 10

NLP Natural Language Processing ix, 2

RNN Recurrent Neural Network ix, 13

RoBERTa Robustly Optimized BERT Pretraining Approach 18, 19

SEER Semantic Emoticon Emotion Recognition 17

SENN Semantic-Emotion Neural Network 16

SVM Support Vector Machine xv, 12, 16, 17

Chapter 1

Introduction

1.1 Context

Currently, having employees experienced in a certain domain is essential for any company to thrive in the competitive business world. As a result, companies have to be at the top of their game to keep their workforce happy in order to assure the retention and evolution of the workers, which in turn can provide better work efficiency and growth of the organization. These concepts, known as talent management and employee retention, are key concepts for the human resources department, and it is its role to handle those correctly.

According to the U.S. Bureau of Labor Statistics [1] an employee stays on average 4.1 years at a given company. This value reflects the difficulty companies currently have in retaining its workforce, which in turn drastically impacts the company's efficiency and its ability to grow, mainly due to dissatisfaction of one of the parties.

In addition, employee retention is crucial for a company's economic growth. Research indicates that the cost of replacing an employee is roughly one third of their annual salary [2]. This includes the decreased productivity of the workforce, as well as the expenses of recruiting and training a replacement. As time passes, the cost of employee turnover is steadily increasing. In 2021, this cost had reached over 700 billion dollars in the United States alone [3].

One of the main reasons for employee departure is dissatisfaction with their job. In 2021, 8% of all employment contract terminations were related to dissatisfaction with management [3]. This can be caused by poor communication between management and the workforce. A satisfied employee is more productive and 87% less likely to leave their job compared to an unsatisfied one, according to studies [4].

The domain of Human Resources (HR) is an area that relies heavily on communication between a company and its employees to ensure that both parties are satisfied. This means that it is essential

for both sides of the exchange to be understood correctly, to not extract inaccurate meanings that could damage the future relationship between the employee and the company. Since humans are prone to errors and there is a large volume of information exchanged between the employees during conversations, it is not uncommon for the intent of one of the speakers to be misunderstood by the other. So, a tool capable of aiding an employee to diagnose a message and analyze the meaning behind it could be useful to ensure a smooth resolution of any exchange.

1.2 The Problem

Dealing with humans is hard, every single human being is unique and needs to be treated differently if one wants to have a healthy environment in the workplace. This can be a draining task for the ones in charge of it, especially if they don't have any tool that guides them on the best practices to apply.

A message in its purest form can be interpreted as a textual representation of one's state of mind. As with all things related to the human psychology, it is very complex and full of nuances, so retrieving the emotion or sentiment behind a piece of text is not an easy task. However, with the evolution of machine learning, Natural Language Processing (NLP) and the ever-increasing amount of data available online, new methods of dissecting messages to study its underlying meaning are surfacing, all with varying degrees of success for its intended use case.

With this evolution, the NLP market has been growing at a staggering rate [5]. With many companies being created with this area as the main focus, some like Censia, HoneIt and Ushur aim the potential of the technology directly to the Human Resources domain [6]. Future Market Insights inc. [7] expects the NLP market to reach 45 Billion U.S dollars in the next 10 years.

Sentiment analysis and emotion recognition are not cutting edge problems to tackle. As said before, over the years several solutions for this problem have surfaced with varying degrees of success. Companies like inVibe [8] use NLP to unravel the true potential of sentiment analysis hidden inside text and some colossal Artificial Intelligence (AI)s like OpenAI's Generative Pre-trained Transformer (GPT)3 [9] are also capable of extracting emotion from prompts, however most solutions are often too generalized or simply not viable to apply in the current context. In order to achieve a better solution than those currently available, the problem's scope should be reduced to enclose only the proposed environment of conversations between HR and the workers.

1.3 The goal and contributions

The main objective of this study is to investigate current Natural Language Processing state of the art and general best practices, requirements, and constraints and apply it to a problem related to the Human Resources domain. In this case, it will focus on the development of a tool capable of analyzing message exchanges between employees and the HR department using NLP techniques for sentiment analysis or,

more precisely, emotion recognition, which will aid them in keeping a good workplace environment and help them make decisions regarding talent management.

The output of this study is not meant to be a full-fledged product ready for deployment, it is to be a proof of concept that, can be showcased in KonkConsulting portfolio if needed. As a result, the research will mainly focus on a balance between innovation and efficiency of the technologies applied and not on the solution itself.

1.4 Document Structure

The structure of this document is divided by the following main sections:

- Introduction
- State-of-the-art
- Early Work
- Solution Implementation
- Conclusion

The introduction provides a comprehensive overview, covering the contextual background and motivation behind the study. It also outlines the objectives and contributions aimed for by the proposed solution. Toward the end of the introduction, there is a preliminary outline of the study's development plan.

The State-of-the-art section delves into a literature review of the articles employed in the research. This section is divided into two subsections. In the first subsection, the methodologies governing the selection process of the articles incorporated into the literature review are presented and explained. In the second subsection, each of the articles is described in detail.

The Early Work section involves an exhaustive exploration of available data and tools. Additionally, it encompasses an investigation of potential data manipulations required to optimize the training process.

The subsequent chapter provides a comprehensive illustration of the entire implementation process. This includes the comparison of datasets, optimal preprocessing steps, and the detailed study conducted to obtain the best model, along with the implementation of the Application Programming Interface (API). Each step is explained and visually represented.

In the conclusion, a brief overview of the whole study is done. The limitations and possible future work are outlined and the accomplished goals are enumerated.

1.4.1 Work Plan

The development of this dissertation is planned to go from October 2022 until October 2023. All communication between the student and his coordinators were done mainly through Microsoft Teams and

Emails, with the occasional in-person meeting either at the KonkConsulting office or at the Department of Informatics at the University of Minho.

A rough enumeration of the development plan goes as follows:

1. Review and analyze the current state of the art to retrieve a conclusion about the best methods and practices to apply.
2. Collect datasets that might contain useful information for the development of the solution.
3. Apply pre-processing methods on said datasets to remove all the unnecessary information contained and in turn increase the solution efficiency
4. Train and benchmark the current mainstream Machine Learning models on the processed datasets to conclude the best method

It can also be seen in the following Gantt diagram:

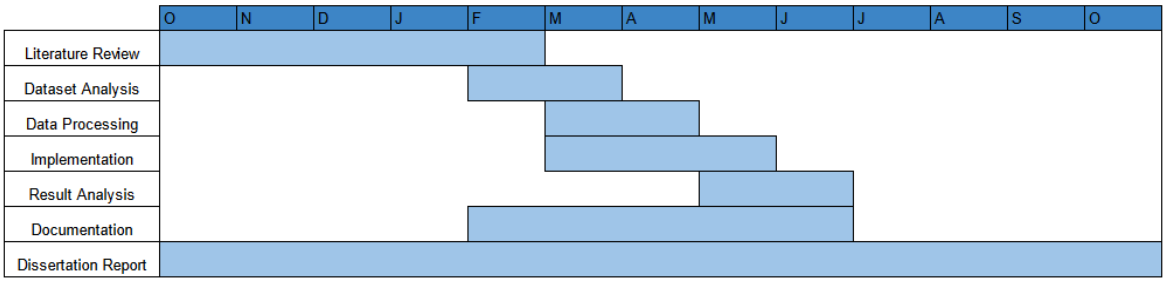


Figure 1: Work plan

Chapter 2

State of the Art

As the old saying goes there's no need to "reinvent the wheel", if the scientific community wants to truly progress at the best rate possible it should use the knowledge of past works as a base for newer ones. That way it can be assured that no time has been wasted gathering conclusions present in older documents. To achieve this, one must make a careful selection of the available documents in order to only extract knowledge relevant to the development of this dissertation. This knowledge can be useful to identify solutions for the proposed problem or to find limitations on possible solutions. In order to review the state of the art Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) will be used, to ensure an organized literary review.

2.1 Research Methodology

This section will describe in detail the methods used for producing the literary review. This includes the research questions, the search terms including inclusion and exclusion parameters and knowledge sources can be found, it is important to note that this methodology only applies to the articles related to machine learning. The articles related to psychology were retrieved without using a fixed methodology.

2.1.1 Research questions

In order to know what to look for, a main question was asked: "How can a computer recognize emotion behind a message accurately?". This question tackles multiple areas of science such as computer science and psychology. Given this question's complexity, it is prudent to split it into simpler subquestions. This approach divides the research process into smaller batches and in turn simplifies the whole process.

The questions are as follows:

1. How can a computer recognize the emotion behind a message accurately?

2. How can we represent an emotion mathematically?

2.1.2 Research Sources

Table 1 presents the sources used to extract information to answer the research questions.

Table 1: Research Sources

Source	**Source Link**
IEEE Xplore	https://ieeexplore.ieee.org/
B-On	https://www.b-on.pt/
ACL Anthology	https://aclanthology.org/

2.1.3 Search terms

In order to filter many irrelevant articles out from our research phase, many search terms are being used in conjunction. Since this study delves into multiple research areas these terms may be used independently for each of the questions asked. As a result, studies can be grouped and categorized for better organization.

Table 2 contain the respective search terms for each of the research questions that the study tackles:

Table 2: Search Terms

Question Scope	**Search Term**
Main topic for research	Emotion Recognition
Context	HR || ""
Text processing	Text mining
Artificial Intelligence Methods	Neural-Network || Deep-Learning || Machine Learning

2.1.4 Inclusion and Exclusion parameters

Even if an article matches this study's research field and includes the assigned search terms, there may be some details that can make it undesirable to extract information from. To ensure that all articles used contain information that is useful for the research stage, some inclusion and exclusion parameters are introduced. Each article must meet the inclusion parameters and not meet the exclusion in order to be added to the literary review.

These parameters will differ for each of the main topic areas investigated, in this case Computer Science and Psychology, since the evolution of these two areas differ a lot, due to their age and the problems they tackle.

The inclusion parameters for the articles linked to Computer Science are as follows:

- The article must be reviewed by its peers

- The article must be written in English
- The base of the article must be linked to machine learning

The exclusion parameters are the following:

- The article is 10+ years old
- The article is duplicated

2.2 Result analysis

This section presents a brief analysis of the results returned from the research done.

Initially, a set of 29,273 articles met the search parameters used for this research. However, after applying the quality criteria as per the PRISMA methodology, a smaller subset was extracted. Following a thorough review of each article's title, abstract, and content, only 5 articles were found to be relevant for the development of this study. These will be further explored in the following sections.

2.3 Results Discussion

In this section, each of the questions asked previously will be answered with the aid of the articles researched.

2.3.1 How can we define an emotion?

Webster dictionary defines an emotion as *a conscious mental reaction (such as anger or fear) subjectively experienced as a strong feeling usually directed toward a specific object and typically accompanied by physiological and behavioral changes in the body* [10]. As with all things related to the human mind, it is highly subjective and differs from person to person. As a result, defining an emotion is not an easy task for a human being and even harder for a computer. In order to extract an emotion behind a message, it is important to grasp the concept of emotions clearly.

There are various ways to define or classify an emotion in a way that is processable by a computer. These can be defined as models that can have varying degrees of accuracy and complexity. Depending on the use case one might prefer to use a complex model in order to reach more complex emotion. This can however decrease the efficiency of the learning algorithms and might affect the accuracy of the resulting solution. Keeping the model as simple as possible while being able to handle the problem given, is key to having the best solution possible. As a result, some research must be done in order to find the best model suitable for the given problem.

Emotions can be classified discretely or dimensionally. As a rule of thumb discrete models tend to be simpler but can only classify basic emotions while dimensional models are able to define emotions that are a result of a mix of more primitive emotions.

2.3.1.1 Discrete classification models

According to William James [11] emotion can be categorized into four basic sections, fear, grief, love, and rage. His concept relies on the study of *organic reverberation* or in simpler terms body language.

Ekman [12] expands this model by defining an emotion as 6 possible concrete types which are anger, disgust, fear, happiness, sadness, and surprise. This model is based on the facial expressions made when reacting to an event.

Richard and Bernice Lazarus [13] classify emotions with a deck of 15 possible types of aesthetic experience, anger, anxiety, compassion, depression, envy, fright, gratitude, guilt, happiness, hope, jealousy, love, pride, relief, sadness, and shame.

There are many more discrete classification models with a varying number of possible emotion types, some, like the Emotion Annotation and Representation Language (EARL) [14] contain over 45 different types of emotion. However, depending on the use case, some emotion types may be discarded. In this study only a few emotions might be useful for an efficient solution, so simpler models should be used

2.3.1.2 Dimensional classification models

A general consensus is that any emotion can be classified by assigning a value to each of the three following dimensions:

- Valence: Represents how positive or negative the emotion is.
- Arousal: Represents how much energy is in the emotion
- Power: Represents the degree of power the emotion has.

Posner, Russel, and Peterson's circumplex model [15] classifies emotions bi-dimensionally by regarding only the valence and arousal dimensions.

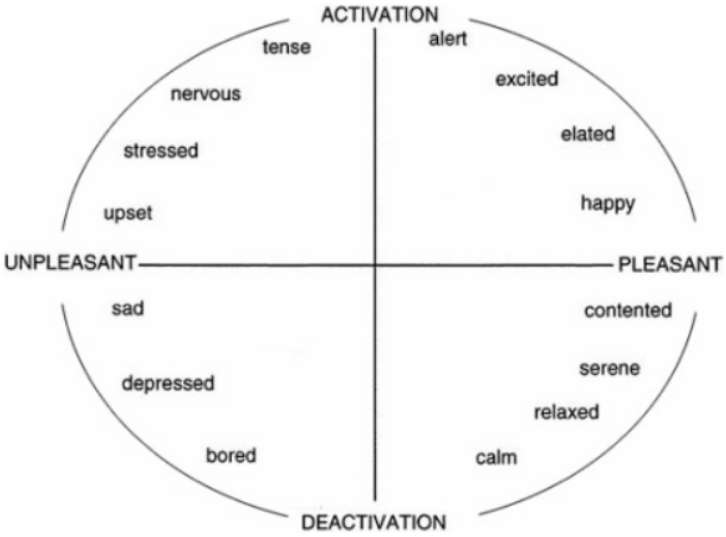


Figure 2: Emotion Circumplex Model

Plutchik [16] handles the concept of emotions multidimensionally by creating different models defined by 8 main emotion types, each of which blend and create more precise subtypes.

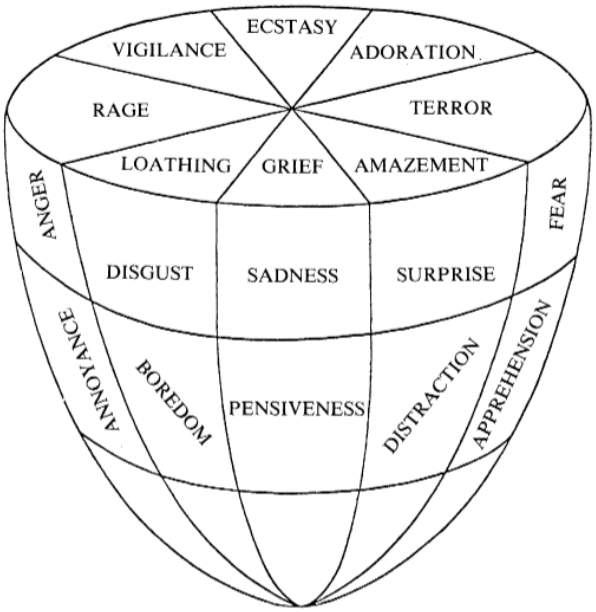


Figure 3: Plutchik's Model

Depending on how precise emotions need to be defined, different models can be used. While better emotion precision may seem desirable it makes the task harder for the computer to process, so it might be wise to prioritize machine learning model efficiency over emotion precision.

For this study's goal, a polarity concept might also be viable since it is enough to know if an emotion is positive or negative without knowing which type it is. This simplifies the problem and makes it trivial for a computer to recognize the polarity.

2.3.2 How can a computer recognize emotion behind a message accurately?

In Poria et al. (2019) [17] a study was made about the environment of emotion recognition in conversations at the time. It highlights some challenges one might face when tackling a problem in this domain. One of those challenges is the definition and categorization of an emotion, which is already addressed in the previous section. Another challenge is the importance of context in a conversation. Emotions can be hidden under the context of a conversation which makes it impossible to extract them if one only has the grasp of each message individually. This affects the viability of models trained with context-independent data such as tweets since this kind of messages are pretty self-contained and are not influenced by past messages. It also highlights some nuances that are usually present in conversations, such as emotional shifts, the presence of sarcasm among others. Since this study focuses on conversations between an HR representative and an employee, we can expect a more formal way of conducting conversations. However there can always be edge cases where those nuances appear that need to be addressed.

The study also enumerates and compares some datasets with relevant data that can be used to train machine learning models, capable of extracting emotions from conversations. Some datasets like the Interactive Emotional Dyadic Motion Capture (IEMOCAP) [18], SEMAINE [19] and Multimodal EmotionLines Dataset (MELD) [20] contain data in multiple formats (audio, video, and text) while DailyDialog [21], EmotionLines [22] and EmoContext [23] only contain data in text form. For this study's scope, only data in text form will be relevant. Out of the datasets mentioned, only SEMAINE defines emotions dimensionally by labeling the data with four values: arousal, valence, expectancy, and power, the other datasets only contain categorical emotion labels. MELD is a dataset dedicated to multiparty conversations, something that will not be considered in the current project's scope.

The study concludes by comparing different state-of-the-art models by using the mentioned datasets in the training phase and testing their performance against each other.

In Alswaidan et al. (2020) [24] a survey regarding state-of-the-art approaches for emotion recognition in text was made. Here, a lot of useful information for someone looking to start a project in the area is condensed.

Like the previous study, it starts by enumerating the different ways that an emotion can be defined. While this has already been discussed in this study, the survey expands by adding a third emotion modeling approach, the Appraisal modeling approach. This approach can be seen as an extension of the dimensional approaches. Its model bases itself on the appraisal theory, which states that an emotion can be extracted by evaluating the events a person has been subjected to and basing the emotion by studying

the person's experience, goals, and opportunities for actions. In this scenario, emotions are extracted by studying changes in many components of the human mind including cognition, physiology, feelings, expressions, etc. Most of these components cannot be evaluated in this study's scope, so this approach is not viable to use in the current context.

The survey then starts listing useful resources to train models to extract emotion. It lists and compares six datasets. The first one is Alm which consists of 185 children's stories, where each one is labeled as neutral, anger-disgust, sadness, fear, happiness, positive surprise, and negative surprise. Aman consists of blog posts labelled with one of the six emotions from Ekman's model [12]. International Survey on Emotion Antecedents and Reactions (ISEAR) [25], a survey conducted, where people labeled some of their experiences as joy, fear, anger, sadness, disgust, shame, and guilt. SemEval-2007 [26] compiles headlines from different news sources and labels them using Ekman's model [12] as well. SemEval-2018 [27] consists of tweets labeled as anger, disgust, fear, joy, love, optimism, pessimism, sadness, surprise, trust, or neutral. SemEval-2019 is another name given to the EmoContext dataset mentioned in [17]. It consists of three turns of exchanges between two individuals where each exchange is either labeled as joy, anger, sadness, or others. For the current context this is the datasets that seem the most viable to use among the mentioned ones. The survey then lists the amount of data available in each dataset and inserts it into a table for comparison.

The survey also enumerates some lexicons related to the problem in question. Lexicons are collections of words and/or phrases along with their associated information, such as their part of speech, meaning, and context. The lexicons mentioned add additional emotional information regarding certain words, simplifying the learning process of emotion recognition models and increasing their overall performance. It lists and briefly explains sixteen lexicons.

Some approaches used to recognize emotion are then listed along with some studies that use them. The first approach, the keyword-based approach, relies on finding occurrences of certain keywords related to a specific label. The most common technique is the keyword-spotting technique. Here each emotion is defined by a list of keywords using lexicons and then after some text pre-processing each remaining word from a sentence is iterated and checked against the keyword lists. Depending on how many words from the sentence are inside each emotion's keyword list, the emotion label for the sentence is determined. Figure 4 illustrates an example of a keyword-based approach.

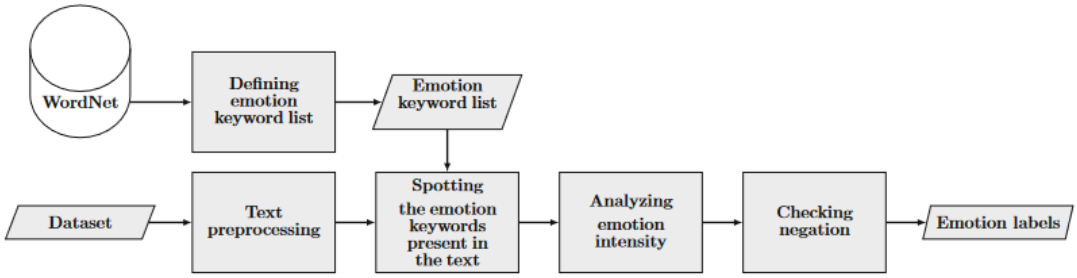


Figure 4: Keyword Spotting

The second approach, the rule-based approach, consists of manipulating the information present in order to be able to extract a conclusion. The information is pre-processed, then, the emotion rules are extracted using linguistic, statistics, and computational concepts and the best ones are selected. Finally, those rules are applied to the dataset in order to extract the emotion labels.

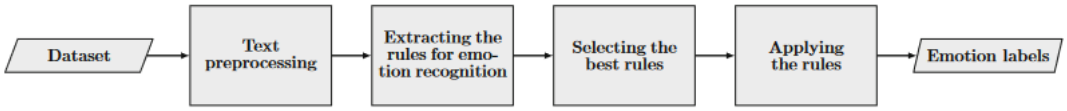


Figure 5: Rule Based Approach

The third approach, the classical learning based approach, provides the system with the ability to automatically learn and improve from experience. These use the more basic forms of machine learning algorithms, the most used one is Support Vector Machine (SVM),shown in Figure 6, which is a supervised algorithm. In this approach, a dataset's content is processed, and its most useful features are extracted, after that it is fed into the algorithm which will then output a model capable of predicting the labels of unseen data.

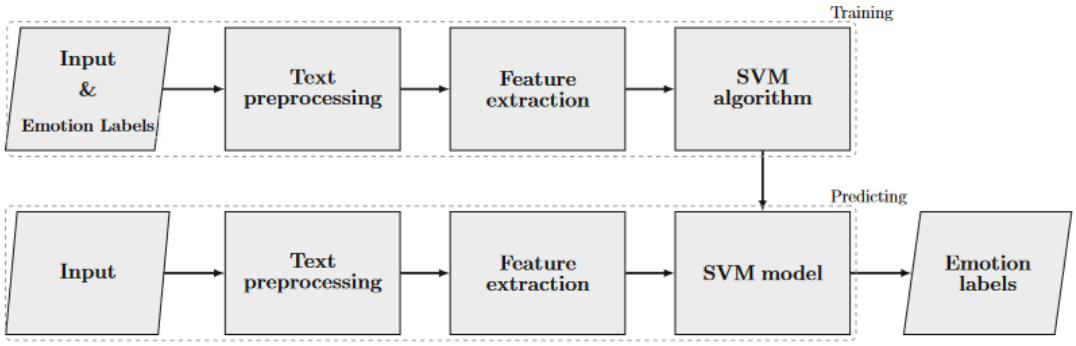


Figure 6: A Classical Learning Approach (SVM)

The fourth approach, the Deep-learning approach, is a branch of machine learning where programs learn from experience and understand what's around them in terms of a hierarchy of concepts. With this method, each concept is created in terms of the relation of simpler concepts. This allows programs to understand complicated concepts by relating them with simpler ones. The most common deep learning model for emotion recognition is the Long Short-Term Memory (LSTM), found in Figure 7. This model expands on Recurrent Neural Network (RNN) by giving it the capability of handling long term dependencies. This means that this model is able to extract emotions by using the whole conversation as a medium, instead of just being able to relate the emotion to the message in which it is embedded. To use these models first the dataset is processed and an embedding layer is created and fed into one or more LSTM layers. Then its output is fed into a dense neural network properly configured to perform the emotion classification.

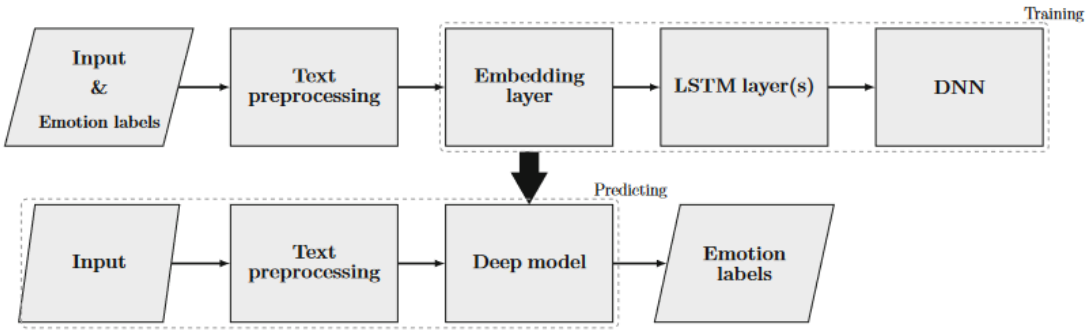


Figure 7: Long Short-Term Memory

In [28], a multi-labeled Deep Convolutional Neural Network (DCNN) was introduced to detect at least two emotions in text. The research concluded that most emotion datasets contain multiple extractable emotions. Since many existing solutions only detect the most prominent emotion in their input, this study suggests potential methods for enhancing the overall ecosystem. This approach consists of an architecture composed of multiple components, each one with a distinct task.

The first component is the dataset. In order for the model to learn, it needs a large amount of correctly labeled data. This enables the model to extract the information present in the data and extract its own conclusions about what influences the data in order to give its respective label. The method used in this solution was to extract and combine all data with more than one label from two already available datasets (Cleaned Balanced Emotional Tweets (CBET) [29] and semEval).

After composing the dataset, it is necessary to process the data for the purpose of optimizing the learning phase of the solution. This approach manipulated the dataset in the following ways:

- **Data cleanup** - All irrelevant data such as usernames, links, etc., only add noise to the dataset which makes it harder and more computationally intensive to analyze it and as such are removed.

- **Deleting Stop-Words** - These are some of the most common words used in standard conversations and, like the data referred above, add little to no value for the learning algorithms. Some examples of stop words are conjunctions, suffixes, and pronouns which are also removed from the whole dataset.

- **Tokenizing** - The process of tokenizing consists of converting all text into tokens which is a more refined way of representing words. These tokens can then be transformed into a vector and fed into the algorithm, improving its learning efficiency.

Not all words have the same impact when defining an underlying emotion in a piece of text, so even with all relevant words stored in a way that is easily readable by the machine might not be enough to produce an efficient model. The study's approach to this problem is adding an extra layer, which they called "**Attention Layer**", to the architecture capable of calculating the weight each word has by creating "*a distributed representation of the entire document with respect to the highlighted words*". The final result will represent the importance the word has in regard to the definition of the text's label.

After allocating an importance value to each word data is fed to the "**Word Embedding Layer**".To tackle this layer the study used two different approaches, one that works in tune with all other layers in order to *learn the semantic vector of the words itself by training and giving weights to the words*. The other uses two pre-trained models, fastText [30] and Global Vectors for Word Representation (GloVe) [31], in order to compare their performance with the first approach.

The last layer of the whole solution's architecture is the deep learning model itself. This layer is composed of many components whose task is to perform mathematic operations on the vectors given by the previous layers and, finally, to be able to produce an accurate prediction of the emotion present in the given input. The whole architecture is too complex to provide a brief summary of its inner workings.

In order to evaluate the performance of the developed method, the study used the following metrics:

- **Jaccard index**: This metric calculates the accuracy of the multi-labeling process by dividing the number of correctly predicted labels by the total number of predicted labels.

$$Jaccard\ index = \frac{1}{N}\sum_{i=1}^{N}\frac{Y_i \cap \hat{Y}_i}{Y_i \cup \hat{Y}_i} \tag{2.1}$$

  Where:

  Y Correct Label

  $\hat{Y}$ Predicted Label

  N Number of samples

- **Hamming Loss**: The Hamming Loss metric computes the fraction of the mislabeled samples by the total number of labels.

$$Hamming\ Loss = \frac{1}{dl}\sum_{i=1}^{d}\sum_{j=1}^{l}\left|h_{ij}\Delta y_{ij}\right| \tag{2.2}$$

Where:

y Correct Label set

h Predicted Label set

d Sample

l Label

For the rest of the metrics, we can assume that:

TP True Positive

TN True Negative

FP False Positive

FN False Negative

- **Micro Average Precision**: The micro average formulas denote the behavior of individual classes. This formula calculates the precision of each class by dividing the sum of all true positive predictions by the sum of all positive predictions, regardless of being correct or not.

$$Micro\ average\ precision = \frac{\sum_{j=1}^{i} TP_j}{\sum_{j=1}^{i} TP_j + \sum_{j=1}^{i} FP_j} \tag{2.3}$$

- **Micro Average Recall**: Like the previous formula, this one also denotes the behavior of each individual class. It calculates the recall of each class by dividing the sum of all true positive predictions by the sum of all positive results.

$$Micro\ average\ recall = \frac{\sum_{j=1}^{i} TP_j}{\sum_{j=1}^{i} TP_j + \sum_{j=1}^{i} FN_j} \tag{2.4}$$

- **Macro Average Precision**: The macro average precision is the arithmetic mean of all the precision values for the different classes.

$$Precision = \frac{TP}{TP + FP} \tag{2.5}$$

$$Macro\ Average\ Precision = \frac{\sum_{m=1}^{k} Precision_k}{k} \tag{2.6}$$

- **Macro Average Recall**: The macro average recall is the arithmetic mean of all the recall values for the different classes.

$$Recall = \frac{TP}{TP + FN} \tag{2.7}$$

$$Macro\ Average\ Recall = \frac{\sum_{m=1}^{k} Recall_k}{k} \tag{2.8}$$

- **F1 Score**: The F1 score is the harmonic mean of all the precision and recall values.

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{2.9}$$

The evaluation was made using 5-fold cross-validation, using the model by itself and in conjunction with both previously mentioned word embedding models separately.

By the time this study came out, the values provided by its metrics showed an all around better performance in comparison to other studies that were available at the time.

In Batbaatar et al (2019) [32], a new Neural Network architecture was proposed, the Semantic-Emotion Neural Network (SENN), a model capable of obtaining and utilizing both semantic and emotional information by adopting already trained word representations. This leads to better quality results in comparison to other state-of-the-art approaches, and can be improved even further with other emotional word embeddings.

The architecture is mainly composed of two subnetworks. The first subnetwork utilizes Bidirectional Long-Short Term Memory (BiLSTM) in order to uncover the context of each word by analyzing the information from both the forward and backward directions. The second subnetwork uses a Convolutional Neural Network (CNN) to extract emotional features and concentrates on the emotional connections between words and the text as a whole.

To train the model, various publicly available datasets were used separately, allowing for a comparison of the model performance between them. All datasets were filtered in order to only include sentences that contained just one label and where the emotion associated with it was included in Ekman's [12] model of emotions (joy, fear, sadness, surprise, anger, and disgust). In those sentences a cleanup was executed where all numbers, special characters and usernames were removed and all uppercase characters were changed to their lower case variant. The optimal parameters of the models were also obtained using the grid search algorithm in order to achieve the best performance out of each model tested.

The performance of each model was evaluated using the Recall (2.7), Precision (2.5) and F1-Score (2.9) metrics. The models were then compared against some baseline models, like Naive-Bayes classifier, Random Forest Classifier, Support Vector Machine, among others. These used basic methods like Bag-of-Words or Term Frequency-Inverse Document Frequency as the class of each text. They were also compared to more complex deep learning models like Convolutional Neural Network, Gated Recurrent Unit (GRU), and others, using frameworks like Word2Vec [33], GloVe [31] or FastText [30] for semantic word embeddings and Emotion Word Embeddings (EWE)[34] for emotion word embedding to convert the data into a more refined state.

The study concluded that its proposed approach performed better than all the other tested models while having a comparable if not lower execution time, which made their solution the best one to use in many cases.

In Liu et al (2021)[35] a model architecture was presented. This new model named Semantic Emoticon Emotion Recognition (SEER) expands on the Bidirectional Gated Recurrent Unit (BiGRU) network by adding an attention mechanism to give weight to more impactful words. SEER also utilizes the presence of emojis to give more accurate predictions instead of discarding them completely like most developed models do. Since emojis can appear in text conversations between HR representatives and employees, this functionality can be beneficial to have in order to obtain the best predictions possible.

The architecture starts with a sentence classification layer. This layer is tasked to parse and divide the sentences given into four groups: sentences that contain explicit emotional words, sentences that contain explicit emotional words as well as emoticons, sentences that have words where the emotion is implicit, and sentences that have implicit emotional words and emoticons.

All the organized data is then fed to the Word Embedding layer. This layer refines the words given into a word vector representation using the framework Word2Vec which is sent to the next layer of the model.

With the words processed into a word vector, the BiGRU layer is capable of extracting semantic features from the word aspect.

In order to be able to understand and extract meaning from emojis, this solution utilizes an emoji embedding layer. This layer contains an emoji distribution where each emoji is labeled with six values which represent its weight for each of the 6 emotions present in Ekman's model [12]. Each weight is computed by gathering sentences that contain emojis from existing labeled datasets and dividing the number of sentences of each emotion where the emoji is present by the total number of sentences where the emoji has appeared.

Like the previous models, this architecture also calculates the impact each word has in defining an emotion by using an attention layer. This layer utilizes a self-attention mechanism to compute the weight of the information present in the BiGRU's output without the need for external data.

The output of the emoji embedding and attention layer is then merged in the connection layer and a final vector containing all the relevant data from the previous steps is sent to the final layer, the emotion recognition layer that, with the input given, is able to predict the emotion behind the initial message.

In the training phase, the study used a dataset that contained posts from the Chinese platform "Weibo". As a result, the model was trained using Chinese characters and all English characters as well as numbers, special characters, Weibo usernames had to be removed in order to eliminate any possible noise in the data.

The model's performance was studied by calculating its macro precision (2.6), macro recall (2.8), and macro F1-Score (2.10) and tested against baseline methods like Word Lexicons, SVMs, BiGRU without emoji support, among others. In the end SEER, outperformed every other method, even the BiGRU implementation without emoji support. This makes it safe to conclude that BiGRU models are a good

choice when choosing the optimal model and that emojis are a valuable asset to analyze when extracting emotions from text.

$$MacroF1\ Score = 2 * \frac{MacroPrecision * MacroRecall}{MacroPrecision + MacroRecall} \tag{2.10}$$

In Adoma et al. (2019) [36], several state-of-the-art transformers are employed to extract emotions from text. Their performance is meticulously studied and compared to identify their individual weaknesses and strengths, ultimately aiming to determine the best model for the task.

In this paper, Bidirecional Encoder Representations from Transformers (BERT) [37], Robustly Optimized BERT Pretraining Approach (RoBERTa) [38], Distilled BERT (distilBERT) [39], and XLNet [40] are trained using the ISEAR dataset.

Before the training process, the dataset underwent preprocessing to enhance the efficiency and performance of all models. All entries lacking relevant information were removed. For the remaining entries, any information that could negatively affect the training phase, such as special characters, double spacing, tags, and other irregular expressions, was eliminated. Stopwords were also removed to improve training time. Subsequently, emotion labels were converted into a numerical scale. The longest string was identified, and its size was used to determine the tokenizer size, thus preventing any truncation. Finally, all entries were padded to match the tokenizer size.

The first model utilized in this study was the BERT-base-uncased model, characterized by twelve layered transformer blocks. Each block comprises twelve self-attention [41] heads and 768 hidden layers, yielding a grand total of approximately 110 million parameters. The model processed one sentence at a time, where input sentences were tokenized and converted into their corresponding indexes using the BERT tokenizer library, denoted as input IDs. To structure the input appropriately, [CLS] (classification token) are included at the beginning and the [SEP] (separate segment token) are added at the end of each sentence. An input attention mask of fixed length was employed, with '0' signifying padded tokens and '1' indicating unpadded tokens. Each transformer block received a list of token embeddings and generated an output feature vector of the same length. Notably, the output of the 12th transformer layer, containing vector transformations of prediction probabilities, served as the aggregated sequence representation used for making classifications.

RoBERTa and distilBERT are both variants of BERT, sharing a similar architecture. distilBERT has been modified to reduce the number of parameters while striving to maintain performance, resulting in a faster and more lightweight solution. On the other hand, RoBERTa represents a fine-tuned iteration of BERT, incorporating the masking process during the training phase to enhance the model's performance. However, this improvement comes at the cost of increased model size and longer training times.

The XLNet-base-cased model featured twelve transformer layers with 768 hidden layers, along with twelve attention head layers. Sequences were tokenized using the XLNet tokenizer, followed by padding, and subsequent classification.

In this study, it is shown that all models are somewhat efficient in detecting emotion in text from the ISEAR dataset. RoBERTa performed the best out of all four. distilBERT while having the worst performance was the one with the fastest training time. XLNet came in second in accuracy but was the slowest of them all.

Chapter 3

Solution Planning

This chapter delves into the initial planning and exploration of available tools and data necessary for developing the final solution. It begins with the description and illustration of the solution's high-level architecture, followed by an enumeration and brief overview of the retrieved datasets. Next, data processing techniques are outlined and examined, followed by a study of methods to address data imbalance. Finally, the chapter includes a brief exploration of the model selection process and available models.

3.1 Solution Architecture Plan

Before any work can be done, a general idea of how the solution will work should already be defined. This ensures that the development of the study does not deviate from the original goals and helps keeping it an efficient process.

As stated in the previous chapters, the goal of this study is to develop a program capable of detecting emotion ingrained in pieces of text so as to give insight of an employee's state of mind. With this in mind, it can be inferred that an operational solution should be able to take a string as input and output it as an emotion (either categorically or discretely).

A high-level architecture can be seen in the image below.

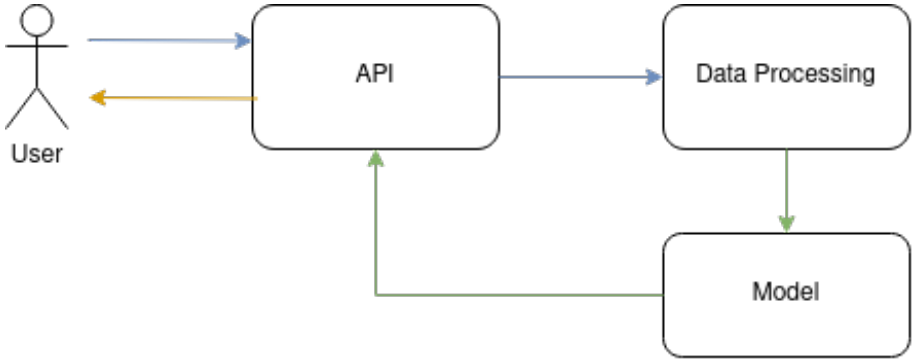


Figure 8: Solution simple architecture

In the figure 8, blue arrows represent data in text format while green arrows represent data in numerical format and orange represents the emotion.

The first component of the architecture is the API. The API will handle all interaction between the user and the system by providing a simple interface where the user can input any text while also exposing entry-points to allow interaction between different systems. This allows the possibility of integrating the solution with other software like Microsoft Teams, Slack, etc.

The second component is Data Processing. It features an extensive set of functions that are tasked with converting the input given by the API into a more refined state that is usable for the model efficiently. It is responsible for filtering irrelevant data or manipulating data in such a way that it is more efficiently used by the model.

The third component is the model. The model is a machine learning algorithm tasked with finding the correlation between the input given and the data in which it was trained in order to reach a correct conclusion, in this case, the correct emotion behind the text.

3.2 Dataset

For the development of this study, no dataset was provided. As such, an in-depth analysis of the publicly available datasets is needed in order to ensure that the data within them is coherent with the goal of the study.

3.2.1 Daily Dialog

Daily Dialog is an English conversation dataset composed of 13 118 different dialogues. Each dialogue contains several speaker turns, each with their assigned label, which corresponds to the emotion displayed by the speaker as well as the topic of the conversation and the act behind the utterance.

The labels for each type are the following:

- Topic

1. Ordinary Life
2. School Life
3. Culture & Education
4. Attitude & Emotion
5. Relationship
6. Tourism
7. Health
8. Work
9. Politics
10. Finance

- Act
  1. Inform
  2. Question
  3. Directive
  4. Commisive
- Emotion
  1. Neutral
  2. Anger
  3. Disgust
  4. Fear
  5. Happiness
  6. Sadness
  7. Surprise

In this study, each dialogue will be flattened and only the emotion label will be used. With the dialogues flattened the model will be provided with 102 979 different utterances, each labeled with the corresponding emotion. The frequency of each emotion can be seen in graph 9.

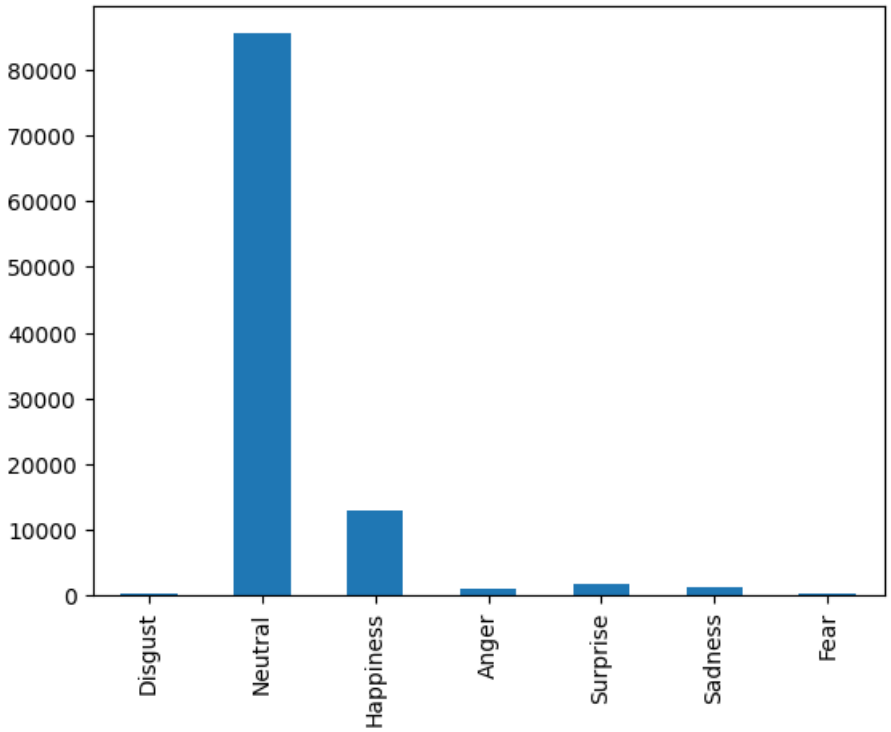


Figure 9: Daily Dialog bar plot

3.2.2 MELD

MELD is a dataset composed of 2 160 dialogues from the television series ”Friends”.

Each line of each dialogue is composed by:

- Utterance
- Speaker
- Emotion
  - Neutral
  - Surprise
  - Fear
  - Sadness
  - Joy
  - Disgust
  - Anger
- Polarity

  - Negative
  - Neutral
  - Positive

- Dialogue_ID The dialogue identifier
- Utterance_ID The utterance identifier inside each dialogue

The remaining columns are not relevant for the goal of this study.

Similarly to other datasets dialogues will be flattened and only the utterances will be used. This results in 12,840 unique utterances with their respective emotion and polarity. The column that represented the speaker of each utterance was also removed to generalize the model.

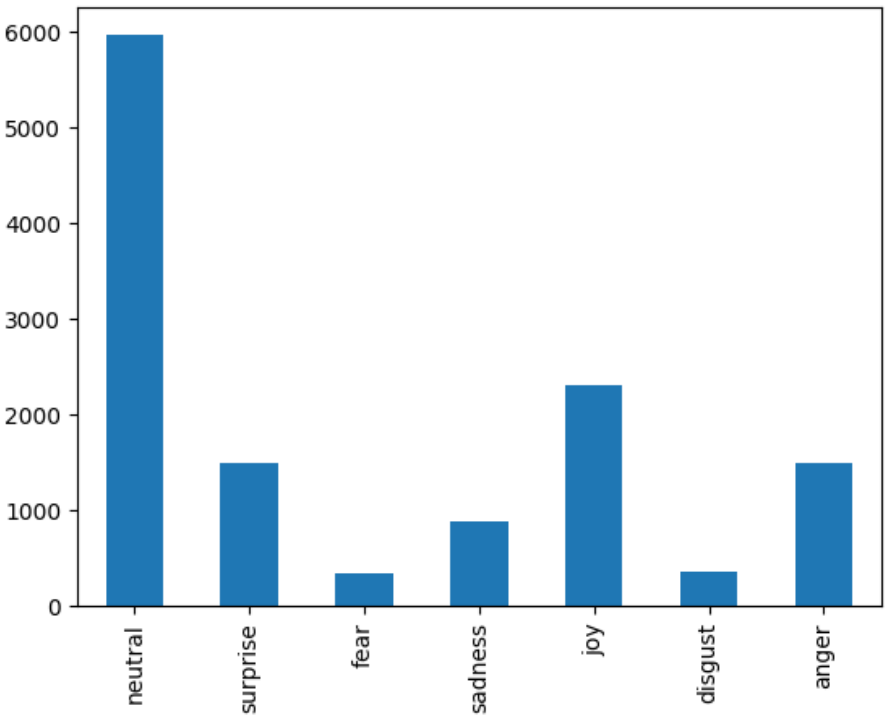


Figure 10: Meld emotion count

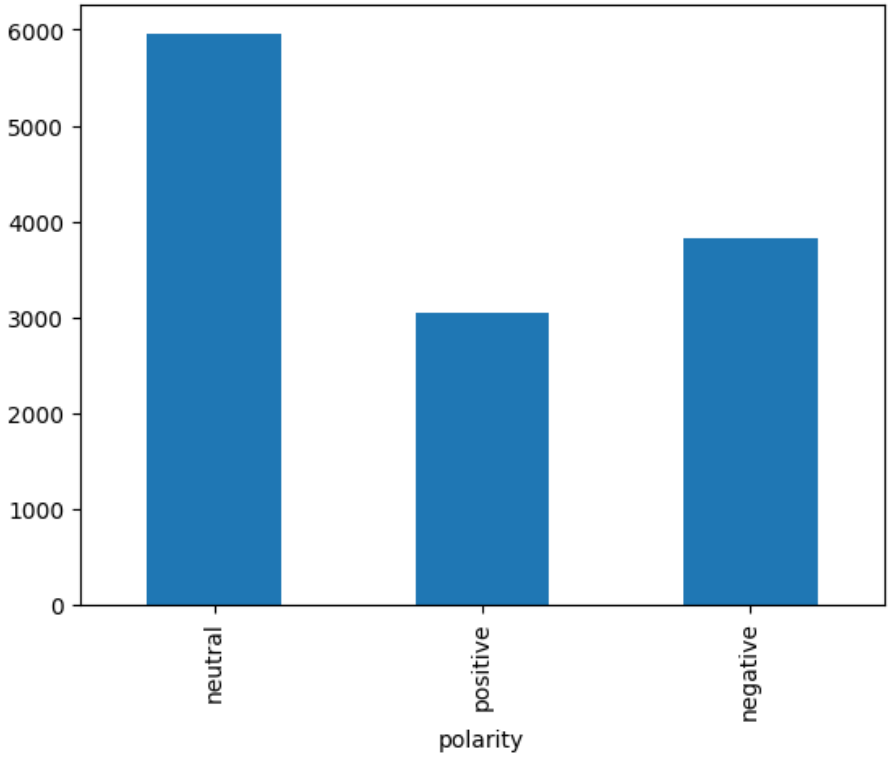


Figure 11: Meld polarity count

3.2.3 ScenarioSA

ScenarioSA is a dyadic database used for sentiment analysis. It is composed of 2 214 dialogues.

Each line of each dialogue is composed by:

- Utterance
- Speaker (A or B)
- Polarity
  - Negative (-1)
  - Neutral (0)
  - Positive (1)

It also contains each speaker's polarity at the end of the dialogue and the topic however these are not relevant for the study.

Since this dataset only contains information regarding polarity and not individual emotions, it can be concluded that its usage is not optimal for this study.

3.2.4 goEmotions

GoEmotions comprises 76 536 comments manually labeled by humans from the platform Reddit, with each comment assigned one or more of 27 unique emotions. This approach allows the development of a model capable of predicting finer-grained emotions, however, to maintain consistency with other tested datasets, the labels were streamlined to encompass only the six emotions outlined in Ekman's model.

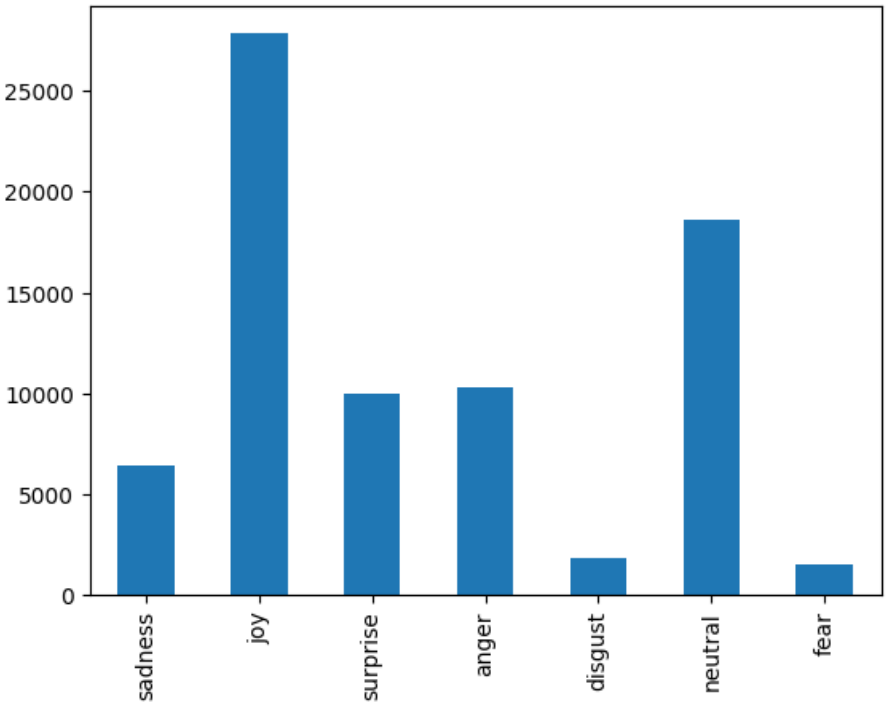


Figure 12: Goemotions emotion count

3.3 Pre processing

Machine learning models work better when handling pure numbers. This results in the necessity to process the data so as to have it in a more appropriate format to feed the models.

3.3.1 Tokenization

Tokenization is the process of splitting data into smaller pieces (called tokens). Some models that handle text use words as tokens while others use individual letters. In the context of this study, it involves separating each word from their respective sentence.

> I'm sorry that you are mad ! -> ["I'm","sorry","that","you","are","mad","!"]

The number of tokens present in the training phase should always be kept at a minimum to ensure an optimized process. So, to accomplish that without losing valuable data, some considerations can be made when analysing the dataset.

- *You're* and *You are* will result in different tokens.
- *You* and *you* will also result in different tokens due to the different casing.
- *You!* and *You* result on different tokens because of *!*.
- There are some words that add little relevance to the meaning of a sentence and its removal **may** be beneficial. (Stopwords)
- There are some words that can be simplified in order to reduce the number of different tokens in the training data. (Lemmatization)

3.3.2 Stopwords

Stopwords are words that are commonly used in any given language. These words usually have no relevance in the development of machine learning models since they are present in almost every sentence and their presence adds little insight for the prediction of the desired label.

> removeStopwords("I am sorry that you are mad !") -> "sorry mad !"

By removing stopwords we are decreasing the amount of data used to train the model, which in return reduces the time needed for the solution to be fully trained. However, the removal of stopwords can lead to the sentence completely changing its meaning. Stopwords such as "*not*"are crucial for the context of a sentence and thus are essential for the underlying emotion in the sentence. For example:

> removeStopwords("I am not having a great day!") -> "great day!"

In this example by removing the stopwords the sentence changes from being a Neutral/Sadness sentence to being related to the Happiness emotion, something that would severely impact negatively the model. So, for this study's domain, this technique is usually not recommended.

3.3.3 Lemmatization

The lemmatization of a sentence is the process of converting each word into its *lemma*. Lemmas are the most primitive form of a word, by using lemmas we can ensure that the words derived from the same source will result in the same token.

> lemmatize("Golden foot is the king of dancing! He danced his feet off.") -> "Golden foot be the king of dance! He dance his foot off.")

Lemmatization may not be an easy task to execute, there are some words where their lemma depends on the context of the sentence. Let's say for example:

> lemmatize("He always leaves the tree leaves scattered on the balcony ") -> "he always leave the tree leave scatter on the balcony"

This lemmatization does not represent the original sentence correctly. It can be seen that both occurrences of the word "leaves"should result in different lemmas, in this case in leave and leaf.

3.3.4 Embedding

In machine learning, embedding refers to the process of representing data - such as words, sentences, or images, as continuous vectors or numerical representations in a lower-dimensional space.

The most basic way of embedding a sentence is by assigning a number to each distinct word and representing the sentence as an array with said numbers. Frequently, each word is represented by its frequency (how common it is) in the dataset, mainly by being represented by its occurrence ranking (most common word is 1, 2nd is 2 and so on...). Since machines don't know any linguistic properties by default, this method is the most basic and simplest way of converting data to a more readable state, being the most commonly used method when there is no additional information about words available. However, models trained with data in this state will only be able to reach conclusions about the properties and relationships between words by analyzing their position in relation to each other.

3.3.4.1 Vectorization

By vectorizing, data is refined in a way that makes additional information more accessible for the model. For example, with vectors, the relationships between words can be easily represented. Considering the following 5 words as 2-dimensional vectors:

- man = (-1,0)
- woman = (1,0)
- royalty = (0,1)
- king = (-1,1)
- queen = (1,1)

It can be inferred that:
> man + royalty = king
> woman + royalty = queen
> The word queen is more closely related to the word woman than to the word man.

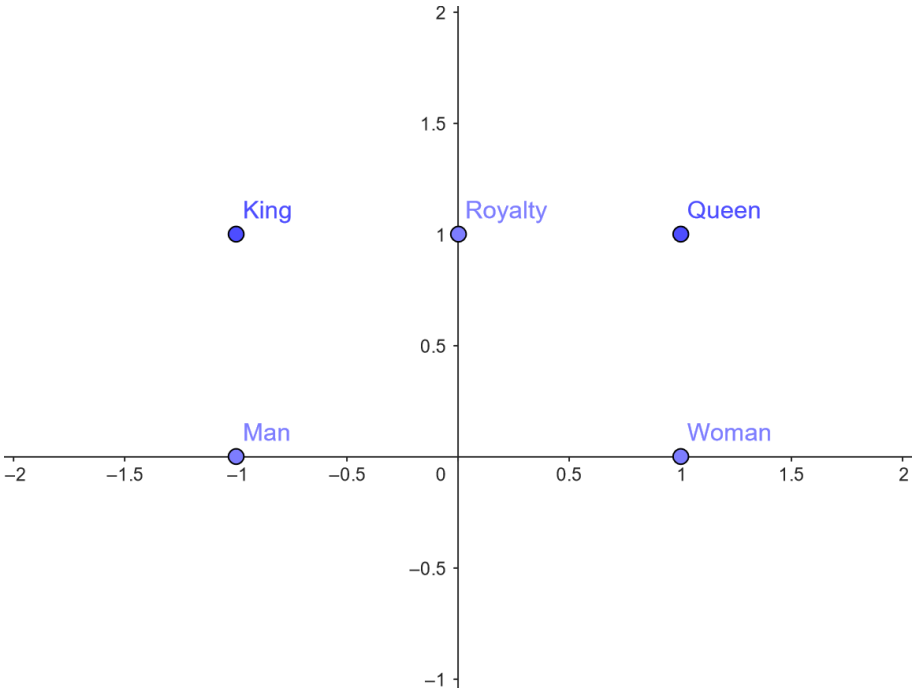


Figure 13: Vectorization Graph

With 2-dimensional vectors, the model has access to additional information. However, due to machines not having any problem handling multiple dimensions in contrast to humans, most models represent data as vectors with dimensions ranging between 100 and 300, which can give magnitudes more information than its raw counterpart. As a result, the training process gets more efficient, since the model has more useful information to extract and to reach correct conclusions.

3.4 Working against bias

When training a model with an imbalanced dataset, the most probable outcome is that the model will lean heavily towards the most common labels. In order to counteract this event, several solutions may be applied. These solutions mostly fit into one of these categories:

3.4.1 Undersampling

Undersampling is a balancing technique that involves the removal of data related to the most common labels. This results in a more balanced dataset. However, one must be cautious, since the removal of useful data may be harmful to the model's development.

For this study, since the dataset that is being used is so unbalanced an undersampling technique is used where it limits the quantity of entries for each class to a certain point. If that limit is, for example,

1000 then every class will have 1000 entries at most. This makes the solution better at generalizing since it doesn't rely as much on the majored classes to predict the output.

3.4.2 Oversampling

In contrast to the previous technique, oversampling battles imbalance by adding data to the less common labels. This can be done by cloning already existing data or by producing new synthetic data that is automatically generated by "mutating"the rows that belong to the less common classes.

3.4.2.1 Cloning data

This method extracts information related to the less frequent classes and duplicates it in order to balance the data. This augmentation technique can be applied to any type of data. However, it is 100% artificial, since no new useful information is being added, which may help the model deal with imbalance but will make the model less capable of generalizing results.

By cloning data the model had a more balanced dataset, however since the minority classes are so outnumbered in relation to the majored class the dataset became composed of mostly duplicates which increased the time needed for the training phase without a big improvement for the model itself.

3.4.2.2 Synthetic data

A dataset can be balanced by creating small changes to existing information in order to generate "new"data to aid the model's training process. This method relies on how the input is represented and may need several tweaks in order to be viable in broader use cases. Synthetic data can be exemplified as follows:

- When the input is a sentence, new data can be generated by replacing words with one of its own synonyms.
- When the input is formed by numerical values, new information may be generated by randomly tweaking its numbers.

This technique reduces the presence of possible duplicates that were present in the previous technique, however, its implementation is more complex and depends on the type of information present in the dataset.

3.4.2.3 Data Collection

While it may seem obvious, one of the most effective ways to augment a dataset is by manually adding new annotated data. This approach is considered the best solution because it ensures that all the added data is both organic and correctly annotated. However, it is also the method that demands the most effort and investigation, making it potentially infeasible in some cases.

3.4.3 Weight distribution

If modifying the dataset's content is not desired then it is possible to still achieve a more balanced and efficient training by assigning different weights to the existing data. This means that some information may have more influence on how the model behaves than others. So, by assigning a larger weight to data related to the less common classes then a more balanced training phase can be achieved.

Weights can be assigned in one of two ways:

3.4.3.1 Class weights

In this method, the same weight is applied to data regarding the same class. For example, for a dataset with two classes, and one class is 10 times more common than the other, a weight of 10 could be assigned to the minority class and a weight of 1 to the majority class. This would tell the model that the minority class is more important, and it would help the model to learn to predict the minority class more accurately.

3.4.3.2 Sample weights

This method goes one step above and assigns a different weight to each sample of the dataset. This can be accomplished if a more in-depth fine-tuning is necessary, as it allows for the adjustment of the importance of each individual data point within the cluster of information.

3.5 Models

3.5.1 Classification or Regression - What to use?

Before choosing the proper models to implement in the study one must research the domain of the problem in order to use the right tools for the job. The first step is to determine if it is a regression or classification problem by identifying the format of the expected output.

Classification models are machine learning algorithms used to categorize or assign discrete labels or classes to input data based on patterns and features. They are employed to make predictions about which category or class new data points belong to. Some examples of Classification problems are:

- Identifying if a certain message is spam;
- Predicting the sentiment behind a piece of information;
- Classifying images in order to identify objects or detect anomalies.

Regression models are statistical techniques used in machine learning to analyze the relationship between one or more independent variables and a dependent variable, aiming to predict or estimate numerical outcomes. Some examples of Regression problems are:

- Predicting stock prices of certain companies;
- Predicting prices based on supply/demand;
- Predicting future climate patterns based on previous behaviour.

By comparing the main uses for each category, it is easy to conclude that Classification models are the more appropriate solution for this study.

3.5.2 Logistic Regression

Despite having regression in its name, Logistic Regression is actually mainly used for classification problems. This technique models the relationship between the input features and the probability of belonging to the positive class (class 1) using a logistic function. This function also known as the sigmoid function is an S-Shaped curve that maps any value to a value between 0-1. This means that Logistic Regression really shines in binary classification problems, however, it can also be used for multiclass predictions.

3.5.3 Support Vector Machine (SVM)

Support Vector Machine is a technique used for classification and regression problems. It maps all available data into an n-dimensional space and finds an hyperplane capable of isolating all data belonging to each class. In a binary classification, it finds a function capable of separating data belonging to the two classes. If there is no proper function capable of isolating the information then all information can be mapped into a higher dimensional to allow for such function to exist. Natively, this technique only supports binary classification, however, it is possible to break down multi-class problems into several binary classifications. Therefore, by merging several SVMs it is possible to use it in a multiclass problem.

3.5.4 Long Short-Term Memory (LSTM)

Long Short-Term Memory [42] is a type of Recurrent Neural Network (RNN) architecture that is designed to address limitations in other RNN architecture such has the incapacity to handle long term dependencies between sequential data. This makes LSTM ideal for datasets where data can be influenced by previous entries such as conversations where context matters. In addition to the standard architecture, a variant exists, the Bidirectional Long Short-Term Memory, or BiLSTM for short, is able to handle dependencies in both forward and backward directions. This means that it is also capable of handling datasets where an entry can be influenced by the previous entries and the following ones.

3.5.5 Transformers

Transformers introduced by Vaswani et al. (2017) [41], are a class of deep learning models gaining traction due to their capacity to revolutionize several domains of machine learning. They utilize a novel self-attention mechanism that enables the capture of intricate relationships among elements in a sequence, including words in a sentence or tokens in a document, even when they are widely separated within the sequence. This characteristic makes Transformers highly effective in various applications, spanning Natural Language Processing, Computer Vision, and more. They serve as the foundational component for Large Language Models (LLMs) like GPT-3, BERT, and XLNet, which are increasingly prominent in diverse sectors of everyday life.

3.5.6 Hyperparameters

Hyperparameters play a crucial role in machine learning models, functioning much like dials in physical machines. These parameters enable control over the learning phase of the models. Given that machine learning solutions are often seen as 'black boxes,' direct modification of the model's behavior is not feasible, so, to ensure optimal training the tuning hyperparameters is typically performed before initiating the training process.

The following subsections list some of the hyperparameters that can be tuned.

3.5.6.1 Epoch

An epoch represents the full iteration of a machine learning model through the entire training dataset. It involves passing the dataset through the model for a number of times determined by the chosen epoch count. Repeated iterations over the dataset can lead to improved performance on the training data since it gradually adapts itself to the data being given.

However, an important consideration is that too many epochs can have a counterproductive effect. This is because, as the model extensively learns the intricacies of the training data, it may become overly specialized and lose its ability to generalize or, in other words, overfit. Overfitting occurs when the model starts capturing noise and peculiarities in the training data, and this can make it less effective in making accurate predictions on unseen data, which is a crucial aspect of machine learning [43].

3.5.6.2 Learning Rate

The learning rate determines the pace at which the model updates its own parameters. A higher learning rate results in more significant parameter updates, as the model takes larger steps in an attempt to reach the optimal values that correspond to the global minimum. Selecting the appropriate learning rate is a challenging task because a large learning rate can lead to faster training but may lead to suboptimal models since it overskips potential optimal values during the search, while a small learning rate leads to longer training and can also potentially get stuck in local minima which also results in suboptimal models.

3.5.6.3 Batch size

The batch size is a neural network parameter that specifies the number of samples processed by the model before updating its weights. Like other hyperparameters, there is no universal, predefined value, as it heavily depends on the specific model and dataset and so, it is essential to strike a good balance when choosing a batch size.

Using a larger batch size can accelerate training by reducing the frequency of weight updates. However, this may result in decreased accuracy and an increased risk of overfitting. It also demands more memory, which can be unfeasible for hardware with limited resources. Conversely, a smaller batch size consumes less memory but executes more frequent weight updates, leading to greater computational complexity and longer training times.

Chapter 4

Solution Implementation

This chapter describes the implementation process of the solution, which is divided into 6 sections. The first section includes an overview of the technologies employed, including programming languages and specialized libraries. Subsequently, the hardware used in the development of the solution is listed. The next section details the entire process of dataset implementation, covering how the dataset is integrated into the code and the comparison of each retrieved dataset. Following that, there is a section outlining all the data processing steps carried out to ensure a high-quality solution. Next, a section illustrates the model's implementation, commencing with the testing and comparison of several possible architectures, ranging from simple algorithms to state-of-the-art solutions. Subsequently, the most suitable architecture is chosen, and the optimal parameter setup is investigated. Finally, the chapter concludes with the description of the component responsible for handling communication between users and the solution.

4.1 Used technologies

Due to the flexibility and available documentation of machine learning and data analysis/manipulation, all components of the architecture will be developed using the ***Python*** programming language. There are several libraries and frameworks that will aid in all steps of the development of the solution available.

The API is composed of two parts, the GUI which gives a graphical interface so a user can use the solution in a simple way and the RESTful API itself which works as an interface between external information and the system. Both parts are developed using Flask, which is a Python web microframework that allows the development of web applications . It has a limited set of features however due to being very lightweight it is very good for making small webApps. The inclusion of the jinja2 template engine allows for the development of both parts without any additional tools.

The Data Processing module features the usage of some NLP focused libraries in conjunction with

pure python code. The nature of this study allows for experimentation and thus some functions may not be used in the final solution. This module relies heavily on the usage of ***Pandas*** dataframes as the main way to store and handle data. Some NLP tasks such as lemmatization and stopword identification are handled partly by the ***NLTK*** library. Word2Vec was one of the tested vectorization methods and is implemented by the ***GenSim*** library.

The model component of the final solution will only include one model, however many models were tested and studied in order to find the most appropriate architecture. The baseline models such as Logistic Regression and Support Vector Machine are implemented using the ***SciKitLearn*** library due to the vast amount of documentation online and general robustness to implement machine learning algorithms. For the deep-learning models, the ***Tensorflow*** library along with the ***Keras*** framework are used due to their higher level approach, maturity and documentation available.

4.2 Used Hardware

No suitable hardware was provided for the development of this study, so all development and training of models were executed on a personal computer. Since the device is not specialized in this field, the development of the solution will be limited by its performance and efficiency. The hardware specifications are as follows:

Table 3: Hardware Used

CPU	Intel® Core™ i5-6600K 4-Core Processor 3.50Ghz
GPU	ASUS GTX 1060 Dual OC 6GB
RAM	16GB (8GBx2) DDR4 3000MHz
Disk	SSD

4.3 Dataset Analysis

4.3.1 Dataset Implementation

In order to quickly and easily test each dataset, a special dataset class was defined. When testing a new dataset, only the importing phase needs to be properly implemented, since all methods of the dataset class can be run regardless of which dataset is used as long as it has been imported properly.

This class consists of a string that holds the dataset's name, two boolean values indicating whether the dataset includes information about emotion polarity or the emotions themselves, a matrix of dictionaries containing dialogue information with speaker details if available, and a dataframe containing all relevant data in that structure.

Once a dataset has been imported into the class, a wide array of implemented methods can be used. These methods encompass text preprocessing tasks, such as the removal stopwords or the lemmatization

```
    def __init__(self,name : str, dialogs : List[List[Dict]] = None,
↪   hasemotion : bool = False, haspolarity : bool = False):
        self.name : str = name
        self.hasemotion : bool = hasemotion
        self.haspolarity : bool = haspolarity
        self.dialogs : List[List[Dict]] = dialogs
        self.dataframe : pandas.DataFrame = None
```

Listing 1: Dataset class structure

of text, as well as dataset vectorization and merging. Additionally, this enables the retrieval of additional information, such as the mean and maximum sentence size within the dataset.

4.3.2 Dataset Comparison

To ensure that the model is fed with the best data possible, a comparison between the available datasets is necessary.

The collected datasets are the following:

- DailyDialog
- MELD
- ScenarioSA
- goEmotions

By defining the goal of the study as the development of a model capable of recognizing the emotion behind a message, it becomes apparent that the ScenarioSA dataset can be easily discarded, as it solely contains data for recognizing sentimental polarity and lacks information about the emotion itself.

The emotion distribution of the remaining emotions is as follows:

Table 4: Emotion Distribution

Dataset	Neutral	Joy	Anger	Surprise	Sadness	Disgust	Fear
MELD	5960	2312	1500	1490	876	364	338
goEmotions	17772	17943	7022	6668	4032	1013	929
DailyDialog	85572	12885	1022	1823	1150	353	174

By analyzing the table we can see that DailyDialog is by far the biggest dataset, while MELD is the smallest. However, size isn't everything. It can be seen that while all datasets are unbalanced, DailyDialog is the most unbalanced dataset out of them all, and it also contains the fewest entries regarding some emotions, such as fear or disgust.

4.3.2.1 MELD

The MELD dataset consists of dialogues originally scripted for a comedy television series. Consequently, the data does not accurately represent real-life conversations, especially in the context of HR-employee communication. However, it's worth noting that the text is well-composed, as the dataset comprises transcriptions from the TV show, devoid of uncommon abbreviations and internet slang.

It's important to recognize that the utterances in the dataset are attributed to one of the show's six main characters. This suggests that the sentences are mostly tailored to reflect their personalities, making it challenging to generalize the dataset for training models in the HR context.

Despite its imbalance, with the most prevalent emotion being neutral, the dataset provides labels for individual emotions and the polarity of those emotions within each utterance. This labeling facilitates the identification of ambiguous emotions; for instance, the "surprise"emotion that can be either positive or negative.

4.3.2.2 goEmotions

The goEmotions dataset stands out from the other datasets due to its composition of informal language and sentences. Comprising Reddit comments, this dataset has no constraints on the use of slang and uncommon abbreviations. While this informality may not be suitable for accurately representing HR-employee communication, it can offer certain valuable insights. Notably, some utterances within the dataset incorporate emojis, a common feature in chats between HR departments and employees. Emojis can provide a direct means of conveying emotions through text, making them highly relevant for emotion extraction purposes.

Additionally, it's worth mentioning that the dataset is somewhat unbalanced. However, it has the most evenly distributed range of emotions among all the collected datasets.

4.3.2.3 DailyDialog

The DailyDialog dataset contains several favorable characteristics. Upon close examination of its content, both advantages and disadvantages in the context of this topic can be identified.

Firstly, it's worth noting that the text within this dataset is well-written, devoid of uncommon abbreviations or slang words. The dialogues are also diverse, encompassing a wide range of topics and conversational styles.

However, as mentioned earlier, a notable drawback is the dataset's considerable imbalance concerning various emotions. The majority of sentences are labeled with the "neutral"emotion, constituting approximately 83% of the entire dataset. Additionally, the conversations in this dataset are not directly related to HR-employee communication.

4.3.2.4 Chosen dataset

After a comprehensive analysis and comparison of various datasets, it becomes evident that the DailyDialog dataset stands out as the most suitable choice in terms of aligning with the problem's scope and context. Although it is not specifically tailored to workplace conversations, its content represents more common dialogs and so is the one that closely mirrors the kinds of conversations that employees might have the most. This inherent similarity makes it the prime candidate for delivering the best results when integrated into the workplace environment.

One of the biggest problems regarding the chosen dataset is how imbalanced it is. Over 83% of the data is related to the neutral emotion. By analyzing the dataset it can be concluded that there are some dialogs that only contain the neutral emotion. Those dialogs can be removed in order to have a more balanced dataset without removing much relevant information.

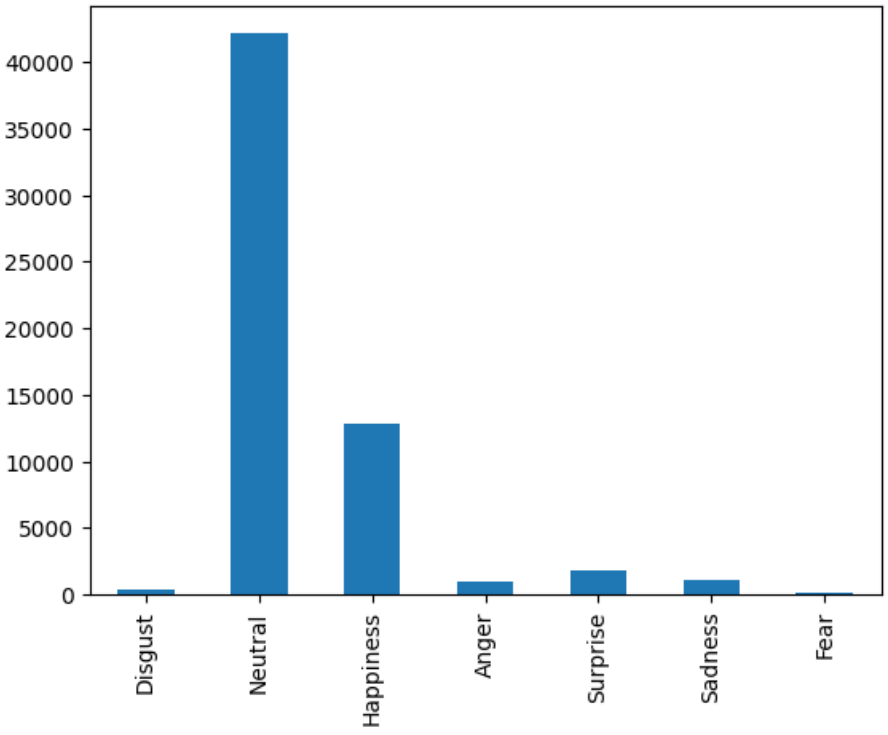


Figure 14: Filtered Daily Dialog bar plot

While still heavily biased toward the neutral emotion, this filtration reduced the model's reliance on the neutral emotion which returned better results. However the model wasn't still able to generalize well enough to be useful with sentences outside the dataset's genre, so additional manipulation was required.

Further oversampling, undersampling and weight distribution techniques were tested and mixed in order to counteract dataset imbalance. The technique that yielded the best results was a truncation undersampling technique where each emotion was limited to having 2000 sentences at most for each. This resulted in the following distribution:

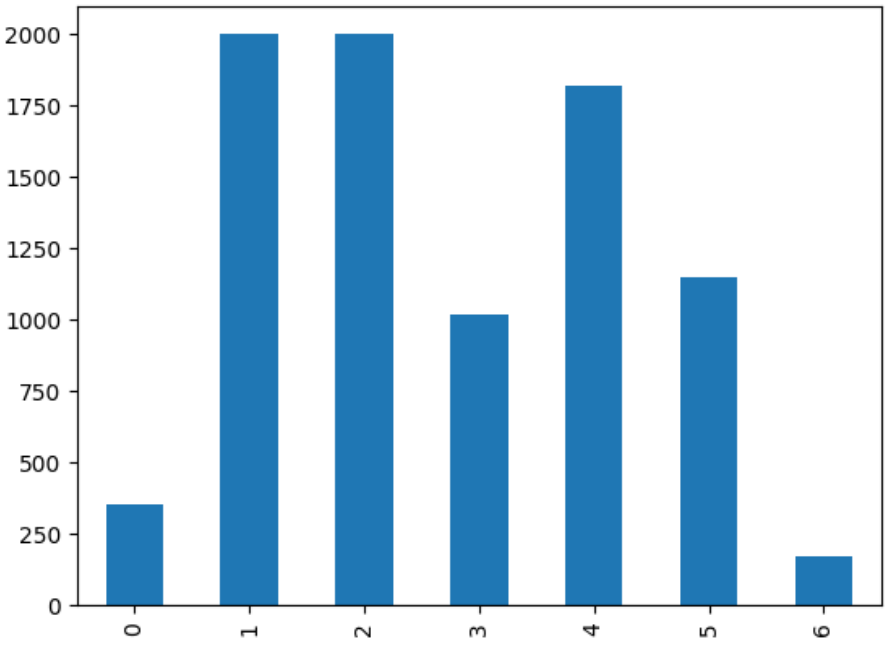


Figure 15: Final Emotion Distribution

It is possible to gain insight into the influence of some words in the emotion labeling process by generating word clouds for each label of the dataset.

Words with a high amount of emotional charge are represented in a large part of sentences belonging to a certain emotion. Words such as hate, sick, awful and horrible are prominent in sentences labeled with disgust while words that have a more positive connotation such as great, nice, love, good and beautiful are common in sentences labeled with joy.

Figure 16: Daily Dialog Disgust WordCloud

Figure 17: Daily Dialog Happiness WordCloud

Neutral labelled sentences do not contain emotionally charged words, these sentences are the only ones where the word "Oh"is not prominently present, this adds relevance to this word since it can be a good way to exclude the neutral emotion when labelling a sentence.

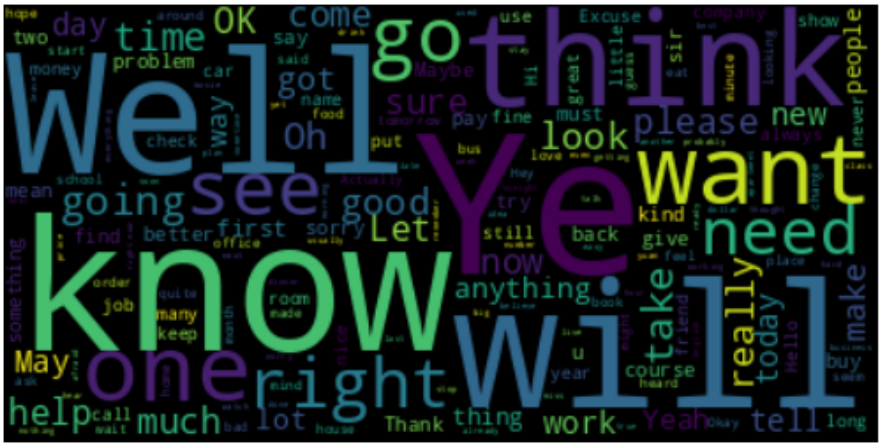


Figure 18: Daily Dialog Neutral WordCloud

4.4 Pre-processing

In order to sanitize all data and optimize the training process for efficiency, several transformations were applied and a pipeline was formed.

First, all special characters were isolated from words to allow the tokenizer to recognize the word and special characters as separate tokens. This results in a reduced number of tokens and enhances the importance of special characters, such as punctuation, in the training process.

Next, each word is individually tagged with its grammatical form (adjective, noun, verb, etc.). This tag assists the lemmatization process by providing additional information, which is particularly helpful for lemmatizing ambiguous words, such as 'leaves'. 'Leaves' can be interpreted as the plural of 'leaf' or as the conjugated form of the verb 'leave'.

Each word is then lemmatized, reducing the number of distinct tokens and highlighting the significance of words with multiple derivations.

Subsequently, the results are converted to lowercase, ensuring that identical words are represented by the same token, regardless of their capitalization.

The sentence is then tokenized at the word level, resulting in an array where each element represents a word from the sentence.

Finally, the results are vectorized. The format of each token depends on the model used. For transformer models like BERT, each token is represented by a numerical value, while for others, each token is vectorized into a 200-dimensional vector using GLoVe embedding. The size of the vectorized result varies based on the dataset used. Since models require consistent input dimensions, the ideal approach is to pad all lists of vectors to match the size of the largest sentence in terms of the number of tokens - this ensures that all data from the dataset is used. However, due to hardware limitations, training models with large arrays of vectors was not possible. As a result, the maximum size of the vectorized results is determined by finding a size that is able to handle whole sentences for 75% of the dataset, with the remainder being truncated.

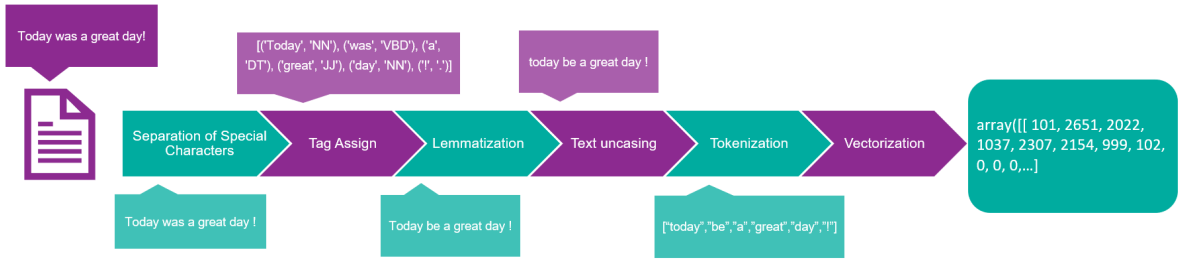


Figure 19: Preprocessing pipeline

In this pipeline Stopwords are not removed due to their important influence in the underlying emotion of some sentences.

4.5 Result Analysis

4.5.1 Model Comparison

In order to find the best solution several models were tested. These models range from basic algorithms such as Logistic Regression and Support Vector Machines which will act as a baseline to State of the Art

transformers such as BERT and XLNet, there will be some Recurrent Neural Networks tested as well.

The diagram in figure 20 shows the compared models:

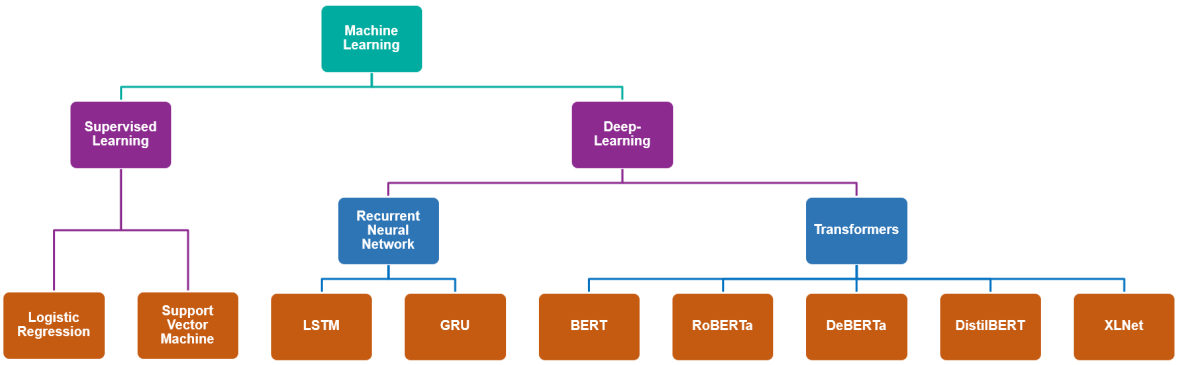


Figure 20: Tested models

4.5.1.1 Baseline Models

In this section, an analysis, and comparison of a Logistic Regression Model and a Support Vector Machine is done. These two supervised learning algorithms will be included as baselines for comparing all others. Due to their simplicity, their implementation is straightforward. Although their performance may not match that of the rest, they can serve as a means to validate the effectiveness of the other solutions.

In order to also test the effectiveness of different embeddings, both algorithms will be trained with tokens represented as single values using a Count Vectorizer and represented as 200-Dimensional vectors using GLoVe embeddings, table 5 illustrates the efficiency of the different setups.

Table 5: Baseline Algorithms Comparison

Setup	Time to train	F1_Score	Accuracy	Loss
CountVectorizer + Logistic Regression	**1.70s**	**0.5619**	**0.6519**	**1.0537**
GLoVe + Logistic Regression	25.02s	0.3537	0.4884	1.3589
CountVectorizer + SVM	19.30s	0.5536	0.6441	1.0745
GLoVe+ SVM	34.56s	0.3400	0.4845	1.3600

From this table, it is evident that the Logistic Regression algorithm outperforms the Support Vector Machine and is faster as well. Additionally, the use of more complex token representations has a negative impact on the model's performance. Therefore, in this case, the simpler solution is recommended.

4.5.1.2 Recurrent Neural Networks

Recurrent Neural Network architectures have been a primary choice in various machine learning applications, including classification problems. In this section, two specific architectures: Long Short-Term Memory and Gated Recurrent Unit (GRU) are tested. Like the models described in the previous section, these architectures do not incorporate a specialized embedding process. Therefore, their performance using embeddings from GLoVe and simple tokenization is evaluated.

Table 6: RNN Comparison

	F1_Score	Accuracy	Loss
LSTM	0.4508	0.6231	1.2397
LSTM + GLoVe	0.4140	0.5792	1.3085
GRU	**0.4563**	**0.6325**	1.1760
GRU + GLoVe	0.3762	0.5139	**1.0041**

Much like the previous section, the utilization of GLoVe embeddings yields subpar results. While GLoVe embeddings typically have a positive impact on RNN architectures, as observed in the literature review, it's noteworthy that these models unexpectedly exhibit poorer performance compared to the baseline. One possible explanation may be an improper implementation of these architectures.

4.5.1.3 Transformers

Finally, a comparison of several transformers was conducted. These models represent state-of-the-art solutions with great potential in various areas of natural language processing, including text classification. Table 7 lists each tested transformer, along with its corresponding number of parameters, reflecting the complexity of each model.

Table 7: Tested Transformers

Transformer	Parameter Count
BERT	109 482 240
DeBERTa	183 831 552
DistilBERT	66 362 880
RoBERTa	124 645 632
XLNet	116 718 336

By analyzing the table it is evident that DistilBERT has by far the fewest parameters, while DeBERTa has the most. The remaining models exhibit relatively similar complexities.

All five transformers were trained and tested under identical conditions, using the same parameters and data. This approach ensures that any variations in performance metrics are solely attributable to the transformer itself, without any external influences.

The testing environment comprises the processed DailyDialog dataset along with the finalized model architecture featuring 128 units per dense layer, a batch size of 16 samples, and a dropout rate of 50%. The number of epochs and the learning rate are dynamically adjusted, as detailed in the following sections.

The plot in Figure 21 illustrates the evolution of each model, displaying the mean of all validation F1-Scores at the end of each epoch.

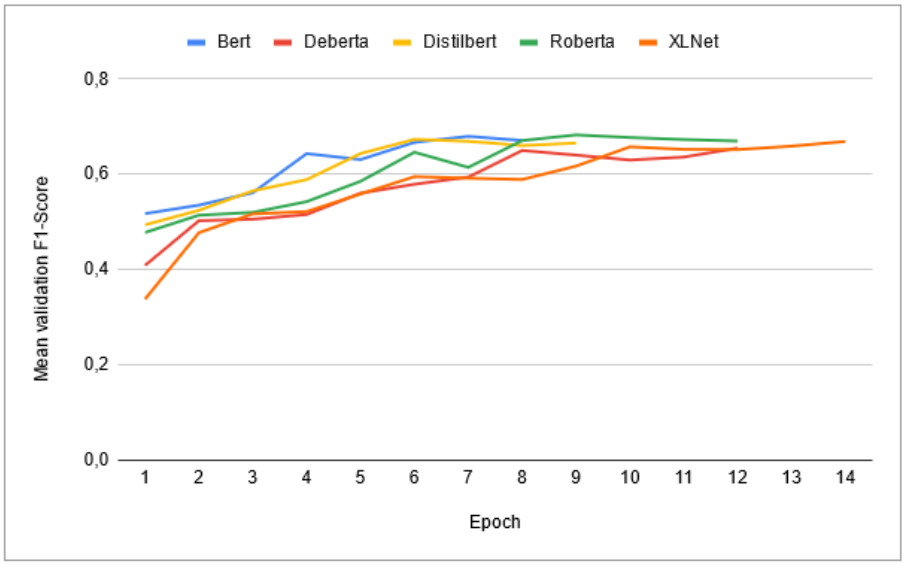


Figure 21: F1-Score Comparison

It is evident that performance remains consistently similar across all transformers, indicating that, in this case, the dataset is the bottleneck. Consequently, employing more complex models does not yield superior results. For this reason, opting for more lightweight and faster models proves to be the better choice.

Table 8: Accuracy per Label table

	Disgust	Neutral	Joy	Anger	Surprise	Sadness	Fear
BERT	0.4642	0.7400	0.8519	**0.6407**	0.8354	0.7612	0.4634
DeBERTa	0.4537	**0.7467**	**0.8666**	0.6310	**0.8385**	0.7652	0.2816
DistilBERT	**0.5510**	0.7313	0.8189	0.6393	0.8265	0.7466	0.4383
RoBERTa	0.4947	0.7301	0.8337	0.6289	0.8271	**0.7694**	**0.4938**
XLNet	0.5205	0.7240	0.8519	0.6262	0.8254	0.7380	0.3958

Upon examining the emotion distribution, as shown in Figure 15, and referencing table 8, it becomes evident that the model's performance is more significantly influenced by the available data rather than the choice of transformer. Disgust and fear emotions notably represent the minority classes within the dataset, and these are the ones that show the lowest accuracy. This highlights the necessity for a more balanced and improved dataset, however, given the rarity of these types of messages and the detrimental

impact of oversampling on the prediction of more common labels, it is not advisable to upsample these classes.

The plot in Figure 22 helps illustrate which model should be chosen based on their respective speeds.

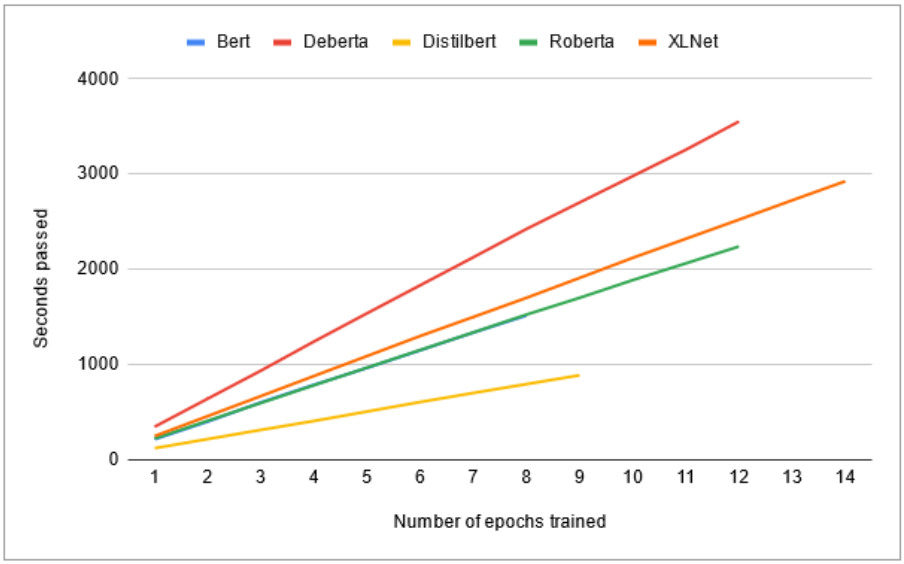


Figure 22: Training time comparison

Upon examining the plot in Figure 22, it becomes apparent that BERT requires the fewest epochs to reach the optimal solution. However, DistilBERT significantly outpaces the others in terms of time per epoch, making it the fastest to converge to the optimal solution.

A small overview of the training process for each model is presented in the table 9, which consists of six rows. The first row indicates the total number of epochs performed until the training met the stopping condition. The second row highlights the epoch when the best solution was achieved, while the third row displays the time in seconds taken to reach that solution. The subsequent rows depict the mean of all F1_Scores, accuracy, and loss values for the best solution of each model.

Table 9: Transformer Comparison

	Total Epochs	Optimal Epoch	Time to reach solution	F1_Score	Accuracy	Loss
BERT	**8**	7	1336	0.6796	0.7622	**1.0533**
DeBERTa	12	12	3553	0.6548	**0.7665**	1.1003
DistilBERT	9	**6**	**607**	0.6732	0.7509	1.1257
RoBERTa	12	9	1701	**0.6826**	0.7560	1.1070
XLNet	14	14	2925	0.6689	0.7528	1.5074

Based on this information, it is apparent that all models possess similar capabilities and can effectively handle the task with acceptable performance. Each model, except for XLNet, outperformed others in at least one metric. Given the overall similarity in performance and the desire for efficiency, DistilBERT was

selected as the final transformer solution due to its significantly lighter weight and faster processing speed compared to the competition.

4.6 Final Architecture

The finalized model is illustrated in Figure 23 comprises the selected transformer model, accompanied by additional layers responsible for processing the transformer's outputs and converting them into classification predictions. These layers are designed to be compatible with every tested transformer, simplifying the testing and comparison process by allowing for easy swapping of transformers.

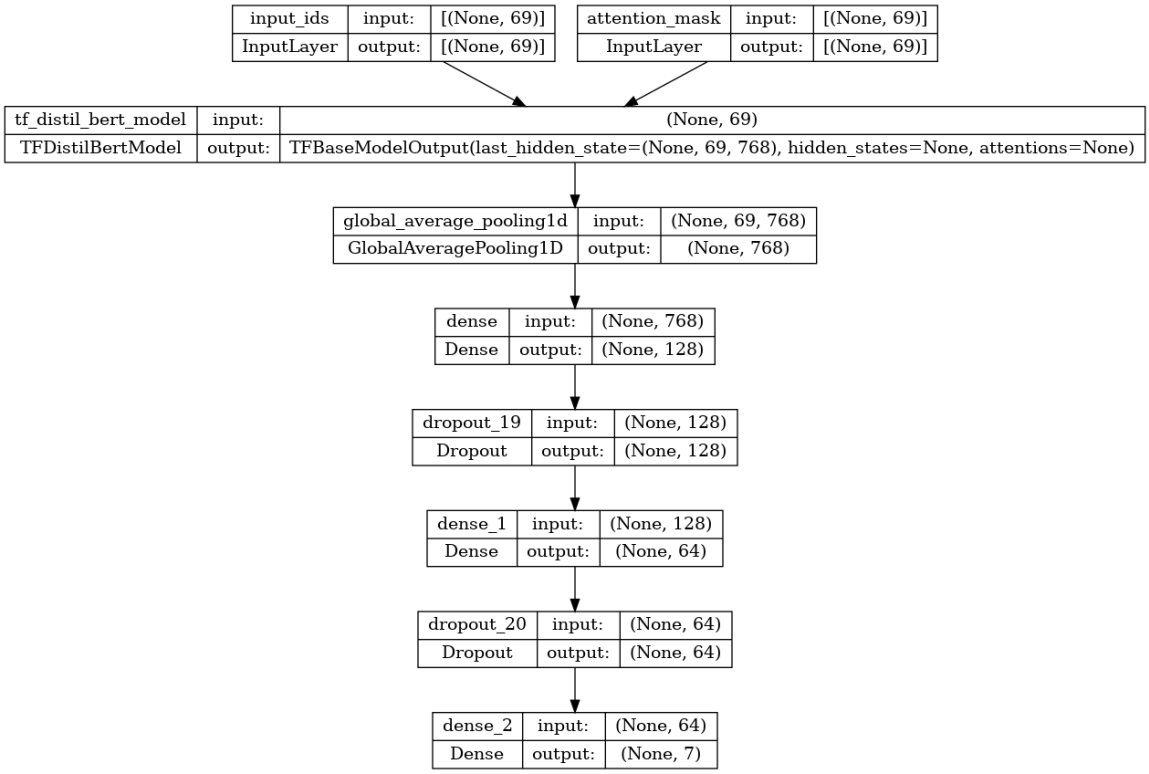


Figure 23: Model Architecture

The initial layer within this architectural design is the transformer model. This fundamental component serves as the backbone of the neural network. It takes two inputs: 'input_ids' and 'attention_mask.' The 'input_ids' input holds the tokens represented as a vector. Depending on the input's size, it may either be truncated or padded to fit within the vector, whose size is defined during model construction. The 'attention_mask' input consists of a vector of the same size, with each element being either 0 or 1, indicating whether the model should ignore the token or incorporate it during the training phase. This mechanism prevents padding from affecting the training process.

Following the transformer layer, the output is directed to the GlobalAveragePooling1D layer. This layer transforms the data into a 1-dimensional array by computing the average of each pooled value. This step is

crucial as it harmonizes the data's shape, making it compatible with subsequent layers while maintaining a balance in model complexity.

Subsequently, this array undergoes processing in a Dense Layer. This layer's role is to condense the information into the desired number of units, facilitating the feature extraction process.

The data then proceeds to a Dropout layer, which introduces an element of randomness by selectively resetting the values of some neurons. This technique is invaluable for mitigating overfitting, a common concern in machine learning.

Continuing on, the data is passed through another Dense layer, effectively compacting the data by reducing the number of neurons by half. This step contributes to the network's overall efficiency.

To further safeguard against overfitting, an additional dropout layer is employed. This acts as a safeguard against the model learning the training data too well and losing generalization capabilities.

Finally, a last Dense layer is employed, with as many units as there are classes in the classification task. This layer takes on the responsibility of making predictions. For each unit, it quantifies the model's confidence in assigning a given input to a particular class. Moreover, it incorporates regularizers to penalize overfitting during training, adding an extra layer of protection against this common issue.

4.6.1 Hyperparameter tuning

While the "heart"of the solution is already mostly defined by the chosen state-of-the-art model, there are several components where some fine-tune can be done in order to ensure the best training process, and as a result the most robust and optimized solution possible.

The best values for some parameters were found by testing and comparing different setups using the Hyperband algorithm [44]. This algorithm identified the parameter settings that yielded the lowest possible validation loss, thereby ensuring that the model's predictions closely align with the labels when provided with unseen data. Using the Hyperband algorithm, a range of dropout rates - including 20%, 30%, 40%, and 50% - was evaluated for the dropout layers. The number of units per dense layer varied between 32 and 128, with a step of 32 units. Additionally, the best batch size was found by testing all powers of 2, spanning from 2 to 64.

An initial learning rate of 2e-5 (or 0.00002) is used during the training phase. This rate is reduced by a factor of 0.5 every time the validation loss of the model starts converging. This allows a faster training, while also ensuring that the model does not converge into a suboptimal point.

A dynamic number of epochs is used during training. This is done by setting a large number of epochs but stopping the training process early if a condition is satisfied. In this case, the model stops learning prematurely if after 2 epochs there hasn't been a decrease of at least 0.02 on the validation_loss metric.

Table 10 lists the optimal hyperparameters obtained with the algorithm.

Table 10: Optimal Hyperparameters

Parameter	Value
Dropout rate	50%
Units per Dense Layer	128
Batch Size	32
Initial Learning Rate	8e-5

4.6.2 Final Results

Table 11 illustrates the performance of the solution after fine-tuning the model parameters present in table 10.

Table 11: Performance of final solution

F1-Score	Accuracy	Loss
0.6545	0.7538	0.8754

After examining the table, it becomes evident that, despite improvements in accuracy and loss, the model shows a lower F1-Score. Overall, the model's performance remains relatively consistent. This underscores the limited influence of the model's architecture on performance when compared to the quality of the training data. If the model were to select an emotion randomly, it would achieve an accuracy of approximately 14.28%. Therefore, the accuracy attained by the model indicates acceptable performance.

4.7 API

The current state of the developed API only serves as a proof-of-concept having the minimal feature set and presentation to be considered functional. It however serves as the foundation for possible future iterations of the component.

It provides two ways to interact with the solution. A GUI and a REST endpoint. The GUI provides a simple page with a single text box where a user can easily input the desired message and the API will return an Emoji symbolizing the underlying emotion in the sentence. Table 12 lists the emoji mapping used to represent the emotions.

Table 12: Emotion-Emoji Mapping

Emotion	Emoji
Neutral	😐
Joy	🙂
Sadness	😢
Fear	😨
Anger	😠
Surprise	😯
Disgust	🤢

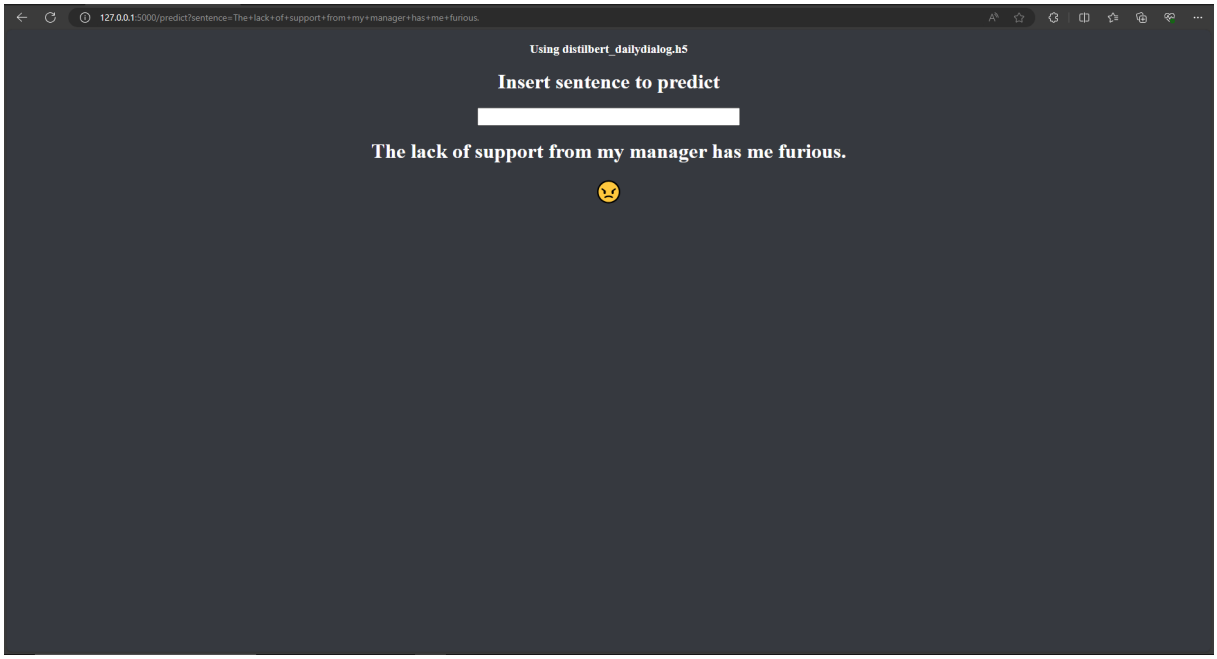


Figure 24: API Web page

When making predictions, if the difference between the emotion with the highest level of confidence and another emotion is less than or equal to 0.1, the GUI will display both emotions. This means that if a sentence elicits a high level of confidence for more than one emotion, it will be classified with multiple emotions. Figure 25 provides an example of a sentence with multiple emotions.

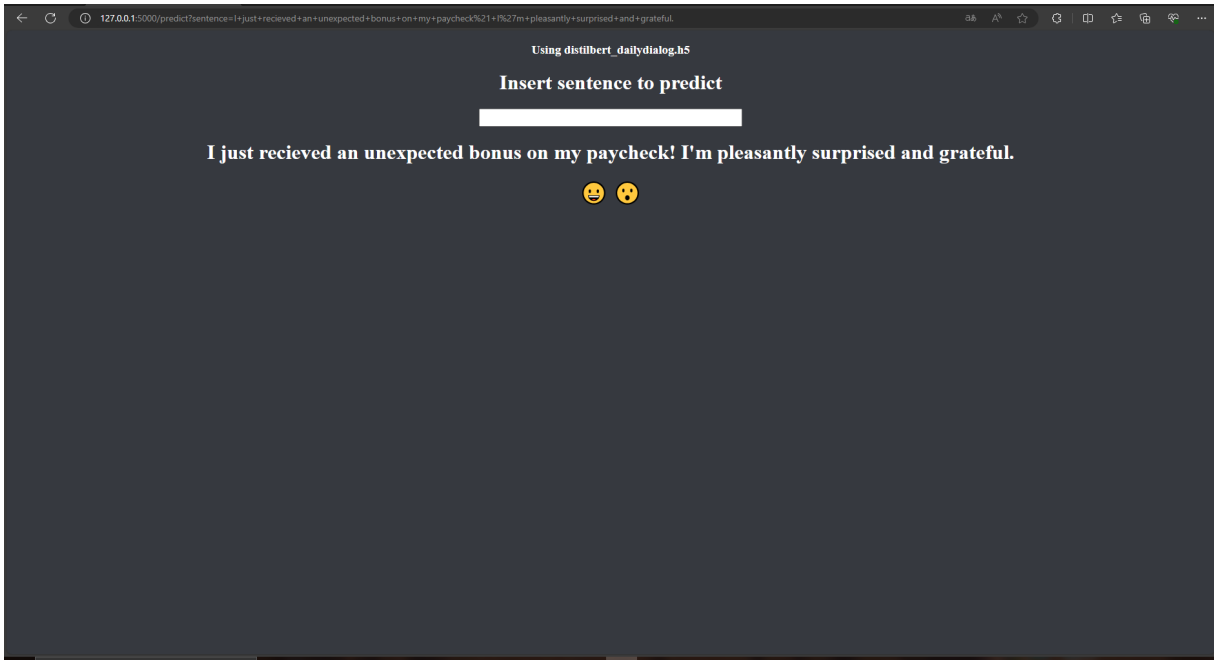


Figure 25: API Web page multiple emotions example

The REST endpoint allows for an alternative way to interact with the solution. Instead of displaying a page, a GET request can be done directly either by a user or an external program with the input sentence as a parameter where the solution will return the prediction as a JSON file containing the level of certainty it has for each emotion.

```
{
    "emotion": [
        ["Anger",0.6039221286773682],
        ["Sadness",0.13900531828403473],
        ["Disgust",0.12019804120063782],
        ["Neutral",0.05487346649169922],
        ["Surprise",0.03991416096687317],
        ["Fear",0.03753994405269623],
        ["Joy",0.004546876531094313]
    ],
    "sentence": "The lack of support from my manager has me furious."
}
```

Listing 2: API response

Chapter 5

Conclusion and Future Work

This chapter is organized into four sections. The first section provides a concise overview of the work involved in the creation of this document. It is followed by a section that outlines the planned objectives for this study and their current status of completion. The subsequent section delves into the limitations encountered during the development of the solution and explores potential areas for future improvement. Lastly, the chapter concludes with some personal reflections on the entire dissertation.

5.1 Conclusion

The departure of an employee has a significant and detrimental impact on the productivity of any business. Moreover, it compounds the issue by incurring substantial costs associated with the recruitment and training of a replacement. Consequently, employee retention emerges as a pivotal concern that every business must prioritize to ensure its ongoing success. In an ever-evolving market landscape, businesses must remain adaptable to maintain a content and stable workforce, thus safeguarding productivity and preventing employee attrition. With the advent of cutting-edge technologies like machine learning and natural language processing, new tools can be innovatively developed to address a wide spectrum of challenges. These technological advancements hold immense potential, especially in the realm of sentiment analysis, offering invaluable assistance in gauging the emotional well-being and satisfaction of employees during their interactions with the HR department.

With this goal in mind, KonkConsulting provided support for the development of a thesis focused on applying Natural Language Processing (NLP) to the human resources domain. The outcome was a comprehensive study and the implementation of a machine learning-powered tool capable of extracting emotions from text, specifically within the context of employee messages.

To achieve this, an extensive review of the current state-of-the-art literature was conducted, encompassing the proposal of various potential machine learning architectures. This stage spanned five months. Subsequently, over the following eight months, a thorough analysis of available data, tools, and techniques was carried out to ensure the creation of an optimal solution whose performance is illustrated in Table 11. Finally, a proof-of-concept tool was designed and implemented, resulting in a functional prototype with satisfactory results. This study and prototype successfully met the company's requirements and are now ready to be included in KonkConsulting's portfolio of solutions.

5.2 Limitations and Future Work

While the finalized solution accomplishes the goal given and can be considered a success, there are several points that can be improved.

Firstly, no dataset was provided, as the solution is intended to operate within a specialized work environment, and conversation dynamics can significantly impact its predictions the lack of data relating to HR-employee interactions imposes a considerable limitation on its effectiveness, considering the specific problem scope. To mitigate this challenge, the solution was designed with a strong emphasis on generalization, prioritizing adaptability over specialization to the provided data. This approach aims to create a model capable of achieving acceptable accuracy across diverse conversational contexts and environments.

The hardware employed for this study's development fell significantly short of optimal performance, being noticeably underpowered. This deficiency led to prolonged training durations and a substantial shortage of GPU memory, which had a severe effect on the study's developmental phase. As a result of these hardware constraints, several limitations had to be imposed, including a preference for faster and more lightweight models over their more performant counterparts. Additionally, a multitude of hyperparameter configurations and architectural choices had to be discarded due to the hardware's incapacity to efficiently handle training or complete it within a reasonable timeframe.

In terms of future work, addressing the most significant bottleneck in the solution involves performing extraction and labeling of a specialized dataset. As a contingency plan, further study of dataset manipulation can be conducted to ensure proper training without discarding so much valuable information. Additionally, the API developed is currently rudimentary and not fully implemented, as it was created solely as a proof of concept. Should this solution be considered for actual use, it will require substantial API development.

5.3 Final Considerations

The development of this dissertation has proven to be my most significant project to date, and its conclusion marks a crucial milestone in my life. The fact that this study was a long-term venture, entirely dependent on my own efforts, demanded that I acquire autonomy and delve into an area where I had little

prior expertise, particularly in the field of machine learning. This experience also enabled me to construct a complete solution from the ground up, something I had never undertaken on such a large scale before. I take great pride in the work accomplished over the past year, which has resulted in the creation of a functional tool with potential real-world applicability.

Bibliography

[1] U. B. of Labor Statistics. *EMPLOYEE TENURE IN 2022*. 2022. url: https://www.bls.gov/news.release/pdf/tenure.pdf (visited on 12/05/2022).

[2] L. Andre. *112 Employee Turnover Statistics: 2023 Causes, Cost Prevention Data*. en. 2021. url: https://financesonline.com/employee-turnover-statistics/ (visited on 02/03/2023).

[3] W. Institute. *2022 Retention Report: How Employers Caused the Great Resignation*. en. url: https://info.workinstitute.com/hubfs/2022RetentionReport/2022RetentionReport-WorkInstitute.pdf (visited on 02/03/2023).

[4] S. Ariella. en. url: https://www.zippia.com/advice/employee-turnover-statistics/ (visited on 02/03/2023).

[5] url: https://www.marketsandmarkets.com/Market-Reports/natural-language-processing-nlp-825.html (visited on 02/03/2023).

[6] K. Linly. *An introduction of NLP and how it's changing the future of HR*. 2021. url: https://www.plugandplaytechcenter.com/resources/introduction-nlp-and-how-its-changing-future-hr/ (visited on 12/05/2022).

[7] I. Future Market Insights. *Natural Language Processing (NLP) Market Size, Share Forecast | US$ 45 billion by 2032*. 2022. url: https://finance.yahoo.com/news/natural-language-processing-nlp-market-070000969.html (visited on 12/05/2022).

[8] J. Franz. *How GPT-3 Unlocks Deeper Listening at inVibe*. 2021. url: https://www.invibe.co/blog/how-gpt-3-unlocks-deeper-listening-at-invibe (visited on 12/02/2022).

[9] T. Brown et al. "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. url: https://papers.nips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

[10] M.-W. Dictionary. *Emotion Definition & Meaning*. url: https://www.merriam-webster.com/dictionary/emotion (visited on 12/18/2022).

[11] W. James. *The Principles of Psychology*. Henry Holt and Company, 1890.

[12] P. Ekman. “Facial expression and emotion”. eng. In: *The American Psychologist* 48.4 (1993), pp. 384–392. issn: 0003-066X. doi: 10.1037//0003-066x.48.4.384.

[13] R. Lazarus and B. Lazarus. *Passion and Reason: Making Sense of Our Emotions*. Oxford University Press, 1994.

[14] M. Schröder, H. Pirker, and M. Lamolle. “First Suggestions for an Emotion Annotation and Representation Language”. en. In: ().

[15] J. Posner, J. A. Russell, and B. S. Peterson. “The circumplex model of affect: an integrative approach to affective neuroscience, cognitive development, and psychopathology”. eng. In: *Development and Psychopathology* 17.3 (2005), pp. 715–734. issn: 0954-5794. doi: 10.1017/S0954579405050340.

[16] R. Plutchik. “A psychoevolutionary theory of emotions”. en. In: *Social Science Information* 21.4–5 (1982), pp. 529–553. issn: 0539-0184, 1461-7412. doi: 10.1177/053901882021004003.

[17] S. Poria et al. “Emotion Recognition in Conversation: Research Challenges, Datasets, and Recent Advances”. en. In: *IEEE Access* 7 (2019), pp. 100943–100953. issn: 2169-3536. doi: 10.1109/ACCESS.2019.2929050.

[18] C. Busso et al. “IEMOCAP: interactive emotional dyadic motion capture database”. en. In: *Language Resources and Evaluation* 42.4 (Dec. 2008), pp. 335–359. issn: 1574-0218. doi: 10.1007/s10579-008-9076-6.

[19] G. McKeown et al. “The SEMAINE Database: Annotated Multimodal Records of Emotionally Colored Conversations between a Person and a Limited Agent”. In: *IEEE Transactions on Affective Computing* 3.1 (Jan. 2012), pp. 5–17. issn: 1949-3045. doi: 10.1109/T-AFFC.2011.20.

[20] S. Poria et al. “MELD: A Multimodal Multi-Party Dataset for Emotion Recognition in Conversations”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 527–536. doi: 10.18653/v1/P19-1050. url: https://aclanthology.org/P19-1050.

[21] Y. Li et al. “DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset”. In: *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, Nov. 2017, pp. 986–995. url: https://aclanthology.org/I17-1099.

[22] S.-Y. Chen et al. “EmotionLines: An Emotion Corpus of Multi-Party Conversations”. In: arXiv:1802.08379 (May 2018). arXiv:1802.08379 [cs]. doi: 10.48550/arXiv.1802.08379. url: http://arxiv.org/abs/1802.08379.

[23] A. Chatterjee et al. “SemEval-2019 Task 3: EmoContext Contextual Emotion Detection in Text”. In: *Proceedings of the 13th International Workshop on Semantic Evaluation*. Minneapolis, Minnesota, USA: Association for Computational Linguistics, June 2019, pp. 39–48. doi: 10.18653/v1/S19-2005. url: https://aclanthology.org/S19-2005.

[24] N. Alswaidan and M. E. B. Menai. “A survey of state-of-the-art approaches for emotion recognition in text.” In: *Knowledge Information Systems* 62.8 (2020), pp. 2937–2937–2987. issn: 0219-1377. doi: 10.1007/s10115-020-01449-0.

[25] K. R. Scherer and H. G. Wallbott. ““Evidence for universality and cultural variation of differential emotion response patterning”: Correction”. In: *Journal of Personality and Social Psychology* 67.1 (1994), pp. 55–55. issn: 1939-1315. doi: 10.1037/0022-3514.67.1.55.

[26] C. Strapparava and R. Mihalcea. “SemEval-2007 Task 14: Affective Text”. In: *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*. Prague, Czech Republic: Association for Computational Linguistics, June 2007, pp. 70–74. url: https://aclanthology.org/S07-1013.

[27] S. Mohammad et al. “SemEval-2018 Task 1: Affect in Tweets”. In: *Proceedings of the 12th International Workshop on Semantic Evaluation*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 1–17. doi: 10.18653/v1/S18-1001. url: https://aclanthology.org/S18-1001.

[28] H. Izadkhah. “Detection of multiple emotions in texts using a new deep convolutional neural network”. en. In: *2022 9th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS)*. Bam, Iran, Islamic Republic of: IEEE, 2022, pp. 1–6. isbn: 978-1-66547-872-4. doi: 10.1109/CFIS54774.2022.9756494. url: https://ieeexplore.ieee.org/document/9756494/.

[29] A. G. Shahraki and O. R. Zaïane. “Lexical and learning-based emotion mining from text”. In: *Proceedings of the International Conference on Computational Linguistics and Intelligent Text Processing*. 2017.

[30] P. Bojanowski et al. “Enriching Word Vectors with Subword Information”. In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146. doi: 10.1162/tacl_a_00051.

[31] J. Pennington, R. Socher, and C. Manning. “GloVe: Global Vectors for Word Representation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. doi: 10.3115/v1/D14-1162. url: https://aclanthology.org/D14-1162.

[32] E. Batbaatar, M. Li, and K. H. Ryu. “Semantic-Emotion Neural Network for Emotion Recognition From Text”. In: *IEEE Access* 7 (2019), pp. 111866–111878. issn: 2169-3536. doi: 10.1109/ACCESS.2019.2934529.

[33] T. Mikolov et al. “Efficient Estimation of Word Representations in Vector Space”. In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2013. url: http://arxiv.org/abs/1301.3781.

[34] A. Agrawal, A. An, and M. Papagelis. “Learning Emotion-enriched Word Representations”. In: *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 950–961. url: https://aclanthology.org/C18-1081.

[35] C. Liu et al. “Individual Emotion Recognition Approach Combined Gated Recurrent Unit With Emoticon Distribution Model”. In: *IEEE Access* 9 (2021), pp. 163542–163553. issn: 2169-3536. doi: 10.1109/ACCESS.2021.3124585.

[36] A. F. Adoma, N.-M. Henry, and W. Chen. “Comparative Analyses of Bert, Roberta, Distilbert, and Xlnet for Text-Based Emotion Recognition”. en. In: *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*. Chengdu, China: IEEE, Dec. 2020, pp. 117–121. isbn: 978-1-66540-503-4. doi: 10.1109/ICCWAMTIP51612.2020.9317379. url: https://ieeexplore.ieee.org/document/9317379/.

[37] J. Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: arXiv:1810.04805 (May 2019). arXiv:1810.04805 [cs]. doi: 10.48550/arXiv.1810.04805. url: http://arxiv.org/abs/1810.04805.

[38] Y. Liu et al. “RoBERTa: A Robustly Optimized BERT Pretraining Approach”. In: arXiv:1907.11692 (July 2019). arXiv:1907.11692 [cs]. doi: 10.48550/arXiv.1907.11692. url: http://arxiv.org/abs/1907.11692.

[39] V. Sanh et al. “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”. In: arXiv:1910.01108 (Feb. 2020). arXiv:1910.01108 [cs]. doi: 10.48550/arXiv.1910.01108. url: http://arxiv.org/abs/1910.01108.

[40] Z. Yang et al. “XLNet: Generalized Autoregressive Pretraining for Language Understanding”. In: arXiv:1906.08237 (Jan. 2020). arXiv:1906.08237 [cs]. doi: 10.48550/arXiv.1906.08237. url: http://arxiv.org/abs/1906.08237.

[41] A. Vaswani et al. “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017. url: https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[42] S. Hochreiter and J. Schmidhuber. “Long Short-term Memory”. In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. doi: 10.1162/neco.1997.9.8.1735.

[43] X. Ying. “An Overview of Overfitting and its Solutions”. en. In: *Journal of Physics: Conference Series* 1168 (Feb. 2019), p. 022022. issn: 1742-6588, 1742-6596. doi: 10.1088/1742-6596/1168/2/022022.

[44] L. Li et al. “Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization”. In: arXiv:1603.06560 (June 2018). arXiv:1603.06560 [cs, stat]. doi: 10.48550/arXiv.1603.06560. url: http://arxiv.org/abs/1603.06560.