UNIVERSITY OF PADOVA

DEPARTMENT OF MATHEMATICS TULLIO LEVI-CIVITA

MASTER THESIS IN BIG DATA MANAGEMENT AND ANALYTICS

MULTILINGUAL FINE-GRAINED SENTIMENT TEXT CLASSIFICATION OF WEB CONTENT FOR ADVERTISING TECHNOLOGY APPLICATIONS

SUPERVISOR
PROFESSOR NICOLÒ NAVARIN
UNIVERSITY OF PADOVA

CO-SUPERVISOR
GIOVANNI VEDANA
ANONYMISED

MASTER CANDIDATE
NICOLE ZAFALON KOVACS

SEPTEMBER 1ST 2023

Abstract

Programmatic advertising involves matching publishers' web page content with advertisers' strategies, usually by filtering out content topics based on audience interests. However, this approach can limit audience reach and overlook content nuances. Sentiment analysis comes into place to enhance content understanding by categorizing web pages into sentiment groups. While existing research often focuses on the "positive", "negative", and "neutral" labels, a broader range of emotion categories can provide more details, better aligning content with advertisers' goals. This study investigates sentiment classification models that use fine-grained emotion categories to classify the content of publishers' web pages. Considering the multilingual nature of the business, this research also explores and compares language-agnostic models. Several models were trained using the GoEmotions dataset, composed of 58k English Reddit comments annotated by humans using 28 emotion categories. These models were assessed across various sentiment groupings and evaluated on manually annotated real-world web page texts. The final proposed emotion groups encompass six categories: "repudiation", "sadness", "neutral", "curiosity", "appreciation", and "positive experience". Among the models compared, transformer models, particularly BERT-large, exhibited superior performance. The best English-only model achieved a weighted F1-score of 44% on the annotated web data. Language-agnostic models showed lower metrics on Italian texts but were comparable to English-only models for English text. The leading language-agnostic model, Multilingual BERT, achieved a weighted F1-score of 37% on the same data translated into Italian. The study achieved promising outcomes across the six emotion groups, surpassing the traditional "positive," "negative," and "neutral" categories. Additionally, the evaluation of multilingual models demonstrated their applicability to multiple languages despite being trained solely on English data.

Contents

Listing of figures

Listing of tables

Listing of acronyms

AdTech	Advertising technology
BERT	Bidirectional Encoder Representations from Transformers
CNN	Convolutional neural network
FNN	Feedforward neural network
GRU	Gated recurrent unit
LABSE	Language-Agnostic BERT Sentence Encoder
LSTM	Long-short term memory neural network
MLP	Multilayer perceptron
NLP	Natural Language Processing
mBERT	Multilingual BERT
mUSE	Multilingual Universal Sentence Encoder
NN	Neural network
RNN	Recurrent neural network
RTB	Real-time bidding
SVC	Support vector classification
SVM	Support vector machine
TF-IDF	Term-Frequency-Inverse Document Frequency
USE	Universal Sentence Encoder
WASSA	Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis

1 Introduction

This chapter presents a summary of the work done in this thesis. The following sections describe background knowledge and state of the art of the field, the problem to be solved, purpose of the study, methods with relevant execution details, results, and their interpretation.

1.1 Background and need

The growth of digital advertising has been driven by the Internet's expansion over the past two decades, bringing to the spotlight the usage of Programmatic Advertising, which uses Real-time Bidding (RTB) to instantly trade ad spaces when a user visits a web page, allowing advertisers (buyers of ad space) to target specific users, and effectively resolving issues of unsold inventories. Machine learning algorithms play an important role in this context by enabling a comprehensive understanding of visitors' characteristics based on their data, facilitating precise targeting of user groups, and improving content matching between publishers (owners of the ad space) and users.

Sentiment text classification is often approached as a binary classification problem of positive and negative sentiments, sometimes including a "neutral" class. However, more detailed insights can be gained from the data by categorizing emotions into finer-grained sentiments. Various emotion taxonomies have been proposed for this purpose. Paul Ekman's model [12] identifies six basic emotions (happiness, sadness, fear, anger, surprise, and disgust) and emphasizes their universality across cultures based on recognizable facial expressions. Robert Plutchik [13] expanded on Ekman's work with the "Wheel of Emotions" which organizes emotions in a circular diagram, including primary emotions and secondary emotions. In contrast, Russell [14] introduced a continuous two-dimensional circumplex model, plotting emotions in a space defined by valence (positive to negative) and arousal (calm to excited), providing a more nuanced understanding of emotions. More recent approaches in psychology use computational techniques to capture the "semantic space" of emotion and have identified various distinct emotional experiences, classifying up to 28 distinct emotion categories [3, 15].

Amongst the most used machine learning models in text classification and sentiment analysis, the following approaches could be mentioned: Logistic Regression [16, 17, 18, 19, 20], Support Vector Machines [16, 17, 18, 19, 20, 21, 22, 23], Feedforward Neural Networks [24, 25], Convolutional Neural Networks [17, 24, 26, 27], Recurrent Neural Networks (including Long Short-Term Memory and Gated Recurrent Unit networks) [19, 23, 24, 26, 27, 28], and Transformers [11, 23, 29, 30, 31, 32, 33].

In order to represent the text data in the classification models, simpler approaches were known to be applied, such as Bag of Words and Term Frequency-Inverse Document Frequency. More recent techniques involve the use of deep learning models that learn the best way to represent words using vectors: such an approach is known as word embeddings. A few examples of word embeddings are Word2Vec [34], GloVe [35], fastText [36], BERT [37], variations of BERT (e.g. distilBERT [38] and RoBERTa [39]), Universal Sentence Encoder [40] and XL-NET [41]. Some embeddings are specific to the English language, but the field of other languages and multilingual embedders has expanded.

Although a lot of research has been done in sentiment analysis over the years, few works can be found regarding fine-grained emotion labels. This thesis aims to work with fine-grained sentiment analysis in the programmatic advertising field.

1.2 Statement of the problem

The main focus of this thesis is the creation of a granular sentiment-based text classification model to be applied in texts from web pages, in the context of digital advertising. Its final application involves presenting content filters based on text sentiment to advertiser campaigns, in order to more finely select the targeted content and audience.

The model is defined as multi-class classification machine learning model, preceded by natural language processing (NLP) pre-processing steps. The model's labels are defined as detailed sentiment groups, with a finer granularity than just positive, negative, and neutral labels. A study on the granularity of the sentiment labels is conducted with the aim of providing a more detailed classification about the text while avoiding high ambiguity between the labels and maintaining a balance between positive and negative sentiment groups. This enables Anonymised to provide more information to advertisers regarding the publishers' content while keeping their choices in a small and meaningful range of emotion labels. This information will be used by the advertisers to determine at which types of web pages they should be advertising.

1.3 Purpose of the study

The research of a text classification model based on granular sentiment groups is intended for Anonymised, a company in the field of digital advertising focused on data privacy solutions that bridge contacts between publishers and advertisers. The content of this work will be deployed by Anonymised with the aim of providing sentiment labels for content of publishers' websites, in order to better categorize the content presented in the ad spaces offered to advertisers. The final goal is to better match publisher content with advertiser strategies.

While the main focus of this work is on English-based text, due to the international aspect of Anonymised's clientele, an additional objective is to research language-agnostic models in order to effectively work in other languages. It was a business requirement to have one model for multiple languages instead of one model for each language, in order to simplify the maintenance of the model and pipeline. The evaluation of such models was performed using text translated into Italian, this choice of language was prioritized as a business decision. Therefore, a section of this thesis is dedicated to analyzing results using Italian text.

In summary, this thesis aims to answer the following research questions:

1. How well can web page texts be classified into more granular sentiment categories?

2. What level of sentiment granularity should be employed?

3. Is it feasible to employ a language-agnostic model for this task? If so, how does it compare to a single language model?

1.4 Methodology

The main dataset used to train and evaluate the models was defined to be Google's GoEmotions dataset [11], due to its extensive number of sentiment categories as well as its large number of data points. Composed of 58k annotated English language Reddit comments, it was constructed with 28 fine-grained emotion categories, that include 12 positive, 11 negative, 1 "neutral", and 4 "ambiguous" emotion categories. Its training and validation splits are already predefined by the authors, therefore they were used as is.

In addition to GoEmotions, another dataset was experimentally used for training the models. The WASSA 2021 shared task dataset [42] presents 1.8k long essays (ranging from 300 to 800 characters) annotated with 6 emotion labels. This dataset was chosen due to its similarity in length to the data that the models will be effectively executed on, as described next.

The web page text content (also referred to as "web data" in this work) for which the models were designed consists of a range of different text formats and different subjects. For instance, it ranges from song lyrics, recipes, and tutorials to news articles about sports and politics. The texts could be short (just a few words) but are generally long with several paragraphs.

The web data is collected by a scraper developed by the company, which extracts the title and main text of the web pages, removes punctuation, and turns characters into lowercase. However, in the context of sentiment analysis, text punctuation and uppercase characters could be potential carriers of emotional information. By preserving these elements within textual data, we may gain insights into their ability to convey intensified sentiments. For instance, the inclusion of exclamation marks at the end of the text or the adoption of an all-uppercase writing style could potentially signify heightened emotional expressions. Despite the potential significance of such modifications, altering the web scraper code was deemed outside the scope of this project. As a result, we identify this aspect as a promising avenue for future work.

For the purpose of evaluating the models' performance in the web data, 200 random web page texts were collected from the company's database and manually annotated by the author and her supervisor from Anonymised. Some texts were as short as 6 words, but others reached 20,208 words. The 200 data samples had an average of 888 words and a standard deviation of 1832. The texts were labeled according to the 28 GoEmotions categories, and later summarized into the 6 sentiment group selection (as described in section 4.2), subsequently summarized into one label to evaluate single-label classifiers. These texts were then translated into Italian with the aim of assessing the models' performance in a multilingual scenario.

The final six emotion labels were chosen by grouping the fine-grained emotions according to both their correlations and their semantic meaning. The number of groups was chosen with the aim of balancing positive and negative emotion groups. This decision was made alongside the company's interests. The final groups were defined as follows:

Repudiation	Anger, annoyance, disapproval, disgust
Sadness	Disappointment, embarrassment, fear, grief, nervousness, remorse, sadness
Neutral	Neutral
Curiosity	Confusion, curiosity, realization, surprise
Appreciation	Approval, gratitude, admiration, pride, caring, desire, love
Positive experience	Amusement, excitement, joy, optimism, relief

The evaluation metrics obtained for evaluating the models were mainly accuracy across all labels, averaged F1-score, and weighted averaged F1-score. The precision and recall per label were also evaluated in order to ensure that the labels' metrics were balanced out. When needed, the F1-score metrics were preferred since they balance out precision and recall.

For the sake of incrementally experimenting and studying different techniques, the experiments were divided into three main categories: firstly, non-neural network models were evaluated on high-level label categories (positive, negative and neutral) using different feature representations for the text input. Then the models with the highest metrics were re-evaluated in the set of 28 emotion categories and compared to neural networks approaches. Finally, the models with the highest metrics were then chosen and compared with several Transformers approaches based on the 6 sentiment groups chosen.

1.5 Results and findings

In the first set of experiments, the models and feature representations with the highest weighted F1-scores were a Linear Support Vector Classifier using distilBERT embeddings and a Logistic Regression using Multilingual Universal Sentence Encoder embeddings. The embeddings feature representations outperformed count-based and feature-based approaches, probably because such embeddings obtain their values based on the context of the whole sentence and not only on the choice of words in each sentence. Furthermore, LinearSVC and Logistic Regression outperformed the other model choices probably due to their robustness against high-dimensional data.

In the second set of experiments, the aforementioned models were re-evaluated in the new target labels. They were nevertheless outperformed by almost all neural network approaches, with highlights to a Gated Recurrent Unit recurrent neural network and a fine-tuned distilBERT classifier. This was probably due to the fact that neural networks are able to better identify complex patterns in text data.

The third set of experiments showed higher weighted F1-scores for the transformers RoBERTa, BERT, XLNET, and LABSE. However, since RoBERTa and XLNET require more extensive computational resources, BERT and LABSE were chosen as the best cost-benefit models. Multi-label experiments were also performed in the data set labeled with 6 sentiment groups, yet they presented lower F1-scores and accuracies when compared to the single-label models.

These models were later evaluated on the manually annotated texts extracted from the web, and the BERT-large model presented the highest weighted F1-score. When analyzing their confusion matrices though, it became clear that the model was predicting most data points as neutral or appreciation. After calibrating the prediction probabilities, the predictions were better distributed, while the weighted F1-score remained the same.

Finally, the multilingual models (Multilingual BERT, LABSE, and neural network with Multilingual Universal Sentence Encoder embeddings) trained on English-only data were evaluated on the manually annotated web data translated into Italian. Such models showed worse results when compared to the results on English data, although calibrating the prediction probabilities slightly increased their metrics.

Since the manually annotated web data was used before revisions on the models and not as a final test set, it is important to note that the results reported could be overfitting, which means that we could be focusing too much on the specific web dataset, and it would not generalize well to unseen data. Further evaluations on new data need to be taken in place in order to assess if the model is able to generalize well to such new data.

The thesis is structured as follows. Chapter 2 presents with more detail previous works related to this thesis: firstly an overview of text classification solutions is introduced, with insights into state-of-the-art approaches, including research on multilingual models. It is then followed by emotion taxonomies and available datasets in the literature. Subsequently, the need for this research is presented.

Chapter 3 presents a detailed description of the problem to be solved in this work, while Chapter 4 describes the datasets used for model training and evaluation, followed by the methodology, results and findings of experiments regarding classifications for three different target types: high-level "positive, negative and neutral" models, fine-grained emotion labels from the GoEmotions dataset, and middle-level emotion groups. The chapter is concluded with web data and a multilingual evaluation of the models chosen. Finally, concluding remarks and future developments are reported in Chapter 5.

2
Background

The sentiment analysis process encompasses several steps, including preprocessing, feature extraction, and classification. During preprocessing, the raw text data is usually cleaned to remove special characters, numbers, and stop words. The text is subsequently transformed into features using techniques such as Term-Frequency-Inverse Document Frequency (TF-IDF) or word embeddings, as described in Section 2.1.5. The next steps entail classifying the processed text into sentiments using machine learning methods, such as logistic regression, naïve Bayes, support vector machines, or deep learning models like convolutional and recurrent neural networks, and transformers.

This chapter provides some background knowledge on the topics mentioned above and further provides an overview of the recent progress in the field of sentiment analysis. It also explains about programmatic advertising, the specific field of application of the sentiment analysis models in this work, and web scraping, the method used to extract the data.

2.1 Preliminaries

2.1.1 Programmatic advertising

The field of digital advertising has grown rapidly in the past two decades due to the exponential expansion of the Internet. Initially, digital advertising involved advertisers (organizations looking to promote their products or services online) purchasing ad spaces directly from publishers (website owners or content providers who seek to monetize their online inventory by offering ad space on their platform), much like traditional magazine ads. Advertisers and publishers would reach agreements for specific time periods on particular websites. Consequently, these ads were indiscriminately shown to all users browsing a publisher's web page, without considering user preferences.

This method proved highly inefficient for two primary reasons. Firstly, the majority of the audience exposed to these ads was uninterested in the promoted products, as potential customers constituted only a small portion of the overall web page visitors. Secondly, the rapid proliferation of websites outpaced the number of companies willing to advertise, leading to numerous unsold ad spaces due to the slow manual trading process.

To address these challenges, an automated solution was developed to manage the digital advertising market, eliminating the need for human negotiations. This solution is known as Programmatic Advertising. Programmatic advertising incorporates various factors to ensure benefits for both advertisers and publishers. Through Real-time Bidding (RTB), trades occur instantaneously when a user clicks on a web page, involving companies interested in targeting that specific user. This approach effectively resolves the issues of unsold inventories and enhances targeting precision [9].

Within this context, the usage of machine learning algorithms brings value as they enable a comprehensive understanding of visitors' characteristics based on their data, thereby facilitating precise targeting of specific user groups. Additionally, machine learning proves valuable in characterizing publishers' content, thereby improving content matching with users' preferences, and further enhancing the overall advertising process.

2.1.2 Web scraping

Web scraping is an automated technique used to extract data from websites that could be later persisted, e.g. in a database. This process involves employing specialized software tools, scripts, or algorithms to retrieve specific information, such as text, titles, images, or other data, from various web pages. The web scraping process can be broken down into two steps: acquiring web resources and then extracting the desired information from the acquired data. Once the web data is downloaded, the extraction process parses, re-formats, and organizes the data into a structured format [43].

In the context of this thesis, Anonymised's web scraper serves the purpose of extracting titles and main content of web articles from publishers who have authorized its usage. The goal is to classify the page content effectively using the extracted information. The scraper's output is ultimately the input of the models developed in this work.

2.1.3 Emotion taxonomies

Sentiment classification is usually formulated as a two-class classification problem, positive and negative, with eventually a third class being "neutral". However, in order to obtain better insights from the data, it is interesting to classify it into finer-grained sentiments. This section describes some of the most popular emotion taxonomies that have been used for this task.

Paul Ekman developed a widely recognized taxonomy based on cross-cultural research [12]. His model identified six basic emotions: happiness, sadness, fear, anger, surprise, and disgust. He focused on the universality of facial expressions associated with these emotions, suggesting that they have a biological basis and are recognizable across different cultures.

Plutchik expanded on Ekman's basic emotions and presented the "Wheel of Emotions" model [13]. He organized emotions into a circular diagram, showing the relationships between emotions and their intensities. Plutchik's model included 4 pairs of contrasting primary emotions: joy-sadness, trust-disgust, fear-anger, and surprise-anticipation, as well as secondary emotions resulting from combinations of the primary ones, as illustrated in Figure 2.1.

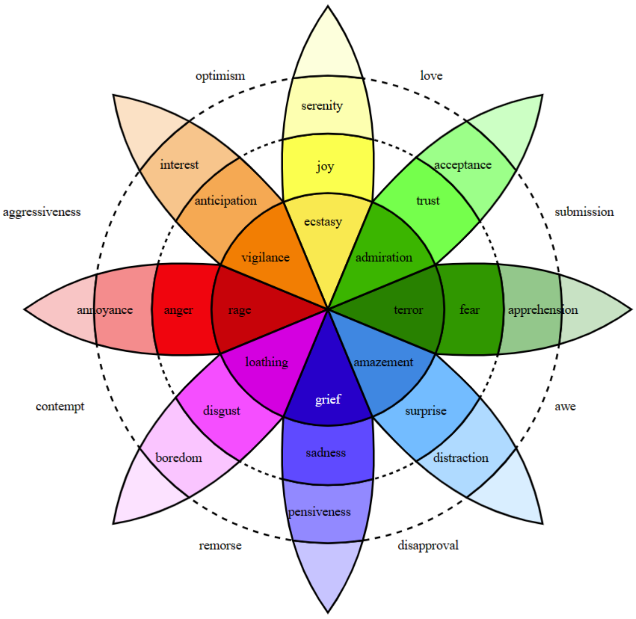


Figure 2.1: Robert Plutchik's Wheel of Emotions [1].

Oppositely to the discrete emotion taxonomies aforementioned, Russell (1980) [14] introduced a continuous two-dimensional model (also known as the circumplex model) that plotted emotions on a space defined by valence (ranging from pleasant to unpleasant), referring to the positive and negative degree of emotion, and arousal (ranging from calm to excited), referring to the intensity of emotion. The circumplex model is illustrated in Figure 2.2. This model allowed for a more nuanced understanding of emotions and their interplay in emotional experiences.

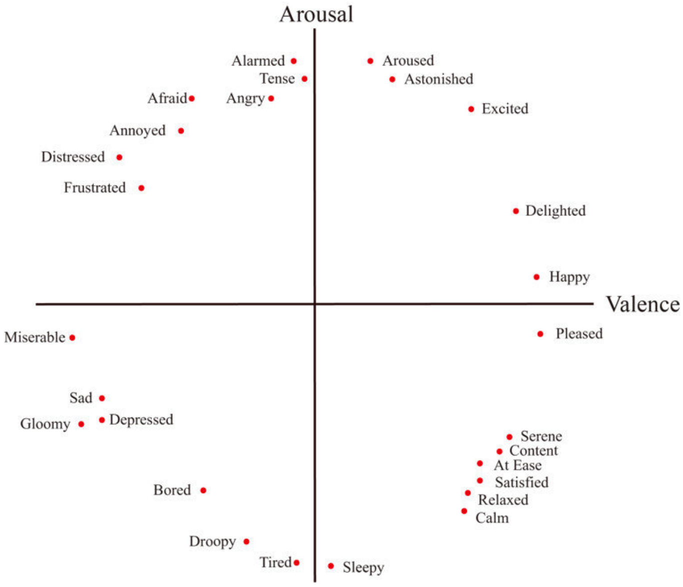


Figure 2.2: Russell's circumplex model [2].

Recent advances in the field of psychology have introduced novel approaches to capture the intricate "semantic space" of emotions [44] by studying the distribution of emotion responses to a diverse range of stimuli via computational techniques. Studies guided by these principles have reported several varieties of emotions labels. For example, Cowen and Keltner identified 27 varieties of emotional experience conveyed by short videos (Figure 2.3 illustrates most of them) [3]. Furthermore, 13 emotion categories were identified conveyed by music [45], 28 by facial expressions [15], 12 by speech prosody [46], and 24 by nonverbal vocalization [47].

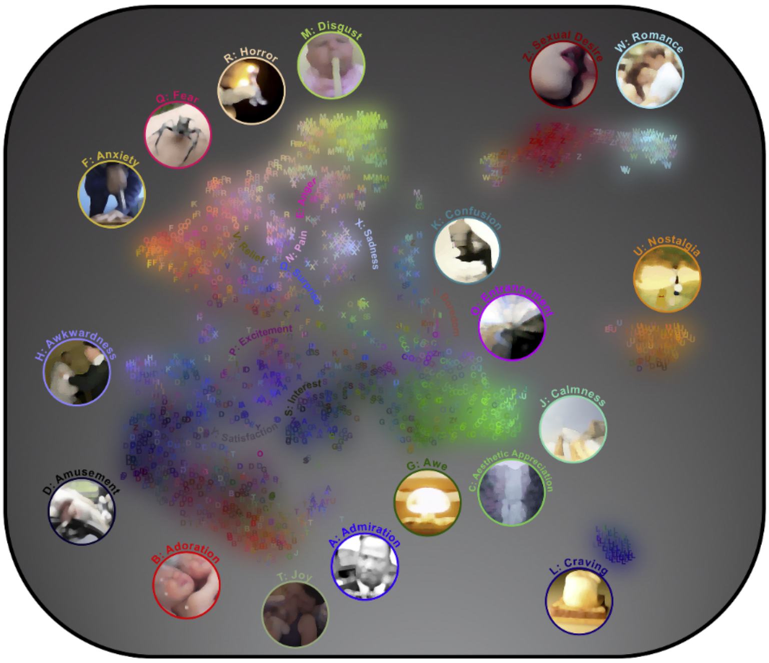


Figure 2.3: Videos mapped along 27 categorical judgment dimensions of reported emotional experience [3] .

2.1.4 Classifiers

Multi-label vs multi-class classification

In the context of machine learning and classification tasks, multi-label and multi-class classification are two different approaches to handling datasets where the target class has more than two possible outcomes for each data point [48].

Multi-class classification is defined as the case where each data point is classified into one and only one class from a set of multiple classes. In other words, each instance in the dataset is assigned to one exclusive category. For example, consider a dataset of animals categorized as mammals, birds, reptiles, and amphibians. Each animal in the dataset can only belong to one of these four classes.

On the other hand, multi-label classification is defined as the case where each data point can belong to multiple classes simultaneously. In this scenario, a data point can be associated with more than one label or category. For instance, consider a news article classification task, where an article may belong to multiple topics such as movies, sports, and entertainment. Multi-label classification algorithms aim to predict the presence or absence of each label independently for each data point.

Support Vector Machines

Support Vector Machines [49] (SVMs) are powerful supervised machine learning algorithms used for classification and regression tasks. SVM maps its data points to points in space where each feature represents an axis. In the case of classification, the fundamental idea behind SVMs is to find an optimal line or hyperplane that separates the data mapped into classes, by maximizing the margin of such line between different classes while still correctly classifying the data points. The margin is the distance between the hyperplane and the nearest data points from each class, known as support vectors.

Kernel functions can be used in SVMs helping to separate classes that wouldn't be separable otherwise, by transforming the data into a more suitable space. Some commonly used kernel functions include polynomial kernels, radial basis function kernels, and sigmoid kernels.

Linear Support Vector Machine (or LinearSVC specifically for Classification) is a specific implementation of SVMs for linearly separable data. It works with a linear kernel and is particularly efficient for large-scale datasets. LinearSVC has been widely used in text classification tasks.

For two classes, Linear SVM is defined as follows: given a training dataset of n points in the form $(x_1, y_1), ..., (x_n, y_n)$, where y_i indicates to which class the point x_i belongs (-1 or 1), the algorithm's objective is to find the maximum margin hyperplane that divides the points x_i between the two classes.

Any hyperplane can be defined as $w^T x - b = 0$, where w is the normal vector to the hyperplane (not necessarily normalized). In the case that the training data is linearly separable, it is possible to obtain two different hyperplanes that separate the two classes with the maximum distance between them. The region between these two hyperplanes is called the "margin", and the hyperplane that lies halfway between them is the maximum-margin hyperplane. With a normalized dataset, these hyperplanes can be described by the equations $w^T x - b = 1$ (describing that any point that falls on or above this boundary belongs to class 1) and $w^T x - b = -1$ (describing that any point that falls on or above this boundary belongs to class -1). These definitions are illustrated in Figure 2.4.

Geometrically, the distance between the two hyperplanes, i.e. the margin, is $\frac{2}{||w||}$. Therefore, in order to maximize the margin we need to minimize $||w||$. However, we also need to make sure that all points fall into the respective side of the margin, that is: for each i, we need either $w^T x_i - b \geqslant 1$ (if $y_i = 1$), or $w^T x_i - b \leqslant -1$ (if $y_i = -1$).

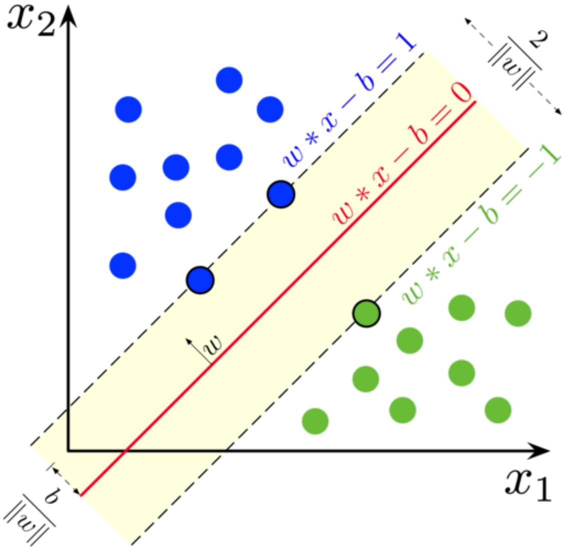


Figure 2.4: Example of a Linear SVM trained with two classes. The red line represents the maximum-margin hyperplane. Blue points represent data belonging to class 1, green points represent data belonging to class -1. The yellow zone represents the margin [4].

Thus, the problem to be solved by Linear SVM is defined as:

$$\begin{aligned} \min_{w,b} \quad & \|w\|_2^2 \\ \text{s.t.} \quad & y_i(w^T x_i - b \geqslant 1), \qquad \forall i \in \{1, ..., n\} \end{aligned} \tag{2.1}$$

Neural Networks

Neural networks are machine learning models composed of interconnected artificial neurons, organized in layers, designed to process complex patterns and make predictions from input data.

The architecture of a neural network consists of three main types of layers: the input layer, one or more hidden layers, and the output layer. The input layer receives raw data, which is then processed through the hidden layers using interconnected neurons that transform the data through mathematical operations. The output layer provides the final results of the model's computation, representing its predictions or classifications [50].

A multilayer perceptron (MLP) is a specific type of neural network, also known as a feedforward neural network, where neurons are arranged in layers, and information flows only in one direction—from input to output. Each neuron takes input values, multiplies them by corresponding weights, and passes the result through an activation function to produce an output, as shown in Equation 2.2, where a represents the neuron's activation, ϕ is the activation function, w_j are the weights, x_j are the neuron's inputs, and b is the bias. The hidden layers in an MLP allow the model to learn hierarchical representations of the data, making it effective in capturing complex relationships and patterns. Figure 2.5 illustrates an MLP, where each neuron (circle) has a variation of Equation 2.2.

$$a = \phi\left(\sum_j w_j x_j + b\right) \tag{2.2}$$

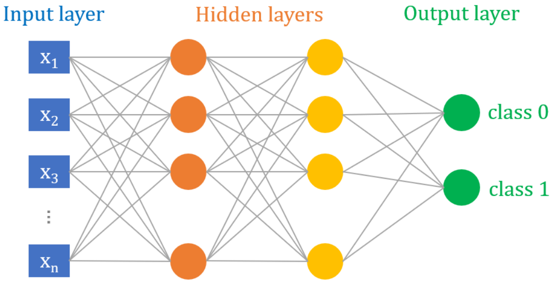


Figure 2.5: Multilayer perceptron, or feedforward neural network.

Neural networks excel in learning complex and nonlinear patterns from large datasets. The advent of deep learning, which involves neural networks with multiple hidden layers, has significantly boosted the performance of these models, enabling them to learn hierarchical representations of data and tackle even more challenging and sophisticated problems.

Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a class of deep learning models commonly used for image recognition tasks. They are designed to automatically learn and extract hierarchical patterns or features from images through an operation called convolution. CNNs consist of multiple layers, including convolutional layers (that scan over the data to capture local patterns and features), pooling layers (that reduce the spatial dimensions of the feature maps generated by the convolutional layers), and fully connected layers (that combine the learned features).

Although CNNs were originally developed for image data, their architectural principles can be adapted for other types of data as well. For example, 1D CNNs can be used for processing sequential data like time series or text data. In text processing, the 1D CNN employs one-dimensional convolutional filters to slide over the text, capturing local patterns, word combinations, or phrases [51]. Figure 2.6 shows an example of a CNN applied to a sentence.

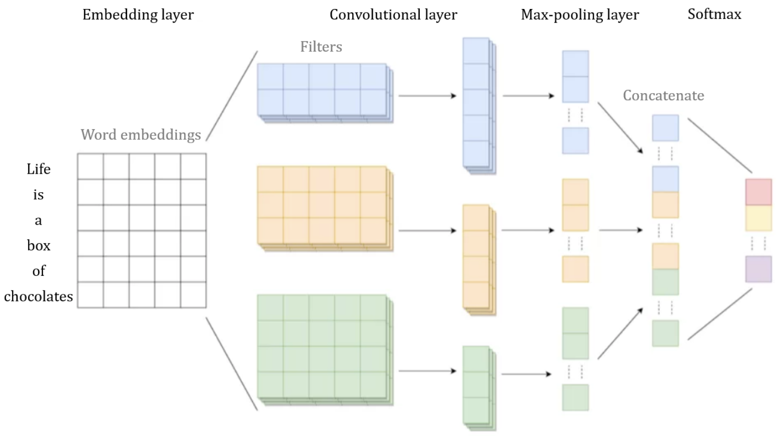


Figure 2.6: Example of a CNN architecture for text classification [5].

Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a class of neural networks designed to handle sequential data, making them particularly useful for text data, since the word order in text is usually important. In an RNN, each word in a text sequence is processed one at a time, and the network maintains an internal hidden state that acts as its memory of previous inputs. This allows the RNN to capture dependencies between words and contextual information in the text, enabling it to make predictions or generate outputs based on the entire sequence. Figure 2.7 illustrates the architecture of an RNN. The hidden states h_t at each time step t are defined as:

$$h_t = \phi(Ux_t + Wh_{t-1}), \tag{2.3}$$

where ϕ is an activation function, U is a weight matrix, x_t in the input at time step t, and W is a hidden state matrix (that weights the value of h_{t-1}).

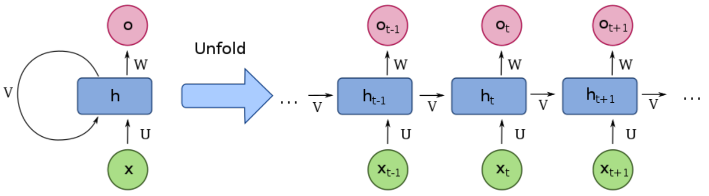


Figure 2.7: Architecture of an unrolled RNN [6].

Long Short-Term Memory (LSTM) networks [52] and Gated Recurrent Unit (GRU) networks [53] are variations of RNNs that can hinder traditional RNNs from effectively capturing long-range dependencies in text data. LSTM and GRU architectures incorporate gating mechanisms that allow the network to retain and control the flow of information through the hidden states. This helps RNNs remember relevant information over longer sequences, making them more effective for handling long texts and maintaining context over extended periods. Figure 2.8 illustrates the diagrams for a simple RNN, an LSTM, and a GRU.

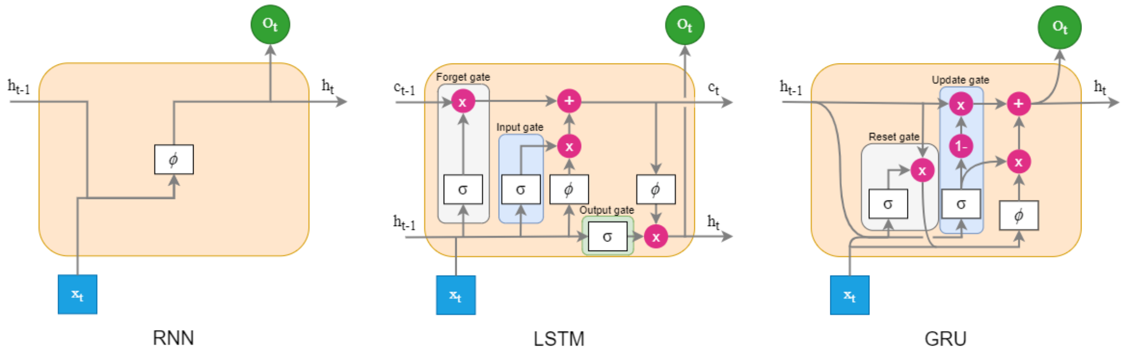


Figure 2.8: Diagram of an RNN, an LSTM and a GRU. x_t is the input at time step t, O_t is the output at time step t, h_t and h_{t-1} are the hidden states at time steps t and $t-1$, c_t and c_{t-1} are the cell states at times t and $t-1$, ϕ represents the activation function, and σ represents selector vectors.

Long short-term memory

LSTMs use a concept called cell state (c_t), which is passed from the input to the output of each cell. It represents the long-term memory part of the LSTM. Furthermore, LSTMs have three gates: input gate, forget gate, and output gate. These gates work as filters and control which information to keep or discard throughout the flow of information between the cells.

The forget gate determines which information should be discarded. It uses a sigmoid function (σ) to represent the amount of information to be kept (from 0 to 1). Its output is defined as a function of the current output (x_t), the hidden state of the previous time step (h_{t-1}), and a bias (b), as shown in Equation 2.4

$$f_t = \sigma(W_{f,x} + x_t + W_{f,h}h_{t-1} + b_f) \tag{2.4}$$

The input gate determines which information should be added to the cell state. In this case, it also uses a sigmoid function (σ), but to decide which values to keep. It is defined as follows:

$$i_t = \sigma(W_{i,x} + x_t + W_{i,h}h_{t-1} + b_i) \tag{2.5}$$

The output gate of the LSTM determines which information from the cell state should compose the output. Described by Equation 2.6, it is responsible for the short-term memory part of the LSTM.

$$o_t = \sigma(W_{o,x} + x_t + W_{o,h}h_{t-1} + b_o) \tag{2.6}$$

The cell state is finally updated using the forget and input gates, as follows:

$$c_t = (f_t \circ c_{t-1} + i_t \circ \phi(h_{t-1}))\,, \tag{2.7}$$

where ϕ represents the activation function $\tanh$. Afterward, the hidden state of the current time step (h_t) is defined through the output gate and a tanh function that limits the cell state between -1 and 1.

$$h_t = o_t \circ \tanh c_t \tag{2.8}$$

Gated Recurrent Unit

Contrary to the LSTM, the GRU does not have a cell state and is defined by only two gates: a reset gate and an update gate. The reset gate represents the short-term memory and determines how much information from the past should be kept or discarded. Similarly to the LSTM gates, it is defined as follows:

$$r_t = \sigma(W_{r,x} + x_t + W_{r,h}h_{t-1} + b_r) \tag{2.9}$$

The update gate, on the other hand, represents the long-term memory and is comparable to the LSTM's forget gate:

$$u_t = \sigma(W_{u,x} + x_t + W_{u,h}h_{t-1} + b_u) \tag{2.10}$$

The hidden state at the current time step is firstly defined as a combination of the current input (x_t) and the hidden state of the previous time step (h_{t-1}), provided to an activation function (ϕ). The reset gate controls the influence of the previous hidden state.

$$\hat{h}_t = \phi(W_{g,x} + x_t + W_{g,h}(r_t \circ h_{t-1}) + b_g) \tag{2.11}$$

Then, the candidate hidden state ($\hat{h}_t$) is combined with h_{t-1} to calculate the current hidden state. The update gate controls how they are combined.

$$h_t = u_t \circ h_{t-1} + (1 - u_t) \circ \hat{h}_t. \qquad (2.12)$$

Bidirectional RNNs (BiRNNs) and Bidirectional LSTMs (BiLSTMs) [54] are extensions of the standard RNN and LSTM architectures. While traditional RNNs process the text sequence from the beginning to the end, BiRNNs and BiLSTMs process the sequence in both forward and backward directions simultaneously. By combining information from both preceding and succeeding contexts, BiRNNs and BiLSTMs can better capture bidirectional dependencies in the text, enhancing their ability to understand the context and make more accurate predictions.

These advanced RNN variants, including LSTM, GRU, BiRNN, and BiLSTM, have become popular choices for various natural language processing (NLP) tasks. They excel in tasks like language modeling, sentiment analysis, machine translation, and text generation. Their ability to model sequential dependencies, handle long texts, and capture bidirectional context makes them powerful tools for extracting meaningful information and patterns from text data. As a result, they play a crucial role in enabling machines to comprehend human language effectively.

Transformers

Unlike traditional Recurrent Neural Networks (RNNs) and its variants, Transformers [8] do not rely on sequential processing to capture dependencies between words in a text sequence. Instead, they are neural networks that use an attention mechanism that allows them to analyze all words in the sequence simultaneously, making them highly parallelizable and more efficient.

The transformer architecture consists of an encoder and a decoder. In NLP tasks such as machine translation, the encoder processes the input text capturing both local and global context simultaneously, while the decoder generates the corresponding output. Figure 2.9 illustrates the encoder in action. When the model processes each word of the sentence, the attention mechanism enables it to look at other words in the input sequence for clues that could help to better encode the current word. The decoder, on the other hand, uses masked attention to ensure that each word can only attend to the words that come before it in the output sequence, preventing information leakage.

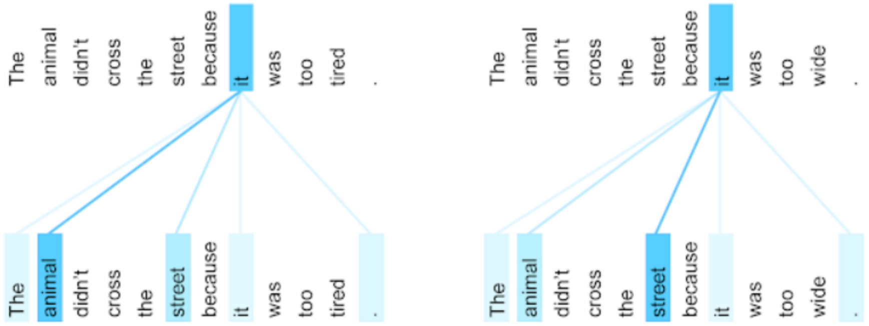


Figure 2.9: The encoder attention distribution for the word "it" of a Transformer trained on English to French translation (one of eight attention heads) [7]. Notice how the word is coupled with a different term (animal vs. street) depending on the context of the sentence (the adjective at the end).

Diving deeper into the transformer architecture, it uses an encoder-decoder structure as shown in Figure 2.10. The encoder is composed of a stack of N identical layers, that have two sub-layers each: one multi-head attention mechanism (described later in this section) followed by an addition and normalization operations (defined as $LayerNorm(x + MultiHeadAttention(x))$, and one position-wise fully connected feed-forward network (FFN) also followed by addition and normalization (defined as $LayerNorm(x + FFN(x))$. All sub-layers and embedding layers have outputs of dimension $d = 512$.

The position-wise fully connected feed-forward network consists of two linear transformations with a ReLu activation function in between, as described by Equation 2.13. It is applied to each position separately and identically.

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2 \tag{2.13}$$

The decoder also has a stack of N identical layers. Firstly it has a masked multi-head attention followed by addition and normalization. The masking part of the multi-head attention means that it sets future positions to $-\infty$, making them unreachable by current positions, and preventing information leakage. Then, another multi-head attention sub-layer is applied over the output of the encoder stack. Finally, a sub-layer with a position-wise fully connected FFN is also applied with the addition and normalization layer.

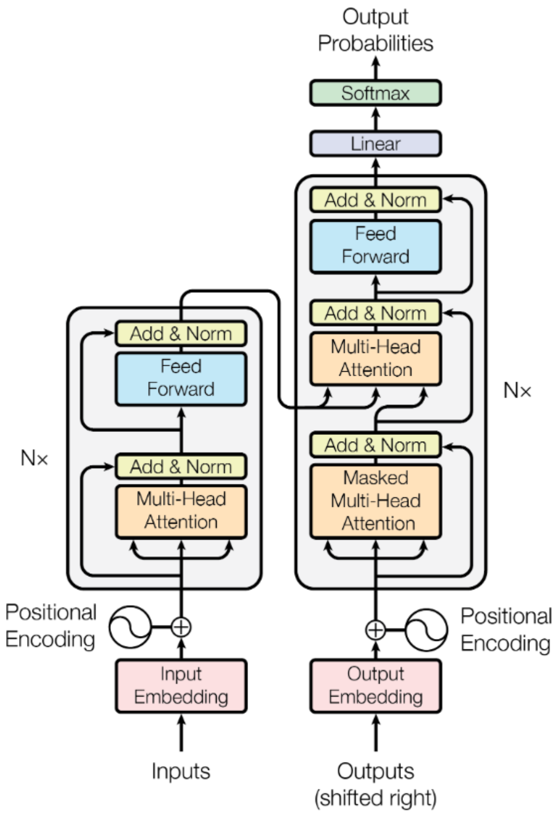


Figure 2.10: Transformer architecture [8]. The left gray square represents the encoder, while the right one represents the decoder.

Attention is a function that maps a query and a set of key-value pairs to an output. The query, keys, values, and outputs are all vectors. The query, key, and value are representations of the input, that can change depending on the context of the problem. Their concept can be extrapolated from retrieval systems: when googling something, for example, the search engine maps the query (the text from the search bar) against a set of keys (website titles, tags, content) in order to present the best matched websites (the values) [55].

The attention mechanism used in the Transformers paper [8] is called Scaled Dot-Product Attention. It uses three input vectors: queries (Q) and keys (K) of dimension d_k, and values (V) of dimension d_v. It is calculated as the dot products of the queries with all keys, divided by $\sqrt{d_k}$. Then, a soft-max function is applied to obtain the weights of the values. These operations are done matrix-wise. The matrix of outputs is computed as follows:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (2.14)$$

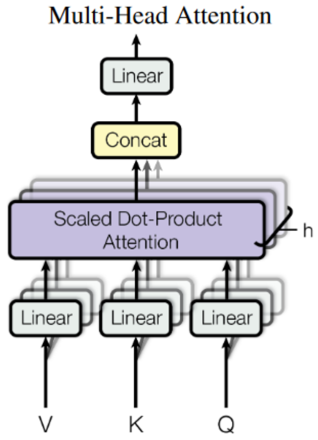


Figure 2.11: Multi-head attention [8].

The multi-head attention applies Scaled Dot-Product Attention focusing on different positions and using multiple representation sub-spaces. It uses multiple sets of query, key, and value weight matrices randomly initialized and later trained to project the input embeddings into different representation sub-spaces. The attention function is performed on each projected version of Q, K and V, yielding output values of dimension d_v. These output values are then concatenated and projected linearly, thus forming the final values, as shown in Figure 2.11. It is represented as:

$$\begin{aligned} MultiheadAttention(Q, K, V) &= concat(head_1, ..., head_h)W^O \\ \text{where } head_i &= Attention(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{2.15}$$

The projections are parameter matrices $W_i^Q, W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$, and $W^O \in \mathbb{R}^{hd_v \times d_{model}}$.

The transformer model uses learned embeddings to convert the input and output text to vectors of dimension d_{model}. Before generating the output probabilities, it also uses a learned linear transformation and a soft-max activation function in order to convert the output from the decoder to predicted next-token probabilities.

The attention mechanism of transformers enables them to assign varying levels of importance to different words in the sequence based on their relevance to each other. This allows them to consider long-range dependencies in the text effectively. By capturing these dependencies, Transformers can understand the context and relationships between words in the text more accurately, leading to better performance in NLP tasks.

A significant benefit offered by the transformer architecture is that at each step we have direct access to all the other steps (through multi-head attention). Additionally, the architecture allows for simultaneous consideration of preceding and subsequent elements, akin to the advantages of bidirectional RNNs, yet without the accompanying twofold computational requirement. Notably, these operations occur in parallel rather than a sequential manner, resulting in faster training and inference processes.

2.1.5 Sentiment analysis

Sentiment analysis constitutes a research field focused on examining people's opinions, sentiments, evaluations, and appraisals concerning various entities, including products, services, organizations, individuals, events, topics, and their attributes. Over the past years, sentiment analysis applications have been applied in several domains, such as consumer products, services, healthcare, financial services, social events, and political elections, among others [56].

Sentiment analysis has been generally studied at three levels:

- **Document level:** Classifying whole documents into positive or negative sentiments. For example, given a news article, the system determines whether the article expresses an overall positive or negative connotation on the news presented. Given that we need to classify whole web pages, document-level analyses are the focus of this work.

- **Sentence level:** Performing the classification task not on the whole document at once, but analyzing sentence-by-sentence and outputting a separate sentiment for each.

- **Entity and aspect level:** Performing finer-grained analysis than the two previous levels. Instead of separating the unit by language constructs (such as documents, paragraphs, sentences, etc.), it looks at the opinion itself, focusing not only on the sentiment but also its target. For instance, the sentence "The south of Italy is beautiful, but its summers are very hot." evaluates two targets: appearance and temperature. The sentiment on the south of Italy's appearance is positive, but the sentiment on its temperature in summer is negative.

When analyzing texts, sentiments can be seen from two perspectives: the author who expresses an opinion, and the reader who can have a different reaction than the one intended by the author. For example, if a text says "the housing price has gone down, which is bad for the economy", the author clearly brings up the negative impact of the house prices on the economy. However, readers might have different interpretations of these news based on their background.

If readers are looking for selling houses the news is definitely negative, but if they want to buy houses, this sentence could be perceived as good news. Since the reader's perspective can have multiple correct interpretations while the writer's perspective is less variable, it was decided to use the interpretation of the writer's proposed sentiment when annotating the data with sentiment labels.

The main goal of sentiment classification is the same as that of a text classification problem. Traditional text classification generally classifies documents of topics like politics, sciences, or sports, where topic-related words serve as key features. However, sentiment classification focuses on words that convey a positive or negative sentiment, such as "lovely", "amazing", "horrible", "terrifying", etc.

Being essentially a text classification problem, supervised learning methods have been employed in the literature, such as naïve Bayes classifiers and support vector machines (SVM) [57, 58], as well as deep learning approaches.

Text representations

In order to prepare the text for a classification task, every word needs to be represented as features. One of the pioneering works to apply feature extraction in this context [59] showed good results using both naïve Bayes and SVMs by representing the text as unigrams (also known as bag of words), which represent the frequencies of words without establishing an order or context between them. Another example used in subsequent research [60] was the usage of Term Frequency-Inverse Document Frequency (TF-IDF), which represents the text by calculating the word frequencies per document and dividing each of them by the frequency of the same words in the entire document collection, with the goal of re-scaling the word frequency by how rare or common they are across the collection.

In recent years, the usage of word embeddings as features has grown popularity. The main difference between word embeddings and previous methods is the usage of machine learning to best represent the text. The following paragraphs detail some of the most commonly used embeddings.

Word2Vec, a "word to vector" interpretation, introduced by Mikolov et al. (2013) [34], is composed of a two-layer neural network. It aims to represent words as vectors in a high-dimensional space, where similar words are positioned closer to each other. The method comes in two variations: continuous bag-of-words (CBOW) and skip-gram. In the CBOW model, the algorithm predicts a target word based on its surrounding context words, while in skip-gram, it predicts the context words given a target word. Word2Vec has proven to be a powerful tool for various natural language processing tasks, as it captures semantic relationships and word similarities in an efficient and scalable manner.

GloVe (Global Vectors for Word Representation) was created by Penningtal et al. (2014) [35]. Unlike Word2Vec which focuses on local context, GloVe captures global word co-occurrence statistics to generate word vectors. By factorizing a co-occurrence matrix that calculates how often pairs of words appear together in a given text, it is able to preserve both semantic and syntactic relationships.

FastText was introduced by Facebook's research team in 2017 [36]. It represents words as continuous-valued vectors, but with a significant improvement: it breaks words down into smaller subword units. By doing so, fastText can handle out-of-vocabulary words and capture morphological information, which is especially useful for the representation of slang, real-world text data with varying spellings, and languages with rich morphology. It operates by constructing a bag-of-words representation for each subword and then learns the word embeddings using a shallow neural network. This approach allows fastText to be computationally efficient and handle large vocabularies effectively.

USE, also known as Universal Sentence Encoder, is a powerful pre-trained sentence embedding model introduced by Google Research in 2018 [40]. Unlike word embeddings that represent individual words, USE was designed to generate fixed-length vector representations for entire sentences or short texts. The model is based on a deep transformer architecture and is trained on a large-scale corpus containing diverse language data. What sets USE apart is its versatility, as it can encode sentences in multiple languages and handle sentences of varying lengths, producing dense and semantically meaningful embeddings that capture the contextual information of the entire sentence.

ELMo, short for Embeddings from Language Models, was introduced by Peters et al. in 2018 [61]. Their innovative work generates contextualized word embeddings, meaning that the representation of a word varies depending on its context in a sentence. It is based on bidirectional LSTM networks, whose higher-level states capture context-dependent aspects of word meaning, while the lower-level states model aspects of syntax. During training, ELMo learns to encode words in a way that considers the entire sentence, resulting in word embeddings that are sensitive to the surrounding context. This allows ELMo to handle terms with multiple meanings and capture nuances in word usage.

BERT, which stands for Bidirectional Encoder Representations from Transformers, was introduced by Devlin et al. (2018) [37]. Like ELMo, BERT also generates contextualized word embeddings, but it utilizes a transformer-based architecture to achieve bidirectional context modeling more efficiently. It employs the attention mechanism of transformers to capture contextual information in a parallel and more scalable manner, making it computationally faster. It has become a dominant model in NLP due to its exceptional ability to capture context, leading to state-of-the-art performance on many natural language understanding tasks and inspiring many other transformer-based embeddings.

Similar to other models, BERT has more than one model variation. In the paper it was introduced there were two models: BERT-base and BERT-large. As the names suggest, the difference is in their sizes. While BERT-base has 12 transformer layers, 12 attention heads, and 110 million parameters, BERT-large has 24 layers, 16 attention heads, and 340 million parameters. BERT-large usually shows better performance than BERT-base, however it also requires more computational resources and/or time to train and fine-tune. For example, BERT-base was trained on 4 cloud TPUs for 4 days and BERT-large was trained on 16 TPUs for 4 days.

Multilingual BERT is a variation of BERT trained on the same large-corpus data, but across 104 languages instead of only English data.

RoBERTa, a Robustly Optimized BERT Pretraining Approach, was developed by Liu et al. at Facebook AI in 2019 [39], as an optimized version of the BERT model. It was trained on a large corpus of diverse text data, including news articles, to improve its language representation capabilities. The training process for RoBERTa involves a larger corpus of data, longer sequences, larger batch sizes, and more iterations compared to BERT, resulting in a more powerful language model.

DistilBERT is a distilled version of the original BERT model, introduced by Sanh et al. in 2020 [38]. The process of knowledge distillation involves compressing a large complex model (teacher model), like BERT, into a smaller and more lightweight version (student model), while retaining as much of its knowledge as possible. During this process, the student model is trained to mimic the output probabilities of the teacher model on a labeled dataset. DistilBERT achieves significant model compression (usually about 40% smaller) and faster inference while still maintaining competitive performance with respect to the original BERT model on various NLP tasks. This allows for more efficient deployment of BERT-like models on resource-constrained devices or systems with limited computational capabilities.

LaBSE, which stands for Language-agnostic BERT Sentence Embedding, was introduced by Feng et al. from Google AI Research in 2022 [62]. It is an extension of the original BERT model with the goal of providing cross-lingual sentence representations. Trained on translated sentence pairs, it learns language-agnostic sentence embeddings that capture semantic information across different languages. This makes LaBSE particularly useful for multilingual applications, as it can handle sentences from various languages and still provide meaningful and comparable embeddings.

XLNET is a state-of-the-art language model introduced by Yang et al. (2020) [41]. It builds upon the transformer architecture and overcomes some limitations of previous models by introducing a permutation-based training approach that enables learning bidirectional contexts. It does so by maximizing the expected likelihood over all permutations of the factorization order, thus considering all context combinations during training. This bidirectional context modeling results in better sentence representations and improved performance on various NLP tasks, being the current model with the highest accuracy on the Sentiment Analysis IMDb benchmark as reported by Papers with Code [63].

Cross-language sentiment classification

Cross-language, or language-agnostic, sentiment classification aims to perform sentiment classification in texts of multiple languages. Its main motivation would be the desire to use English-trained models that are much more abundant in the literature in other languages and other countries. In the context of this project, training on English data to create cross-language models enables us to use the GoEmotions dataset to train models for all languages. This is interesting because GoEmotions is one of the largest emotion human-annotated datasets and it has an ample number of emotion categories, being greater than datasets in other languages.

In order to achieve language-agnostic classification, a few approaches have been proposed in the literature. The simplest approach could be to translate the texts into English and then apply the English-trained models. A few setbacks can be seen in this approach, such as the bias introduced by the translator and the difficulty in translating words while keeping meaning and sentiment being represented. In order to overcome part of these issues, Wan (2008) [64] used the output of several Chinese-to-English translators, which produced different English versions of the same document. Those were then each classified separately and their results were aggregated using different ensemble methods, such as average, maximum, voting, etc. More recent approaches involve training word embeddings on large corpora of data in multiple languages in order to obtain multilingual vector representations. Such approaches have reached good results, as shown in the following section.

2.2 Related work

In recent years the number of studies encompassing text classification and even sentiment analysis specifically has grown greatly, due to the advances in text embedding and deep learning techniques that come coupled with the advances in computational resources. This section provides a summary of the state-of-the-art with regard to sentiment analysis models.

2.2.1 Machine learning models

Some of the commonly used classifiers in sentiment analysis include non-deep learning models, such as support vector machine (SVM), naïve Bayes, logistic regression, random forest, and decision trees [65].

For instance, Jung et al. (2016) [66] used multinomial naïve Bayes to classify Tweets into positive, negative, or neutral. They reported an accuracy of 85% on the Sentiment140 dataset [67], which is a dataset collected by Stanford University that consists of 1.6 million labeled tweets from customers, evenly split between the positive and negative sentiment classes.

Hemakala and Santhoshkumar (2018) [16] compared multiple machine learning models, such as decision tree, random forest, support vector machine, k-nearest neighbors, logistic regression, Gaussian naïve Bayes, and AdaBoost. The authors collected over 14k tweets related to Indian Airlines, also labeled as positive, negative and neutral. The results showed that Random Forest had the highest F1-score, reaching 87%.

Rahat et al. (2019) [21] compared sentiment analysis models on 10k tweets also labeled as positive, negative and neutral. The models analyzed were multinomial naïve Bayes and support vector classifier (SVC) with linear kernels. Its results showed that the latter had the highest accuracy, of 82% as opposed to 77% from multinomial naïve Bayes.

A study conducted by Saad (2020) [18] on positive, negative and neutral classification compared six different machine learning models: SVM, logistic regression, random forest, XGBoost, naïve Bayes, and decision trees. Its results showed that SVM achieved the highest accuracy of 83%, followed by logistic regression with 82%.

Finally, Jemai et al. (2021) [20] employed five machine learning models using Twitter data available in the Natural Language Toolkit [68] with an equal number of positive and negative samples. The models employed were naïve Bayes, multinomial naïve Bayes, Bernoulli naïve Bayes, logistic regression, and linear support vector machines. The preprocessing of the data included tokenization, stop word removal, URL and symbol removal, case folding, and lemmatization. Its results showed that the naïve Bayes method achieved the highest accuracy of 99.7%.

It is important to emphasize that the metrics across the papers cannot be directly compared to each other, since they use different test datasets. These datasets may exhibit divergent label distributions, cover distinct subject matters leading to disparate terminologies, and encompass varying text lengths. For instance, while tweets are confined to 140 characters, other textual formats might not be subject to any length limitations. These variations can produce different results and need other types of models.

Furthermore, models exhibiting proficiency in one task (such as categorizing tweets concerning airline opinions) may not necessarily excel in another task (for instance, analyzing lengthy passages discussing war). This means that even though multinomial naïve Bayes reached an impressing accuracy of 99.7% in a specific setting, it might not be the perfect model for all solutions. Hence, the analysis of related work should be done by analyzing possible candidate models to be tried for the specific goal of this work. This entails experimenting with models that achieved good performance metrics, rather than applying those models to a different task and aiming to achieve the same accuracy as the task reported in one of these papers. Table 2.1 shows a summary of the papers presented in this section.

Reference	**Dataset**	**Dataset size (# rows)**	**Labels**	**Models compared (ordered by highest scoring first)**
Jung et al. (2016) [66]	Sentiment140 [67]	1.6M	3 emotions (positive, neutral, negative)	MultinomialNB
Hemakala and Santhoshkumar (2018) [16]	Twitter US Airline Sentiment [69]	14k	3 emotions (positive, neutral, negative)	Random Forest, SVM, AdaBoost, Logistic Regression, Gaussian Naïve Bayes, Decision Tree, KNN
Saad (2020) [18]	Twitter US Airline Sentiment [69]	14k	3 emotions (positive, neutral, negative)	SVM, Logistic Regression, Random Forest, XGBoost, Gaussian Naïve Bayes, Decision Tree
Rahat et al. (2019) [21]	Twitter Airline Reviews	10k	3 emotions (positive, neutral, negative)	MultinomialNB, LinearSVC
Jemai et al. (2021) [20]	NLTK's Twitter corpus *	10k	2 emotions (positive, negative)	Naïve Bayes, MultinomialNB, BernoulliNB, Logistic Regression, LinearSVC

Table 2.1: Comparison of papers performing sentiment classification using non-deep learning models

2.2.2 Deep learning models

Deep learning approaches have become more popular over the past years in the field of sentiment classification. Ranging from multi-layer perceptrons (MLP) to transformers.

For example, a study conducted by Dholpuria et al. (2018) [17] compared several machine learning and deep learning methods, such as naïve Bayes, SVM, logistic regression, k-nearest neighbors, ensemble models, and a convolutional neural network (CNN). The dataset used contained 3k reviews with positive and negative labels from IMDb movie reviews. The results showed that the highest accuracy was 99.3% belonging to the CNN model.

Furthermore, Yang (2018) [26] proposed a CNN that uses RNNs as the convolution filters. The Stanford Sentiment Treebank (SST) dataset [70] was used, encoded with GloVe word embeddings. The SST dataset consists of sentences from movie reviews, where each sentence is parsed into a tree structure with sentiment labels assigned to nodes representing phrases or words. It is represented in two forms, SST-2 (or SST binary) which only has positive and negative labels, and SST-5 (or SST fine-grained), which has labels negative, somewhat negative, neutral, somewhat positive, and positive. Yang's work concluded with an accuracy of 89% on the SST-2 dataset and 53% on the SST-5 dataset.

Rhanoui et al. (2019) [27] used a dataset of two thousand articles and news articles labeled as positive, negative and neutral and preprocessed using a pre-trained doc2vec model [71] (an extension of word2vec) in their work. They presented a hybrid model that combines CNNs and bidirectional long short-term memory (BiLSTM) networks for sentiment analysis, which reached an accuracy of 91% on the dataset.

Dang et al. (2020) [24] performed a comparative study on sentiment analysis using 1.6 million tweets and movie, book, and music reviews obtained from eight datasets labeled as positive, negative or neutral sentiments. The data was preprocessed using word2vec embeddings and TF-IDF. An experimental study was conducted using a DNN, a CNN, and an RNN. The results showed that TF-IDF has worse models and requires longer computational time than word embeddings. Among the deep learning models, the CNN was concluded to have the best trade-off between processing time and accuracy, although the RNN had the highest accuracy over all datasets.

The Sentiment140 dataset was once again used, along with the Twitter US Airline Sentiment dataset, a collection of customer reviews over six major American airlines published in 2017 by CrowdFlower. Harjule et al. (2020) [19] used such datasets to compare multinomial naïve Bayes, logistic regression, SVM, LSTM, and an ensemble of such models with majority voting.

Raza et al. (2021) [72] used an MLP model with five hidden layers to classify COVID-19 related tweets into positive, negative and neutral sentiments. The text data was represented using two feature extractors: a TF-IDF and a count vectorizer, which were separately classified and compared. The representation using the count vectorizer achieved the highest accuracy of 94%.

AL-Smadi et al. (2023) [73] developed a Pooled-GRU model with multilingual universal sentence encoder embeddings to perform aspect-based sentiment analysis in Arabic text. The SemEval 2016 competition Task-5 dataset [74], that is composed by comments and reviews from customers on six different domains (restaurants, hotels, museums, laptops, mobile phones, and digital cameras) written in 8 languages, was used for training and evaluation. They achieved an F1-score of 93%.

Table 2.2 summarizes the papers described in this section.

Reference	Dataset	Dataset size (# rows)	Labels	Models compared (ordered by highest scoring first)
Dholpuria et al. (2018) [17]	IMDb movie reviews	3k	2 emotions (positive, negative)	CNN, SVM, Logistic Regression, KNN, Naïve Bayes
Yang (2018) [26]	SST dataset [70]	11k	5 emotions (negative, somewhat negative, neutral, somewhat positive, positive)	CNN with RNN as filters
Rhanoui et al. (2019) [27]	News articles	2k	3 emotions (positive, negative, neutral)	CNN with BiLSTM
Dang et al. (2020) [24]	Movie, book, and music reviews from Twitter	1.6M	3 emotions (positive, negative, neutral)	RNN, CNN, DNN
Harjule et al. (2020) [19]	Sentiment140 [67], Twitter US Airline Sentiment [69]	1.6M, 14k	3 emotions (positive, negative, neutral)	RNN with LSTM, SVM, Logistic Regression, MultinomialNB
Raza et al. (2021) [72]	COVID-19 tweets	65k	3 emotions (positive, negative, neutral)	MLP with count vectorizer, MLP with TF-IDF
AL-Smadi et al. (2023) [73]	SemEval 2016 competition Task-5 dataset [74] (Arabic language)	3k	3 emotions (positive, negative, neutral)	Pooled-GRU

Table 2.2: Comparison of papers performing sentiment classification using deep learning models

2.2.3 Transformers models

The following research was done specifically using transformers models, comparing them amongst themselves and with other approaches.

Munikar et al. (2019) [29] performed sentiment classification on the Stanford Sentiment Treebank (SST) dataset using BERT, both "base" and "large" models. They reported an accuracy of 83.9% for BERT-base and 84.2% for BERT-large for the SST-5 dataset (labeled with 5 emotion labels), while the accuracy increased to 94% and 94.7% respectively for the SST-2 dataset (with only "positive" and "negative" labels).

Younas et al. (2020) [31] compared two deep learning models, multilingual BERT (mBERT) and XLM-RoBERTa (a version of Roberta trained as a cross-language model), for sentiment analysis of multilingual social media text. The dataset used (called Multisenti [75]) was collected from Twitter during the 2018 general election in Pakistan, comprising 20k Tweets in both English and Roman Urdu. The Tweets were categorized into positive, negative, and neutral classes. After fine-tuning the learning rate of the models, the outcomes demonstrated that mBERT achieved 69% accuracy, while XLM-R achieved 71% accuracy.

In another study, Dhola and Saradva (2021) [23] compared the performance of SVMs, multinomial naïve Bayes, LSTM, and BERT models using the Sentiment140 dataset. The results indicated that the best-performing model was BERT, which achieved 85% accuracy.

Smetanin and Komarov (2021) [32] compared several models for sentiment analysis in Russian, including Multilingual BERT, RuBERT, and two versions of the Multilingual Universal Sentence Encoder. They used multiple datasets with 3 emotion classes and two datasets with 5 classes (namely, RuSentiment [76] and LINIS Crowd [77]). The study showed that RuBERT had better metrics than the other approaches, reaching an F1-score of 72% in the RuSentiment dataset, but multilingual BERT and multilingual Universal Sentence Encoder had competitive metrics as well, presenting F1-scores of 71% and 69%, respectively, for the same dataset.

Fattoh et al. (2022) [25] performed semantic sentiment classification on COVID-19 tweets labeled as positive, negative, and neutral using a feedforward deep neural network (DNN) fed with Universal Sentence Encoder embeddings, as described in Section 2.1.5. They achieved an accuracy of 78% over 60 epochs.

The papers above are summarized in Table 2.3.

Reference	Dataset	Dataset size (# rows)	Labels	Models compared (ordered by highest scoring first)
Munikar et al. (2019) [29]	SST [70]	11k	5 emotions (negative, somewhat negative, neutral, somewhat positive, positive)	BERT-large, BERT-base
Younas et al. (2020) [31]	Multisenti [75] (multilingual: English and Roman Urdu)	20k	3 emotions (positive, neutral, negative)	XLM-Roberta, mBERT
Dhola and Saradva (2021) [23]	Sentiment140 [67]	1.6M	2 emotions (positive, negative)	BERT, LSTM, MultinomialNB, SVM
Smetanin and Komarov (2021) [32]	RuSentiment [78] (Russian language)	31k	3 emotions (positive, neutral, negative)	RuBERT, mBERT, Multilingual USE
Fattoh et al. (2022) [25]	COVID tweets	10k	3 emotions (positive, neutral, negative)	DNN

Table 2.3: Comparison of papers performing sentiment classification using Transformers

2.2.4 Fine-grained sentiment analysis

The research shown in the subsections so far focuses on sentiment analysis based on positive, negative and neutral emotion categories. In this subsection, related work including fine-grained emotion categories is presented.

Adoma et al. (2020) [30] compared four Transformer-based models (BERT, RoBERTa, distilBERT, and XLNet) performing sentiment classification on the ISEAR dataset [79], which consists of 7 thousand sentences labeled using 7 emotions: anger, disgust, fear, sadness, shame, joy, and guilt. The outcomes showed RoBERTa has the highest accuracy (74%), followed by XLNet (73%) and BERT (70%). DistilBERT achieved an accuracy of 67%. The authors also presented that distilBERT was the fastest model, while XLNet was the slowest one.

When Demszky et al. (2020) [11] published the collected GoEmotions dataset that includes 28 emotion categories as described in Section 4.1.1, they used a BERT-based model to provide a baseline for future experiments. Their model achieved an average F1-score of 46% over their taxonomy.

Suresh and Ong (2021) [80] proposed a Label-aware Contrastive Loss (LCL) that adaptively weights a given input's positive/negative samples based on the label-relationships between them. This loss brings semantically-similar labels (such as "sad" and "devastated") closer, as opposed to treating them the same as other label pairs (such as "sad" and "happy"). In order to compare their results, they used an ELECTRA-base model [81] and 5 different datasets: Empathetic Dialogues [82], a dataset of 25k conversations grounded in emotional situations labeled with 32 emotions; GoEmotions; ISEAR; EmoInt [83], a dataset created for the WASSA 2017 shared task consisting of tweets labeled with 4 emotion categories (joy, sadness, fear, and anger); and

Reference	Dataset	Dataset size (# rows)	Labels	Models compared (ordered by highest scoring first)
Adoma et al. (2020) [30]	ISEAR [79]	7k	7 emotions (anger, disgust, fear, sadness, shame, joy, guilt)	RoBERTa, XLNET, BERT, distilBERT
Demszky et al. (2020) [11]	GoEmotions [11]	58k	28 emotions	BERT
Cortiz (2022) [33]	GoEmotions [11]	58k	28 emotions	RoBERTa, distilBERT, XLNET, BERT, ELECTRA
Suresh and Ong (2021) [80]	Empathetic Dialogues [82], GoEmotions [11], ISEAR [79], EmoInt [84], SST [70]	25k, 58k, 7k, 7k, 11k	32 emotions, 28 emotions, 7 emotions (anger, disgust, fear, sadness, shame, joy, guilt), 4 emotions (joy, sadness, fear, anger), 5 emotions (negative, somewhat negative, neutral, somewhat positive, positive)	ELECTRA

Table 2.4: Comparison of papers performing fine-grained sentiment classification

the Stanford Sentiment Treebank. The results showed that the LCL approach had higher accuracy and F1-score for the ELECTRA-base models in all datasets, achieving 65.5% accuracy and 64.8% F1-score in the GoEmotions dataset.

Cortiz (2022) [33] performed a comparison on Transformers models on fine-grained emotion recognition using the GoEmotions dataset. The models compared were BERT, DistilBERT, RoBERTa, XLNET, and ELECTRA. For the 28 sentiment labels, ELECTRA showed the lowest results, with an F1-score of 33%, even though it was the fastest model. RoBERTa achieved the highest F1-score of 49%, performing best for 14 out of the 28 emotion classes. The model was followed by distilBERT (48%), XLNET (48%), and BERT (46%). Even though distilBERT and BERT achieved relatively low results, they were the only models who achieved good results for the "pride" class (with an F1-score of 22% and 36%, respectively), while the other models had an F1-score of zero. Furthermore, distilBERT was highlighted as a quick model to train when compared to the other models, being presented as a good trade-off between time and performance.

A summary of the papers presented is shown in Table 2.4. As can be seen in the research presented above, there are few studies focused on more than five fine-granular sentiments in sentiment classification. While positive, negative, and neutral classes are frequently utilized in sentiment analysis, they might not encompass the entire spectrum of emotions and intensities a person can convey. In order to obtain more nuanced insights into the sentiment expressed in a text, fine-grained sentiment analysis encompassing more specific classes like happy, caring, sad, angry, or surprised could be pursued. This work presents an opportunity to fill this gap in the field of programmatic advertising.

We can also see that even though Transformer models have been used in recent years, there has also been space for other machine learning and deep learning approaches. Models such as SVM, logistic regression, CNN, RNN, LSTM, and transformers are explored in this work.

3
Problem exposition

This chapter presents a more detailed description of the field of programmatic advertising in section 3.1, followed by an exploration of the business need for employing sentiment-based text classification in section and a finer detailing of the research questions in section 3.2. Finally, sections 3.3 and 3.4 provide a description of the data where the models will be applied and the technical constraints that need to be met by the model.

3.1 Business Understanding

In the field of advertising, there are two main characters: publishers and advertisers. In the past, agencies had to write directly to publishers in order to submit orders to publish their ads and negotiate price, size, and other details. As the internet gained its space in society, it brought the need for much more agility in this market. The need for higher speed brought up the process of programmatic media-buying. It involves the automated negotiation of digital ad space, encompassing various formats like display ads, banners, and videos. This process requires a third player in the negotiation: the ad network, an advertising technology (AdTech) platform that acts as a broker between the advertiser and publisher, efficiently buying and selling online ads. The actions of the three players are described in Figure 3.1.

Real-time bidding (RTB) is a method used in programmatic media-buying where ad space (or ad inventory) is negotiated by the number of impressions through an auction-based system. An ad impression is the representation of one ad viewed by one user. The bidding system

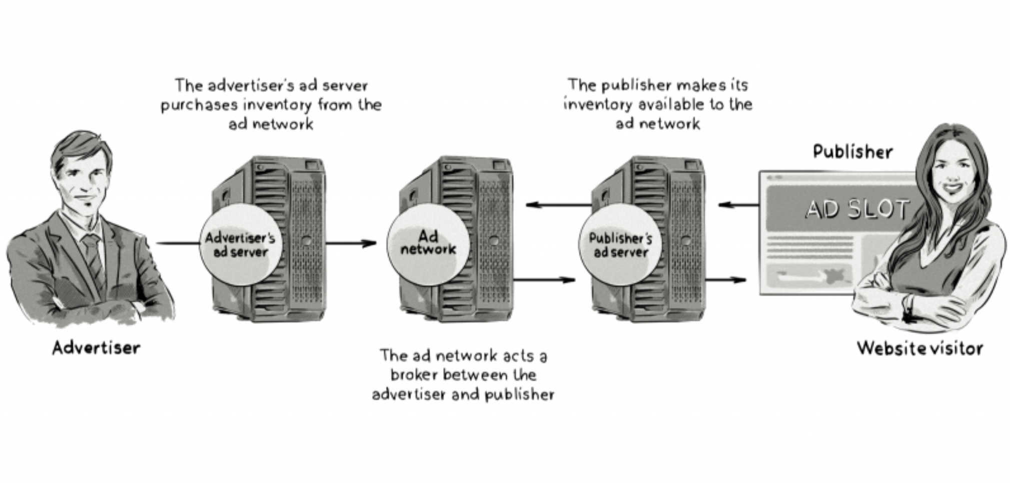

The advertiser's ad server purchases inventory from the ad network
The publisher makes its inventory available to the ad network
Publisher
AD SLOT
Advertiser's ad server
Ad network
Publisher's ad server
Advertiser
Website visitor
The ad network acts a broker between the advertiser and publisher

Figure 3.1: Description of the roles of the publisher, advertiser, and ad network in the field of programmatic media-buying [9].

involves the buying and selling of individual ad impressions, where advertisers bid for the opportunity to display their ads to specific target audiences on websites or apps, which empowers advertisers to display ads only to users who are more likely to engage with their products or services [9].

In this context, Anonymised[*] has the role of the ad network, connecting advertisers and publishers in a way that preserves the privacy of users, by anonymizing user data while still sharing clusters of audiences with the targeted interest to advertisers.

One interesting feature of the ad network is the ability to properly categorize the publishers' content in order to connect with the audiences targeted by the advertisers. Advertisers might want to avoid specific topics, such as "violence", however filtering only by topics could be restricting websites and articles unnecessarily. The tone of the articles could also influence the advertisers' choices.

For example, articles that talk about war are potentially undesirable for advertisers. But if instead the website talked positively about places where there is a war happening (e.g. "Beautiful places in Ukraine before the war"), there might be an audience to be positively reached by advertisers. Changing the point of view to publishers, there are articles about war video games (such as "God of War" or "Call of Duty: Modern Warfare") that could be avoided by advertisers simply because of the topic, even though when actually analyzing the tone of the content, they could be seen as good ad spaces for advertisers.

[*] `https://www.anonymised.io/`

This issue brings forward an opportunity to apply sentiment analysis in publishers' articles in order to properly categorize them and make them available (or filtered out) to advertisers.

3.2 Detailed Research Questions

Generally, existing solutions in the field of AdTech offer basic sentiment analysis models with binary outputs (positive/negative). However, a more innovative product that provides nuanced emotions like sadness or anger, instead of a generic "negative" can offer greater value from a business standpoint.

On the other hand, including similar sentiments with overlapping meanings, such as "amusement" and "joy" may not be advantageous, since it is difficult for advertisers to discern between them. When choosing ad spaces to reach their target audience, web content that conveys amusement or joy would have similar (if not the same) results. Understanding how to effectively group these sentiments into meaningful categories is crucial for delivering value beyond simplistic "positive," "negative," and "neutral" classifications.

Furthermore, considering the company's international clients, an essential aspect to consider is employing multilingual models. The development of a single language-agnostic model capable of interpreting multiple languages (without the need for additional training on specific datasets for each language) would be ideal. Although having one model per language could possibly result in better metrics per language, finding large datasets specific for each language and labeled with fine-grained emotion categories is difficult. Moreover, maintaining multiple models requires more computational resources, when compared to one multilingual model.

Hence, the objective of this thesis is to address the following research questions:

1. How well can web data be classified into more granular sentiment categories?
2. What level of sentiment granularity should be employed?
3. Is it feasible to employ a language-agnostic model for this task? If so, how does it compare to a single language model?

3.3 Data Understanding

This section describes the web data on where the models will be applied.

The web data being referenced in this thesis is composed of different types and topics, e.g. news articles, recipes, tutorials, and song lyrics. Its texts range from a few sentences to long texts with multiple paragraphs. The company has a web scraper that collects the texts from the domains made available and authorized by the publishers and saves those texts and URLs in a database. The texts collected can range from a couple of words, in the case of web pages mostly composed of images, to long texts. Besides the main text, the scraper also collects article titles. As of the development of this project, the scraper also performs some pre-processing on the texts, by removing punctuation and turning all characters to lowercase.

3.4 Resource constraints

In order for this model to be feasibly used, a few constraints must be set in place. Here are described resource constraints on the computational resources and time being utilized for the training and prediction of the model.

The project's goal is to train the model once and subsequently make daily inferences on newly scraped publisher data. Therefore, the time constraints for the training phase are relatively flexible. Regarding computational resources, it is assumed that a machine equipped with a GPU and 12GB of RAM will be available for this purpose. However, if larger models yield significant and justifiable benefits, more extensive resources can be provided to accommodate them.

Regarding execution resources, the model's runtime for processing the daily new data should ideally be limited to a few hours, given that the model will likely be run during the early-hours of the day.

4
Methods, Results and Findings

This chapter presents the methodology, results, and their interpretation for the development of machine learning models that predict granular sentiments of web text data.

Section 4.1 describes the datasets used for training and evaluating the models, followed by a description of the sentiment groups chosen in Section 4.2. Then Section 4.3 describes the preprocessing techniques used, followed by the description of the experiments using different feature representations and algorithms in Section 4.4. Section 4.5 discusses the results of the evaluation of the chosen models in real-world web data. Finally, Section 4.6 describes the evaluation of the selected models for multilingual data.

Namely, the research questions described in chapter 3 are answered in the following sections:

1. How well can web data be classified into more granular sentiment categories? Section 4.4.3.
2. What level of sentiment granularity should be employed? Section 4.2.
3. Is it feasible to employ a language-agnostic model for this task? If so, how does it compare to a single language model? Section 4.6.

Notes on the supervision of this project

This work was mainly supervised at Anonymised by Giovanni Vedana as Senior Data Scientist and Mattia Fosci as Product Owner. The supervision included directions on the final goal of the project, the choice of the GoEmotions dataset as a starting point, which models to begin with, and which approaches to pursue. For instance, the choice of starting from higher-level sentiment categories and then studying models on fine-grained emotion categories later was suggested by Giovanni. The emotion groups were first experimentally chosen by the author who then presented them to the supervisors, who then contributed with the final groupings. The supervisors also assisted in manually annotating the real-world web data, in order to provide different points of view for the labeled data.

4.1 Data sources

There are multiple data sources available in the literature. It was decided to use GoEmotions due to its number of emotion labels, as well as its extensive amount of data. The need for an additional dataset was brought up during the experiments, which brought to attention the WASSA 2021 shared task dataset, whose data is more similar to the web data to be used in production. Both datasets are described in the subsections below.

4.1.1 GoEmotions Dataset

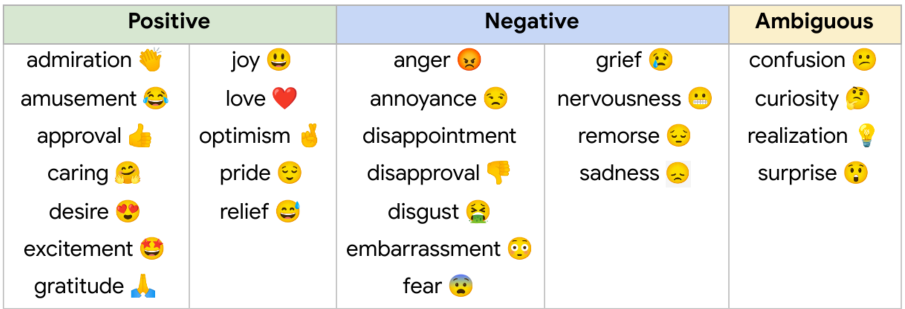
GoEmotions[*] [11] is a large manually annotated dataset created by Google that consists of 58k English Reddit[†] comments, labeled using 28 emotion categories, including "neutral". The Reddit comments were gathered from multiple subreddits that had at least 10k comments. Non-English and deleted comments were discarded, and several curation measures were taken in order to ensure that the content did not reinforce general or emotion-specific language biases. Not safe for work content was also filtered out.

The emotion categories used are: admiration, amusement, approval, caring, desire, excitement, gratitude, joy, love, optimism, pride, relief, anger, annoyance, disappointment, disapproval, disgust, embarrassment, fear, grief, nervousness, remorse, sadness, confusion, curiosity, realization, surprise, and neutral; such categories are broadly grouped into positive, negative and ambiguous emotion categories by the authors. This classification is illustrated in Figure 4.1. Such choice of emotion categories brings a greater spectrum of positive emotions when compared to Ekman's taxonomy [12] that only has one positive emotion amongst 6 basic emotion categories (joy, anger, fear, sadness, disgust, and surprise). For the goal of classifying website

[*] `https://github.com/google-research/google-research/tree/master/goemotions`

[†] `https://www.reddit.com/`

texts, it was a business need to distribute the spectrum of positive and negative emotions more evenly, so the choice of using a taxonomy broader than Ekman's made sense.



Positive		Negative		Ambiguous
admiration 👏	joy 😃	anger 😡	grief 😢	confusion 😕
amusement 😂	love ❤️	annoyance 😒	nervousness 😬	curiosity 🤔
approval 👍	optimism 🤞	disappointment	remorse 😔	realization 💡
caring 🤗	pride 😌	disapproval 👎	sadness 😞	surprise 😲
desire 😍	relief 😅	disgust 🤮		
excitement 🤩		embarrassment 😳		
gratitude 🙏		fear 😨		

Figure 4.1: GoEmotions taxonomy: 27 emotions categorized into positive, negative, and ambiguous [10].

For the data annotation, three English native speakers from India were assigned per data example. The examples where no raters agreed on at least one emotion label were assigned to two additional raters. Raters could choose multiple emotions but were instructed to only select emotions for which they were reasonably confident that it was expressed in the text. In case they were not certain about any emotion being expressed, they were instructed to select "neutral".

Regarding the choice of emotion labels, the authors began experimenting with the categories presented by Cowen and Keltner (2017) [85]. After reviewing the results from the pilot rounds, they removed emotions that were scarcely selected by annotators and/or that had low interrater agreement due to being too difficult to detect or being too similar to other emotions. These emotions were boredom, doubt, heartbroken, indifference, and calmness. Other emotions were added as suggested by the raters, which were desire, disappointment, pride, realization, relief, and remorse.

The GoEmotions dataset provides multi-labeled data, which means that each data point could have more than one label. The distribution of emotion labels is illustrated in Figure 4.2. This distribution shows that the neutral label composed the vast majority of the data points, followed by three positive sentiments (admiration, approval, and gratitude). It can also be observed that there are sentiment labels with very low frequency, such as grief, pride, relief, nervousness, and embarrassment.

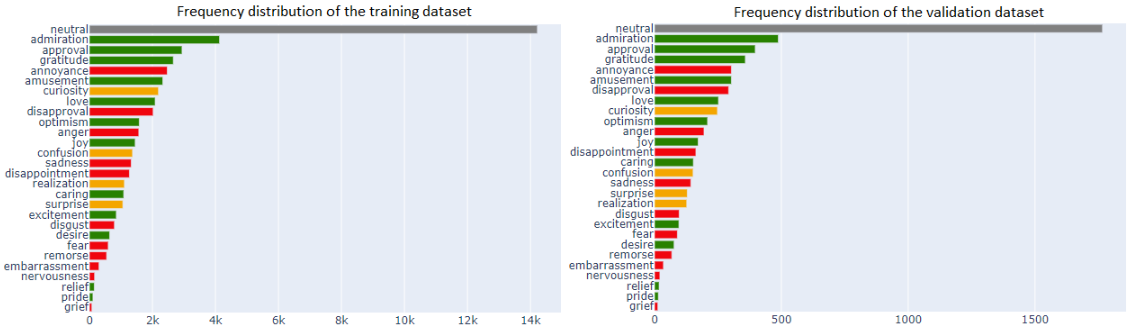


Figure 4.2: Distribution of training and validation data labels of the GoEmotions dataset. The colors indicate the macro categories of the emotions (neutral, positive, negative, and ambiguous)

4.1.2 WASSA 2021 Dataset

The 11th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis (WASSA 2021)[‡] has the goal of bringing together researchers in the field of text analysis. Each year the workshop proposes a shared task to its participants and provides the dataset for it. The 2021 edition presented the Shared Task on Empathy Detection and Emotion Classification[§].

The dataset provided by the shared task is composed of a group of essays between 300 and 800 characters long, that present empathic reactions to news stories [42]. This dataset is enriched with person-level demographic information (age, gender, ethnicity, income, education level) as well as personality information, but such sensitive information is not used in the scope of this project, since such data is not available in the web text data where the model will be applied. The dataset also provides emotion labels to the essays, at both document and sentence levels. In the scope of this project, only the essays and emotion labels at the document-level were used.

The dataset was created through a crowdsourcing task. The participants read a random selection of five news articles, subsequently rated their level of empathy and distress (other information available in the dataset but not used in the context of this project), then wrote about their thoughts and feelings. The collected essays were then classified into the 6 basic Ekman emotion labels, firstly automatically predicted and then manually verified.

The distribution of emotion labels in the training dataset (the only portion of the dataset used in this project) is shown in the plot illustrated in Figure 4.3. The majority of the samples

[‡] https://wt-public.emm4u.eu/wassa2021/

[§] https://competitions.codalab.org/competitions/28713 are represented by "sadness" and "anger", while there is an even distribution amongst "disgust", "fear", and "surprise". There is also a smaller percentage of "joy" labels. The "no emotion" labels can be interpreted as "neutral".

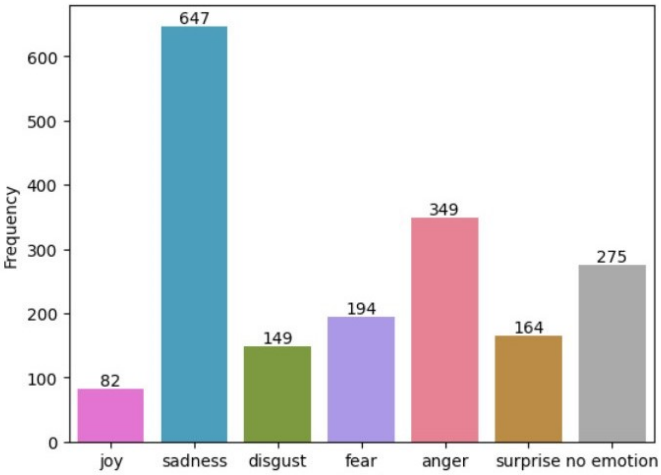


Figure 4.3: Emotion label distribution of the WASSA 2021 Shared Task Dataset.

4.1.3 Labeled Web Data

In order to properly evaluate the models in a production-like scenario, it is important to have a sample of the data they will be predicted on. Therefore, a joint effort was made to manually label 200 data points extracted from randomly sampled domains in the scraper's output database.

Each data point was labeled by the author and her supervisor using the 28 GoEmotions labels, which were later translated into the 6 sentiment groups proposed as described in Section 4.2. The label provided could be a single emotion or a list of emotions. The output from both annotators was afterward merged into one list. In order to facilitate the comparison of the true labels with the output of the single-label models, this list of emotions was later summarized into one label, which was qualitatively chosen to be the most representative of each data point.

4.2 Emotion groups

Although there was a business need to have more than one positive emotion (as opposed to Ekman's taxonomy), it was determined that the output of 28 categories was excessive for practical use during this stage of the product. Upon qualitative observation, certain emotions were not semantically distinct enough. For instance, texts describing grief or sadness could be interpreted similarly among advertisers; as would texts conveying excitement and joy. Hence there was a need for reducing the number of emotion categories, while still maintaining a balance among the number of positive and negative emotions. This balance was asserted by the business in order to prevent biasing the advertiser's choice of emotion to filter.

A study on the co-occurrence of the 27 emotion labels ("neutral" excluded) was explored in the GoEmotions paper [11], which clustered the emotions using a dendrogram, where emotions that occurred together in the same text would appear closer to each other in the graph, as illustrated in Figure 4.4. For instance, it can be observed that emotions such as excitement and joy are closely correlated, while excitement and love are further apart. An analysis of the possible dendrogram clusters was done, taking into consideration the semantic meaning of the emotions as well.

The number of emotion groups should be reduced in order to better represent similar emotions and not confuse the user, while still maintaining an even number of positive and negative emotions. The "ambiguous" emotion labels described in the GoEmotions paper were considered not to be positive or negative. Therefore, using the dendrogram clusters illustrated in Figure 4.4, six groups of emotions were proposed. They are:

- **Appreciation** (positive emotions towards others)
- **Positive experience** (positive emotions towards oneself)
- **Repudiation**
- **Sadness**
- **Curiosity**
- **Neutral**

The 27 original GoEmotions categories are grouped according to Figure 4.5. The frequency distribution of the GoEmotions training dataset categorized in the newly formed sentiment groups is illustrated in Figure 4.6.

4.3 Preprocessing

We implemented a preprocessing method that resembled as much as possible the output of the company's web scraper, by converting all characters to lowercase, removing numbers and punctuation. The use of proper punctuation and uppercase letters could improve the model's performance, especially in the case of word embeddings and transformers. Modifying the scraper

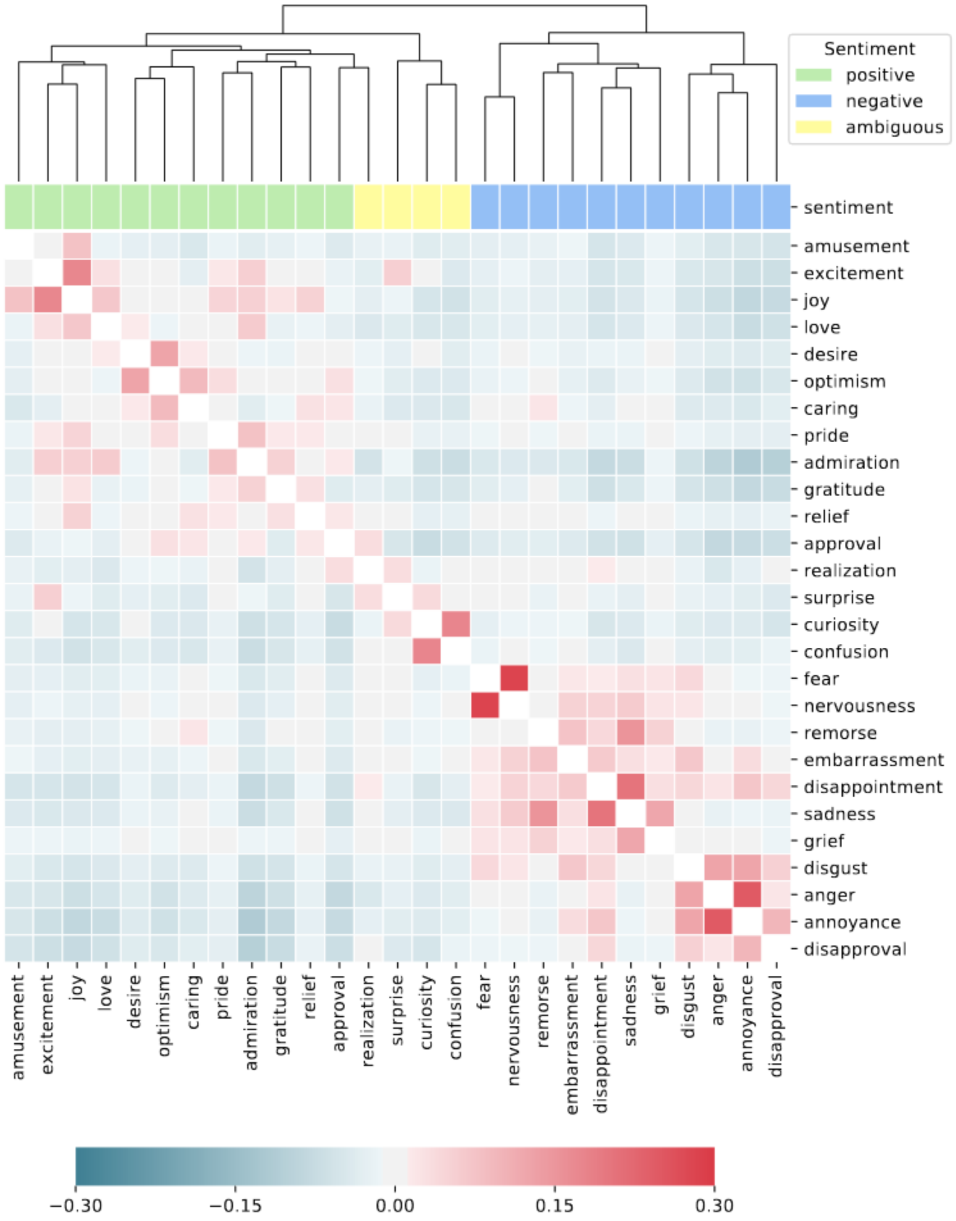
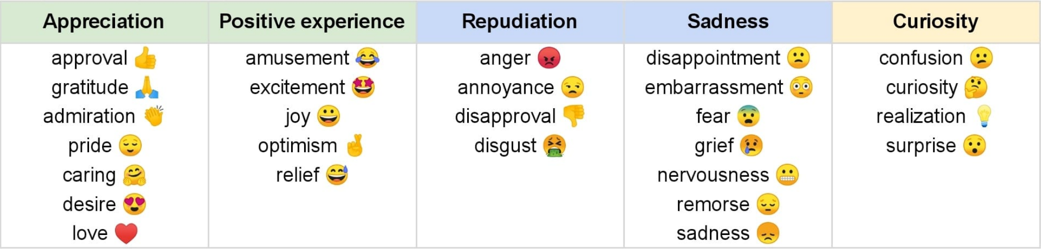


Figure 4.4: Emotion co-occurrence/correlation represented by the heatmap, while the dendrogram showcases the hierarchical clustering of the emotion labels [11].

fell outside of this project, but it is suggested to have it as future work and re-evaluate the model training with different preprocessing steps.

Some experiments were performed in order to evaluate the benefits of removing stop words (commonly used words that generally don't provide much meaningful information in a text),



Appreciation	Positive experience	Repudiation	Sadness	Curiosity
approval 👍	amusement 😂	anger 😡	disappointment 🙁	confusion 😕
gratitude 🙏	excitement 🤩	annoyance 😒	embarrassment 😳	curiosity 🤔
admiration 👏	joy 😀	disapproval 👎	fear 😨	realization 💡
pride 😌	optimism 🤞	disgust 🤮	grief 😢	surprise 😮
caring 🤗	relief 😅		nervousness 😬	
desire 😍			remorse 😔	
love ❤️			sadness 😞	

Figure 4.5: Emotion labels grouped in the 5 sentiment groups. The 6th group is the "neutral" category.

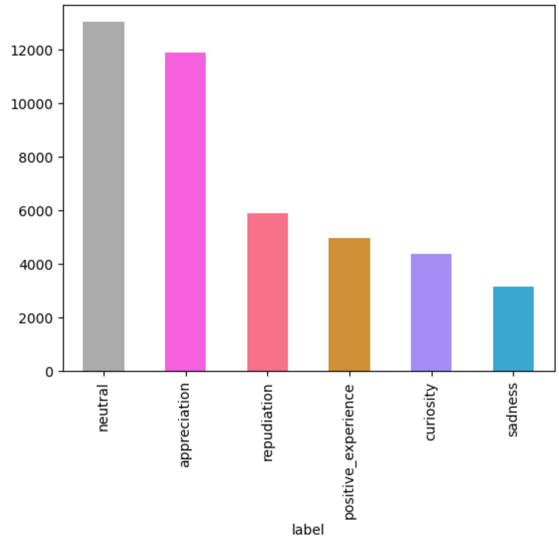


Figure 4.6: Frequency distribution of the training dataset of each label in the new sentiment group.

replacing the "n't" suffix with the word "not", and removing numbers. They were tested in a distilBERT classifier using distilBERT embeddings and a simple neural network using Multilingual Universal Sentence Encoder (mUSE) embeddings. The results are presented in Table 4.1. It can be observed that most of these changes did not improve the results of any model. Particularly, removing stop words showed worse results than the default models that use such stop words. This could indicate that the use of stop words when using word embeddings that take the whole sentence into consideration actually brings meaning into the sentences' interpretations. Given these results, stop words were not removed in the subsequent experiments.

Since the GoEmotions dataset is multi-labeled, certain criteria were determined to change the labels of rows that were assigned more than one label in order to train single-label models. The choice of experimenting with single-label models first was taken due to their being more simple than multi-label approaches. Section 4.4.4 describes the experiments done on multi-label models.

Pre-processing step applied	Model	Accuracy	Average F1-score
default	distilBERT	0.65	0.63
replace "n't" by " not"	distilBERT	0.65	0.63
remove stop words	distilBERT	0.63	0.60
remove numbers	distilBERT	0.65	0.63
default	Neural Network + mUSE	0.57	0.52
replace "n't" by " not"	Neural Network + mUSE	0.57	0.52
remove stop words	Neural Network + mUSE	0.56	0.50
remove numbers	Neural Network + mUSE	0.57	0.52

Table 4.1: Accuracy and average F1-score metrics of different pre-processing steps. "Default" represents the usage of the models without the listed steps.

From the business perspective, predicting other emotion labels is more useful than predicting "neutral" when possible. For instance, if a whole text has some neutral and some sad sentences, the company finds it more valuable to categorize the text as sad. Therefore, instances in the training dataset where a data point was labeled as both neutral and another emotion had the neutral label removed. That way, when some portions of a text are neutral but pending to another emotion (such as sadness), the text will be labeled with that emotion. This decision introduces the potential for the model to exhibit a bias towards emotions other than neutral. However, since the proportion of neutral data points in the dataset remains considerable even after this adjustment, any impact on the models' performance is expected to be minimal.

Furthermore, in order to avoid confusion in the model training, all remaining data points that retained multiple labels (approximately 7% of the training dataset) were eliminated. This prevents single-label models from being improperly trained with one label but not the other. While this might lead to a slight performance reduction compared to models trained with multi-labeled data, the percentage of discarded data is small enough that it is unlikely to significantly impact the overall performance of the models.

4.4 Text classification

This section describes the experiments done with models using three different label groupings: macro labels (only positive, negative, and neutral), all 28 GoEmotions labels, and the chosen 6 emotion groups. A final subsection is dedicated to the experiments regarding multi-label models. All the metrics provided in this section were obtained by evaluating the models in the validation set of the GoEmotions dataset. The training and validation split is already provided separately in the `tensorflow_datasets` library. This library was used in order to ensure that the same data (and split) is used as other works.

In order to evaluate the models in the experiments performed, metrics such as accuracy and F1-score were obtained. The goal of the final product is to be able to predict reasonably well all the emotion categories, balancing out negative and positive emotions. Therefore the evaluation of a model's accuracy does not bring enough information. Precision and recall of each label are evaluated to ensure that the models are evenly balanced between the emotion labels. Since the metrics for each label and model are too many to feasibly compare between the models, the weighted averaged F1-score was chosen as a comparison metric among the models. Then further investigation into the precision and recall per label is done in the best-performing models.

4.4.1 Text classification on macro groups

This section describes the experiments done for modeling on labels based on macro groups: positive, negative, and neutral. The goal of this study is to understand how different models behave on a simpler choice of categories.

For the first experiment, non-neural network models were chosen. The following models were evaluated: Logistic Regression, K-Nearest Neighbors Classification (KNeighborsClassifier), Multinomial Naive Bayes (multinomialNB), Random Forest Classification, and Linear Support Vector Classification (LinearSVC), described in Section 2.1.4.

Different feature representation techniques were also used along with the models aforementioned, such as Bag of Words, TF-IDF, Word2Vec (using word embeddings from Fast-Text, Twitter, and Google News), distilBERT embeddings, and Multilingual Universal Sentence Encoder embeddings. Different preprocessing steps were also assessed (such as removing punctuation, stopwords, and turning the text into lowercase). For each classifier, hyperparameter tuning was also applied. LinearSVC and Logistic Regression were tested with different C values (the regularization parameter), ranging from 1 to 100. MultinomialNB was tested with different alpha values (the smoothing parameter), ranging from 0.5 to 2. KNeighborsClassifier was tested with different K values (the number of neighbors), ranging from 5 to 10, and RandomForestClassifier was tested with different max_depth values (which determines how deep the trees can be), ranging from 3 to 5. Table 4.2 illustrates the best results per model and embedding strategy, based on the weighted average F1-score of all three labels.

Following table 4.2, the LinearSVC and the Logistic Regression models showed higher weighted F1-score for all embedding options. Moreover, the choice of embedding had a small difference (up to 5%) between each other for both Logistic Regression and LinearSVC but had higher differences for MultinomialNB, KNeighborsClassifier, and RandomForestClassifier. When

Model	Bag of Words	BERT embeddings	distilBERT embeddings	TF-IDF	Multilingual USE embeddings	Word2Vec
LinearSVC	0.61	0.62	0.64	0.60	0.63	0.59
LogisticRegression	0.62	0.61	0.62	0.62	0.63	0.58
MultinomialNB	0.53	0.45	0.48	0.54	0.53	0.27
KNeighborsClassifier	0.50	0.52	0.53	0.41	0.52	0.50
RandomForestClassifier	0.22	0.41	0.41	0.22	0.46	0.44

Table 4.2: Weighted F1-score for different models and feature representations predicting macro groups (positive, negative, and neutral).

comparing the feature representations, distilBERT and Multilingual USE word embeddings had the highest weighted F1-scores.

The good results for the linear models can be attributed to their better handling of high-dimensional data, as opposed to Random Forest and K Nearest Neighbors which generally perform worse with too many features. When comparing the linear models to Multinomial Naive Bayes, one explanation for the latter's worse results is its assumption of independence between features, which tends not to hold for longer texts. Additionally, LinearSVC and Logistic Regression tend to be more robust when dealing with irrelevant features in the text data. They can assign smaller weights or coefficients to less informative features, which helps in improving the model's generalization performance.

When comparing the feature representation options, Bag of Words, TF-IDF, and Word2Vec have lower F1-scores when compared to BERT, distilBERT, and Multilingual USE. This is probably due to the fact that the better-performing representations create embeddings based not only on the words themselves but also on the positioning and co-occurrence with other words in each sentence. That characteristic enables them to embed more meaning into the words, depending on how they are used in a sentence.

Tables 4.3 and 4.4 present the average training and prediction time (for the validation dataset) for each of the models and feature representations discussed above.

Model	Bag of Words	BERT embeddings	distilBERT embeddings	TF-IDF	Multilingual USE embeddings	Word2Vec
LinearSVC	25.2	242.5	276.0	16.7	189.3	106.9
RandomForestClassifier	43.4	101.8	57.5	38.0	55.9	29.0
LogisticRegression	154.4	4.4	5.3	27.1	3.1	2.1
KNeighborsClassifier	2.9	0.1	0.1	12.4	0.1	0.1
MultinomialNB	1.6	0.2	0.2	1.7	0.2	0.2

Table 4.3: Average training time in seconds per model/feature representation pair.

It is noticeable that even though LinearSVC is the model with the highest weighted F1-score,

Model	Bag of Words	BERT embeddings	distilBERT embeddings	TF-IDF	Multilingual USE embeddings	Word2Vec
KNeighborsClassifier	109.9	6.1	7.1	63.0	4.5	3.3
RandomForestClassifier	1.8	0.4	0.4	1.5	0.4	0.3
LogisticRegression	1.3	0.3	0.4	0.8	0.3	0.4
LinearSVC	1.4	0.3	0.3	0.9	0.4	0.3
MultinomialNB	0.6	0.2	0.2	0.7	0.2	0.2

Table 4.4: Average scoring time in seconds per model/feature representation pair.

it also presents the highest training time for all word embeddings options (BERT, distilBERT, and Multilingual USE). This is probably due to its time complexity being quadratic, along with the high dimensionality of the word embeddings. Random Forest Classifier was the second slowest model to train (on average), especially when using BERT embeddings, probably due to the number of trees used in training. With the exception of Logistic Regression with Bag of Words, all the other models were relatively fast to train.

Regarding scoring time, K Nearest Neighbors stood out as the model with higher times when using all feature representations, but exceptionally high times when using Bag of Words and TF-IDF. This model is known to take long to score since it has to compute each data point's neighbors in order to calculate its results. The remaining models took relatively low time to predict the validation data points.

For the goal of predicting positive, negative and neutral labels in the GoEmotions dataset, the models that performed better were LinearSVC with distilBERT embeddings and Logistic Regression with Multilingual USE embeddings. When analyzing their training and scoring times, LinearSVC showed a disadvantage of taking much longer to train, when compared to Logistic Regression. Even though the models themselves are not neural networks, they use a powerful text representation: word embeddings obtained through the training of neural networks.

4.4.2 Text classification on all GoEmotions labels

Before choosing the final sentiment groups, experiments were made using all sentiments provided by the GoEmotions dataset.

As shown in the previous section, the experiments for macro sentiments (positive, negative, and neutral) showed that the best non-neural networks model was LinearSVC with distilBERT embeddings, followed by Logistic Regression using Multilingual USE embeddings. Since our focus is on fine-grained sentiment classification, we proceeded with our experiments directly to classify the 28 GoEmotions labels when considering neural network models. Thus, the next steps in the experiments were to train both models with all 28 labels from the GoEmotions dataset and compare them to neural network approaches.

The metrics for the top-performing models from the previous section are shown in Table 4.5, which outlines the mean accuracy across labels, the mean F1-score, and the mean weighted F1-score, which is weighted based on the label frequency in the validation dataset. Notably, the weighted F1-score holds particular interest as it assigns more importance to the performance of minority classes, in contrast to the mean F1-score. It can be observed that both models have similar metrics between each other, with LinearSVC having slightly higher accuracy. The results per sentiment label for the LinearSVC model are shown in Table 4.6. As expected by the sentiment frequency distribution analyzed in section 4.1.1, some sentiments are more frequent in the training dataset and have higher metrics than others. It can also be observed in Table 4.6 that the recall of certain emotions is extremely low, such as annoyance, approval, disappointment, disapproval, excitement, nervousness, pride, and realization, and reaching zero in the case of grief and relief.

	Accuracy	Average F1-score	Weighted F1-score
LinearSVC (distilBERT embeddings)	0.51	0.31	0.45
Logistic Regression (multilingual USE embeddings)	0.49	0.31	0.45

Table 4.5: Performance metrics of linear models using word embeddings predicting the 28 emotion labels from the GoEmotions dataset.

For the Neural Networks approaches, the following models were initially attempted: a convolutional neural network (CNN), a recurrent neural network (RNN), a bidirectional recurrent neural network, a long-short-term memory neural network (LSTM), a gated recurrent unit recurrent neural network (GRU), whose architectures are illustrated in Figures 4.7, 4.8 and 4.9, and a fine-tuned distilBERT classifier. All of them used distilBERT word embeddings. The metrics of the models are shown in Table 4.7.

Table 4.7 shows that the distilBERT classifier presented the highest metrics overall. This was expected since it is a more complex model that was pre-trained with a previous dataset. When analyzing the other models, it can be observed that simple neural network models already outperform the linear models, with the exception of the Simple RNN and the Bidirectional RNN which performed worse than LinearSVC and Logistic Regression. The GRU RNN is the non-pre-trained neural network model with the highest metrics.

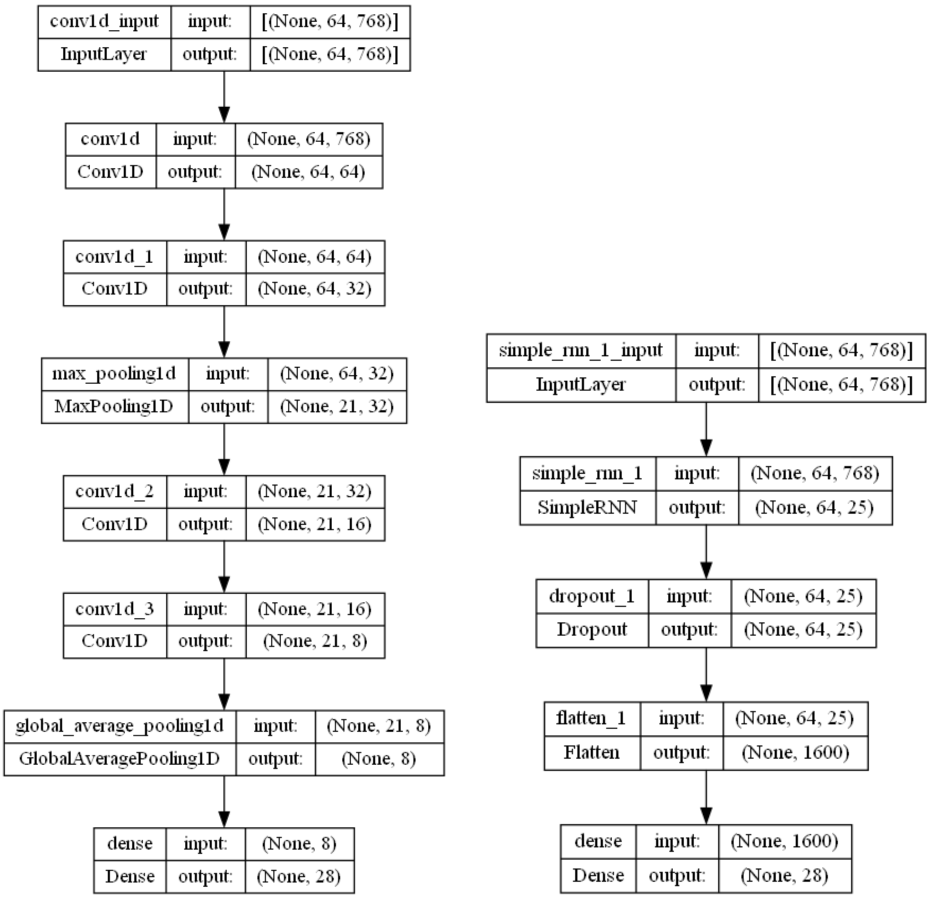
	precision	recall	F1-score	support
admiration	0.61	0.63	0.62	488
amusement	0.69	0.65	0.67	297
anger	0.45	0.34	0.39	192
annoyance	0.31	0.04	0.08	247
approval	0.44	0.09	0.15	355
caring	0.46	0.23	0.31	138
confusion	0.46	0.17	0.25	136
curiosity	0.44	0.41	0.43	205
desire	0.44	0.27	0.33	64
disappointment	0.15	0.02	0.04	129
disapproval	0.38	0.11	0.17	246
disgust	0.51	0.26	0.34	74
embarrassment	0.50	0.18	0.26	28
excitement	0.42	0.13	0.20	78
fear	0.49	0.36	0.42	74
gratitude	0.80	0.84	0.82	297
grief	0.00	0.00	0.00	10
joy	0.51	0.25	0.33	121
love	0.57	0.75	0.65	181
nervousness	0.20	0.09	0.13	11
neutral	0.46	0.86	0.60	1606
optimism	0.46	0.20	0.28	127
pride	1.00	0.11	0.20	9
realization	1.00	0.01	0.02	79
relief	0.00	0.00	0.00	8
remorse	0.47	0.47	0.47	47
sadness	0.37	0.26	0.31	84
surprise	0.37	0.15	0.21	95
macroavg	0.46	0.28	0.31	5426
weightedavg	0.49	0.50	0.45	5426

Table 4.6: Precision, recall, F1-score, and support per label for the LinearSVC with distilBERT embeddings model applied in the 28 GoEmotions labels.

Model	Embeddings	Average accuracy	Average F1-score	Weighted F1-score
LinearSVC	distilBERT	0.50	0.31	0.45
Logistic Regression	Multilingual USE	0.49	0.30	0.45
Simple CNN	distilBERT	0.52	0.32	0.47
Simple RNN	distilBERT	0.48	0.29	0.44
Bidirectional RNN	distilBERT	0.46	0.25	0.41
LSTM	distilBERT	0.53	0.37	0.5
GRU	distilBERT	0.54	0.40	0.51
distilBERT classifier	distilBERT	0.58	0.45	0.56

Table 4.7: Performance metrics per model and embedding pairs for all 28 emotion labels, evaluated on the GoEmotions dataset.

Table 4.8 presents the metric per label for the distilBERT classifier model. Similar to the LinearSVC classifier, some emotions still have very low recalls, such as grief and relief. However,

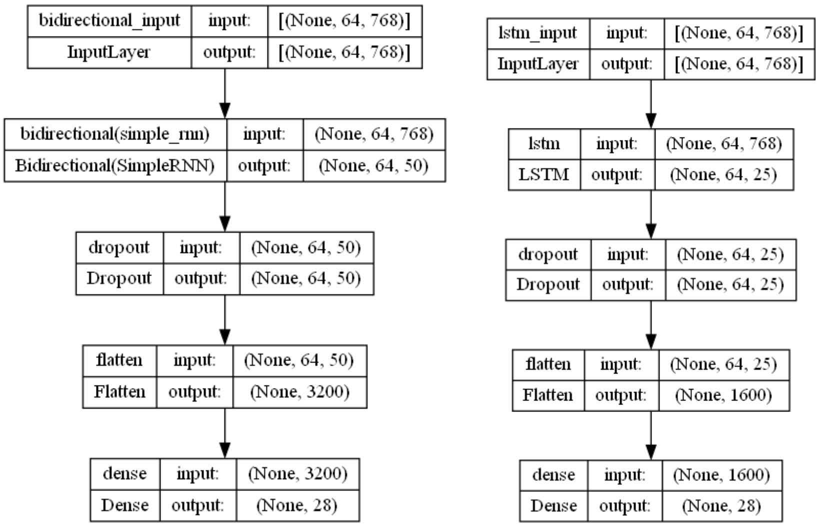

conv1d_input | input: | [(None, 64, 768)]
InputLayer | output: | [(None, 64, 768)]
conv1d | input: | (None, 64, 768)
Conv1D | output: | (None, 64, 64)
conv1d_1 | input: | (None, 64, 64)
Conv1D | output: | (None, 64, 32)
max_pooling1d | input: | (None, 64, 32)
MaxPooling1D | output: | (None, 21, 32)
conv1d_2 | input: | (None, 21, 32)
Conv1D | output: | (None, 21, 16)
conv1d_3 | input: | (None, 21, 16)
Conv1D | output: | (None, 21, 8)
global_average_pooling1d | input: | (None, 21, 8)
GlobalAveragePooling1D | output: | (None, 8)
dense | input: | (None, 8)
Dense | output: | (None, 28)
simple_rnn_1_input | input: | [(None, 64, 768)]
InputLayer | output: | [(None, 64, 768)]
simple_rnn_1 | input: | (None, 64, 768)
SimpleRNN | output: | (None, 64, 25)
dropout_1 | input: | (None, 64, 25)
Dropout | output: | (None, 64, 25)
flatten_1 | input: | (None, 64, 25)
Flatten | output: | (None, 1600)
dense | input: | (None, 1600)
Dense | output: | (None, 28)

(a) CNN (b) RNN

Figure 4.7: Architecture of the initial neural network models (CNN and RNN).

distilBERT had a worse recall for the label "pride" when compared to the LinearSVC model. Nevertheless, other labels were better predicted, such as annoyance, approval, disappointment, disapproval, and excitement. The best predicted emotions were admiration, amusement, fear, gratitude, joy, love, neutral, and remorse.

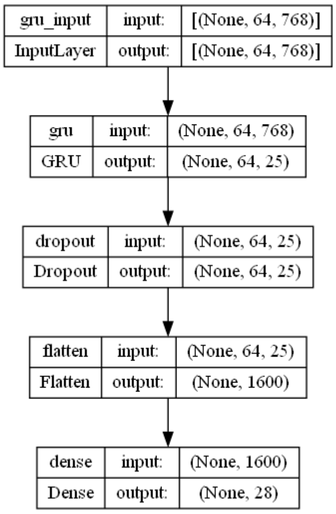
In order to better indicate to the model that mistakes within the same macro category (e.g. predicting sadness instead of grief) were more acceptable than mistakes across macro categories (e.g. predicting joy instead of grief), a custom loss function was developed based on distilBERT's default loss function. This custom loss multiplies each pair of predicted data point and true label by a weight as described by Table 4.9. The results of the experiment are shown in Table 4.10. Given that the model's loss was changed, it is expected to have worse F1-score and accuracy metrics, however it was expected to have a smaller "custom error", defined as an


bidirectional_input | input: | [(None, 64, 768)]
InputLayer | output: | [(None, 64, 768)]
bidirectional(simple_rnn) | input: | (None, 64, 768)
Bidirectional(SimpleRNN) | output: | (None, 64, 50)
dropout | input: | (None, 64, 50)
Dropout | output: | (None, 64, 50)
flatten | input: | (None, 64, 50)
Flatten | output: | (None, 3200)
dense | input: | (None, 3200)
Dense | output: | (None, 28)
lstm_input | input: | [(None, 64, 768)]
InputLayer | output: | [(None, 64, 768)]
lstm | input: | (None, 64, 768)
LSTM | output: | (None, 64, 25)
dropout | input: | (None, 64, 25)
Dropout | output: | (None, 64, 25)
flatten | input: | (None, 64, 25)
Flatten | output: | (None, 1600)
dense | input: | (None, 1600)
Dense | output: | (None, 28)

(c) Bidirectional RNN

(d) LSTM

Figure 4.8: Architecture of the initial neural networks models (Bidirectional RNN and LSTM).


gru_input | input: | [(None, 64, 768)]
InputLayer | output: | [(None, 64, 768)]
gru | input: | (None, 64, 768)
GRU | output: | (None, 64, 25)
dropout | input: | (None, 64, 25)
Dropout | output: | (None, 64, 25)
flatten | input: | (None, 64, 25)
Flatten | output: | (None, 1600)
dense | input: | (None, 1600)
Dense | output: | (None, 28)

(e) GRU RNN

Figure 4.9: Architecture of the initial neural networks models (GRU RNN).

error metric based on the weight matrix proposed. Yet the custom error was still larger for the weighted loss, as indicated in the third column of the table. One possibility for the cause of this result is the magnitude and range given by the multipliers. Subsequent trials with more complex weight matrices yielded no enhancement compared to the default model's outcomes.

	precision	**recall**	**F1-score**	**support**
admiration	0.66	0.80	0.73	488
amusement	0.75	0.85	0.80	297
anger	0.44	0.49	0.46	192
annoyance	0.36	0.26	0.30	247
approval	0.44	0.28	0.34	355
caring	0.47	0.41	0.44	138
confusion	0.49	0.36	0.42	136
curiosity	0.47	0.49	0.48	205
desire	0.62	0.53	0.57	64
disappointment	0.41	0.26	0.32	129
disapproval	0.44	0.31	0.36	246
disgust	0.44	0.49	0.46	74
embarrassment	0.65	0.54	0.59	28
excitement	0.36	0.28	0.32	78
fear	0.60	0.64	0.62	74
gratitude	0.87	0.83	0.85	297
grief	0.00	0.00	0.00	10
joy	0.55	0.50	0.52	121
love	0.65	0.82	0.73	181
nervousness	1.00	0.09	0.17	11
neutral	0.59	0.68	0.63	1606
optimism	0.55	0.54	0.54	127
pride	0.00	0.00	0.00	9
realization	0.52	0.19	0.28	79
relief	0.00	0.00	0.00	8
remorse	0.61	0.70	0.65	47
sadness	0.44	0.52	0.48	84
surprise	0.47	0.53	0.50	95
macro avg	0.49	0.44	0.45	5426
weighted avg	0.56	0.58	0.56	5426

Table 4.8: distilBERT classifier metrics per label for the 28 emotion labels, evaluated on the GoEmotions dataset.

For the 28 emotion labels, the best results achieved were from the distilBERT classifier. However, some emotions did not achieve desirable results, such as pride, grief, relief, nervousness, and realization, that had F1-scores below 0.3. The next section continues the experiments by grouping the labels into the 6 chosen emotion groups.

4.4.3 Text classification on 6 sentiment groups

This subsection describes the experiments done for the target being the 6 emotion groups described in Section 4.2: repudiation, sadness, neutral, curiosity, appreciation, and positive experience. Given the good results of a fine-tuned model in the previous section, the experi-

	Negative sentiment	Neutral sentiment	Positive sentiment
Negative sentiment	0.25	0.5	1
Neutral sentiment	0.5	0.25	0.5
Positive sentiment	1	0.5	0.25

Table 4.9: Proposed weight matrix for the custom loss.

	Accuracy	Average F1-score	Custom error metric
default	0.58	0.44	0.21
with weighted loss	0.55	0.43	0.22

Table 4.10: Performance metrics of the distilBERT classifier comparing the usage of a weighted loss function, evaluated on the GoEmotions dataset.

ments will be focused on different transformer models, being compared to a neural network using Multilingual Universal Sentence Encoder (USE) embeddings. The transformers assessed were: distilBERT, BERT, RoBERTa, XLNET, Multilingual BERT (mBERT), and Language-Agnostic BERT Sentence Embedding (LABSE). Even though there are multiple multilingual models in this evaluation, they are assessed for English texts only in this subsection, while multilingual evaluations are presented in Section 4.6.

Table 4.11 presents the metrics for each of the models evaluated. It can be observed that the transformer models with their own embeddings perform better than the neural network with Multilingual Universal Sentence Encoder embeddings. This is expected, since the transformer models were pre-trained on a larger corpus of data, as opposed to the simple neural network built for the mUSE embeddings. Further investigation could be performed using the mUSE embeddings in more complex neural networks, however it was decided to prioritize the research using only the transformer models since they showed more promising results.

Model	Accuracy	Average F1-score	Weighted F1-score
distilBERT	0.65	0.63	0.65
BERT-base	0.66	0.64	0.66
BERT-large	0.66	0.64	0.66
RoBERTa	0.67	0.65	0.67
mBERT	0.65	0.64	0.65
XLNET	0.66	0.64	0.66
LABSE	0.66	0.64	0.66
mUSE	0.57	0.52	0.57

Table 4.11: Performance metrics of the transformer models trained on the 6 sentiment groups, evaluated on the GoEmotions dataset.

It can also be observed that the transformers perform relatively similarly amongst each other, with RoBERTa presenting slightly higher metrics overall. This can be due to the fact that RoBERTa was pre-trained using a larger corpus. There was one drawback regarding RoBERTa and XLNET though: when training the models, such models required a higher tier of GPU and RAM than the one available by the provided training machine. These models had to be run in a machine with an A100 GPU with 32GB of RAM, as opposed to the free and standard T4 GPU with 12.7GB of RAM provided by Google Colaboratory[¶]. Due to these limitations in the working environment, it was decided to proceed further experiments with the other models that are less computationally intensive.

When comparing the distilBERT classifier results between the 6 group classification (shown in Table 4.11) and the 28 emotion labels classification (shown in Table 4.7, it can be noted that all metrics (average F1-score, weighted F1-score, and accuracy) are higher at least by 8 percentage points for the classification using the 6 label groups. This is expected for a few reasons. First of all, the emotion labels that have low frequencies and showed F1-scores below 0.3 in the 28-label classification (such as grief, relief, and pride) are merged together in the 6-label classification (in groups sadness, positive experience, and appreciation, respectively) along with emotion labels that are more frequent in the GoEmotions dataset and consequently present better metrics. Secondly, just like humans, the model might have a hard time differentiating between similar or ambiguous emotions, thus having worse metrics. Finally, a higher number of labels increases the complexity of the classification problem. The model needs to distinguish between a larger number of classes, and this can make it more challenging for the model to generalize well to unseen data.

With the aim of improving the results reached so far, the training dataset was augmented by adding the WASSA 2021 shared task dataset to it. The WASSA labels were replaced by their respective labels in the 6-label emotion group. The final training data was then shuffled to avoid clustering all the WASSA data in one training epoch. In order to compare the results, the validation dataset remains the same across all experiments, it remains as the GoEmotions predetermined validation dataset. The results of this experiment are shown in Table 4.12. When comparing these results with the ones obtained by training without the WASSA dataset shown in Table 4.11, it can be observed that the results are very similar among the two approaches, and the results using only the GoEmotions trained models are slightly better. This could be justified by the fact that the validation dataset is sourced from the GoEmotions dataset, following the same distribution as the training dataset. The data augmentation with the WASSA dataset

[¶] `https://colab.research.google.com/`

is further evaluated on manually annotated real-word web texts in Section 4.5.

Model	Accuracy	Average F1-score	Weighted F1-score
distilBERT	0.65	0.63	0.65
BERT-base	0.65	0.64	0.65
BERT-large	0.66	0.64	0.66
RoBERTa	0.66	0.64	0.66
mBERT	0.65	0.63	0.65
XLNET	0.66	0.63	0.65
LABSE	0.66	0.65	0.66
mUSE	0.56	0.52	0.56

Table 4.12: Performance metrics of models trained with GoEmotions + WASSA 2021 datasets for the 6 emotions groups as labels.

4.4.4 Multi-label experiments

Given the fact that the GoEmotions dataset is multi-labeled and even the human difficulty of providing only one label to a text (see Section 4.1.3 for more details), it is intuitive to think that multi-labeled modeling would be more reasonable and would probably work better than the models presented so far. Therefore, this section details the experiment done using multi-label modeling.

As in Subsection 4.4.3, several models were trained and evaluated, this time using the multi-labeled data from the GoEmotions dataset. Table 4.13 presents the best results obtained with multi-label approaches, after iterations of trying to use a weighted custom loss, providing sample weights, and training with the additional WASSA 2021 dataset. The best results were obtained by training the model with an unweighted binary cross entropy loss function, without providing sample weights to the model's training, and without the WASSA dataset in the training of the model. The F1-scores shown in Table 4.13 were firstly independently calculated and then averaged (without weights in the first column and with weights in the second column). The subset accuracy was omitted due to its very low results (close to zero in almost all cases), since it **only** outputs a correct prediction if **all** sentiments were correctly predicted. For example, the subset accuracy would count it as correct when exactly predicting that a task belongs to two specific labels, and only those labels (a task that is extremely hard even for humans, as noticed by the author when manually labeling the web data and comparing her labels with the ones from other annotators).

Model	Average F1-score	Weighted F1-score
distilBERT	0.37	0.42
BERT-base	0.34	0.40
BERT-large	0.32	0.38
RoBERTa	0.38	0.44
mBERT	0.35	0.40
XLNET	0.32	0.38
LABSE	0.36	0.42

Table 4.13: Best metrics obtained for multi-label models, evaluated on the GoEmotions dataset.

When comparing the multi-label results in Table 4.13 with the ones shown in Table 4.11, it is evident that the multi-label approach yielded worse results. This could be the case of needing more multi-labeled data in order to yield more significant results, or changing the classification problem to associate a single label for each sentence instead of the entire text snippet (as a multi-labeled text could present different sentiments over different sentences). Since it wasn't a business priority to output more than one label in the final product, this experiment was discontinued. Further investigation is necessary to understand how to properly model a multi-label classifier.

4.5 Evaluation on web data

Given the results obtained in Section 4.4.3, the next step was to evaluate the models on the data where they will be effectively applied: the web data provided by the company's scraper. After applying the models in the manually labeled data points sampled from the company's dataset, the results obtained were gathered and presented in Table 4.14.

Model	Accuracy	Average F1-score	Weighted F1-score
distilBERT	0.45	0.26	0.38
BERT-base	0.46	0.31	0.41
BERT-large	0.48	0.37	0.44
RoBERTa	0.48	0.35	0.43
mBERT	0.45	0.25	0.37
mUSE	0.39	0.21	0.31
LABSE	0.43	0.25	0.35

Table 4.14: Performance metrics of the models applied to the manually labeled web data.

Based on the information in Table 4.14, it is clear that the BERT-large model provided the highest accuracy and F1-scores, even higher than RoBERTa (a model that was more expensive to train).

In order to further evaluate the BERT model, Table 4.15 presents the precision and recall for each of the emotion groups. Additionally, Figure 4.10 illustrates the confusion matrix given by the model's results. From these results, it becomes apparent that neutral and appreciation are the groups with the highest recall but also the highest bias in the model. This is probably due to the higher frequency of these two groups in the training dataset.

	precision	recall	F1-score	support
repudiation	1	0.14	0.25	14
sadness	0.83	0.19	0.31	26
neutral	0.54	0.71	0.61	85
curiosity	0.43	0.18	0.26	33
appreciation	0.35	0.73	0.47	37
positive experience	0.47	0.23	0.31	30
macro avg	0.6	0.36	0.37	225
weighted avg	0.55	0.48	0.44	225

Table 4.15: Metrics per emotion group for the BERT-large model, evaluated on the manually annotated web data.

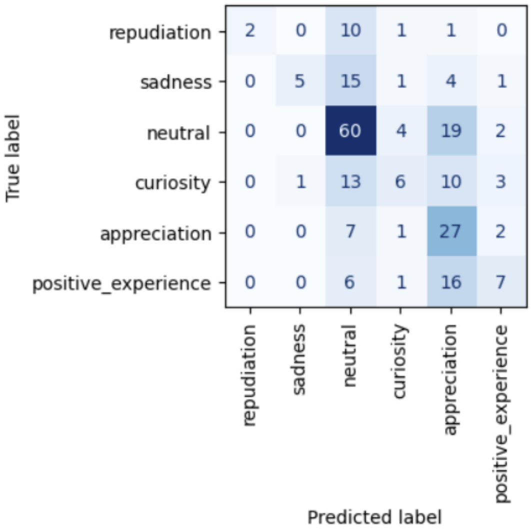


Figure 4.10: Confusion matrix of the BERT-large model applied on the manually labeled web data.

It can be seen in the results that the models predict most data points as neutral and appreciation. One way to remediate this issue is to work on the thresholds for the decision-making of each label, which was the next experiment done. The thresholds chosen were the mean of the prediction probabilities computed in the training dataset. The mean and standard deviation per label of the training dataset predictions were then used to re-calibrate the prediction probabilities per label of the scoring data in the following manner:

$$y_{pred}[label] = \frac{y_{pred}[label] - y'_{mean}[label]}{y'_{stdev}[label]} \tag{4.1}$$

where y_{pred} represents the prediction probabilities given by the scoring of the model for each of the 6 labels and y' represents the training dataset prediction vector for the respective labels. This model calibration proves useful not only for balancing the prediction probabilities but could also reduce the bias towards the training dataset label distribution since it doesn't necessarily follow the real data label distribution.

After calibrating the model, the maximum prediction is chosen as the output. This way, the highest value amongst the predictions is still chosen, but only after balancing out the training dataset bias. The results of this modification are presented in Table 4.16.

	Accuracy	**Average F1-score**	**Weighted F1-score**
distilBERT	0.48	0.43	0.47
BERT-base	0.47	0.47	0.48
BERT-large	0.43	0.46	0.44
RoBERTa	0.47	0.48	0.48
mBERT	0.43	0.41	0.43
mUSE	0.33	0.26	0.32
LABSE	0.46	0.39	0.43

Table 4.16: Performance metrics of the models applied to the manually labeled web data after re-calibrating the prediction probabilities.

Upon reviewing Table 4.16, it becomes apparent that the accuracy of some models decreased, but their F1-scores (both averages and weighted averages) increased. Re-calibrating changed the values of the probabilities, increasing the recall and reducing the precision of the models. Additionally, it can be observed that the models BERT-base and RoBERTa have the highest F1-scores. Since RoBERTa's training is more expensive and BERT-base has similar results, we have decided to choose BERT-base as the final model.

Additionally to model calibration, in order to solve the model's bias towards the neutral and appreciation labels, the training dataset could be artificially balanced. By either downsampling the most frequent labels or over-sampling the other labels, the over-represented categories would become balanced with the others. However, this technique could lead to a loss of valuable information that might have been present in the discarded samples. This is suggested to be investigated in future work.

In order to compare the results with the ones shown in Table 4.15, the metrics per label and confusion matrix of the BERT-large model are shown in Table 4.17 and Figure 4.11. Finally, the results for the BERT-base model are shown in Table 4.18 and Figure 4.12.

	precision	recall	F1-score	support
repudiation	0.75	0.64	0.69	14
sadness	0.53	0.38	0.44	26
neutral	0.66	0.36	0.47	85
curiosity	0.41	0.42	0.42	33
appreciation	0.29	0.62	0.39	37
positive experience	0.3	0.33	0.32	30
macro avg	0.49	0.46	0.46	225
weighted avg	0.5	0.43	0.44	225

Table 4.17: Metrics per emotion group for the BERT-large model after re-calibrating the prediction probabilities, evaluated on the manually annotated web data.

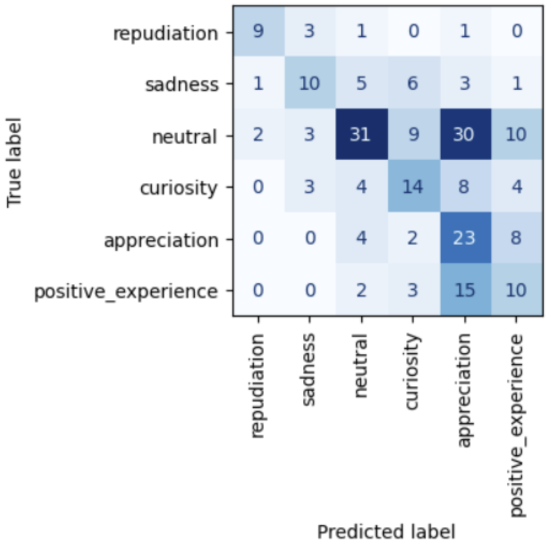
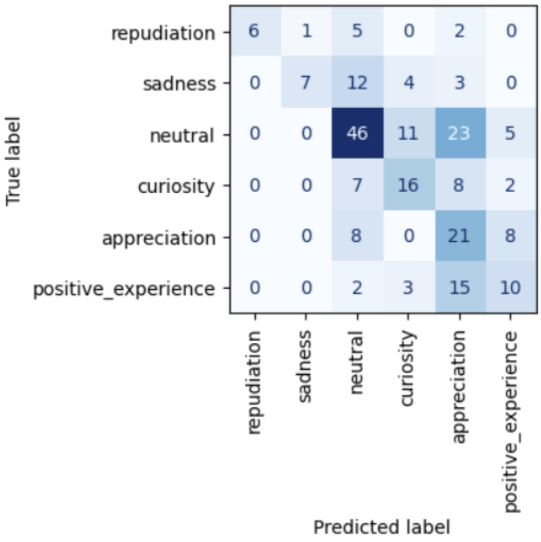


Figure 4.11: Confusion matrix of the BERT-large model applied on the manually labeled web data after re-calibrating the prediction probabilities.

	precision	recall	F1-score	support
repudiation	1	0.43	0.6	14
sadness	0.88	0.27	0.41	26
neutral	0.57	0.54	0.56	85
curiosity	0.47	0.48	0.48	33
appreciation	0.29	0.57	0.39	37
positive experience	0.4	0.33	0.36	30
macro avg	0.6	0.44	0.47	225
weighted avg	0.55	0.47	0.48	225

Table 4.18: Metrics per emotion group for the BERT-base model after re-calibrating the prediction probabilities, evaluated on the manually annotated web data.



	repudiation	sadness	neutral	curiosity	appreciation	positive_experience
repudiation	6	1	5	0	2	0
sadness	0	7	12	4	3	0
neutral	0	0	46	11	23	5
curiosity	0	0	7	16	8	2
appreciation	0	0	8	0	21	8
positive_experience	0	0	2	3	15	10

Figure 4.12: Confusion matrix of the BERT-base model applied on the manually labeled web data after re-calibrating the prediction probabilities.

As a business decision, it was decided to prioritize models that correctly predict negative sentiments more than positive ones since it is more harmful to publish an ad on a website falsely seen as positive, but actually conveying negative sentiments (and thus possibly harming the reputation of the advertising brand), rather than avoiding publishing an ad in one website that conveys positive emotions because it was misclassified with a negative sentiment. Hence, the BERT-large model was selected as the final choice from among the options with the highest F1-scores, primarily based on the highest recall for the "repudiation" and "sadness" labels.

4.6 Multilingual evaluation on web data

The final goal of this project is to evaluate if the models developed can be used for a language other than English without further training. Given that the company has multiple clients in Italy, Italian was chosen as the language to validate the language-agnostic models.

In order to perform such evaluation, the web data that was previously manually annotated was then translated into Italian using Google Translate and input into the models. Even though there are Italian datasets in the literature that could be used instead of the translated data chosen, these datasets [86, 87, 88] don't provide the same data distribution and variety as what is expected from the real web data and don't have the same diversity of emotion labels as the ones suggested in this work.

Table 4.19 presents the results of the language-agnostic models used in the experiments, while Table 4.20 presents the results after re-calibrating the predictions according to Equation 4.1. It can be observed that re-calibrating the predictions increased the average F1-score and the weighted F1-score (with the exception of mUSE, where it was merely maintained), at the cost of decreasing accuracy. Nevertheless, it is clear that the results for the same models on Italian texts are worse than for English data, as expected since the models were trained with English text. These results could also be caused by partially incorrect translations of the text, which could alter its overall sentiment in the Italian language.

	Accuracy	**Average F1-score**	**Weighted F1-score**
mUSE	0.37	0.18	0.28
LABSE	0.40	0.18	0.29
mBERT	0.42	0.23	0.32

Table 4.19: Performance metrics of the multilingual models on manually annotated web data translated to Italian.

	Accuracy	**Average F1-score**	**Weighted F1-score**
mUSE	0.32	0.22	0.28
LABSE	0.38	0.32	0.35
mBERT	0.36	0.32	0.37

Table 4.20: Performance metrics of the multilingual models on manually annotated web data translated to Italian after re-calibrating the predictions.

	precision	**recall**	**F1-score**	**support**
repudiation	0.40	0.14	0.21	14
sadness	0.33	0.08	0.12	26
neutral	0.43	0.92	0.58	85
curiosity	0.00	0.00	0.00	33
appreciation	0.41	0.30	0.34	37
positive experience	0.50	0.07	0.12	30
macro avg	0.34	0.25	0.23	225
weighted avg	0.36	0.42	0.32	225

Table 4.21: Metrics per label for the Multilingual BERT model applied in manually annotated web data translated to Italian.

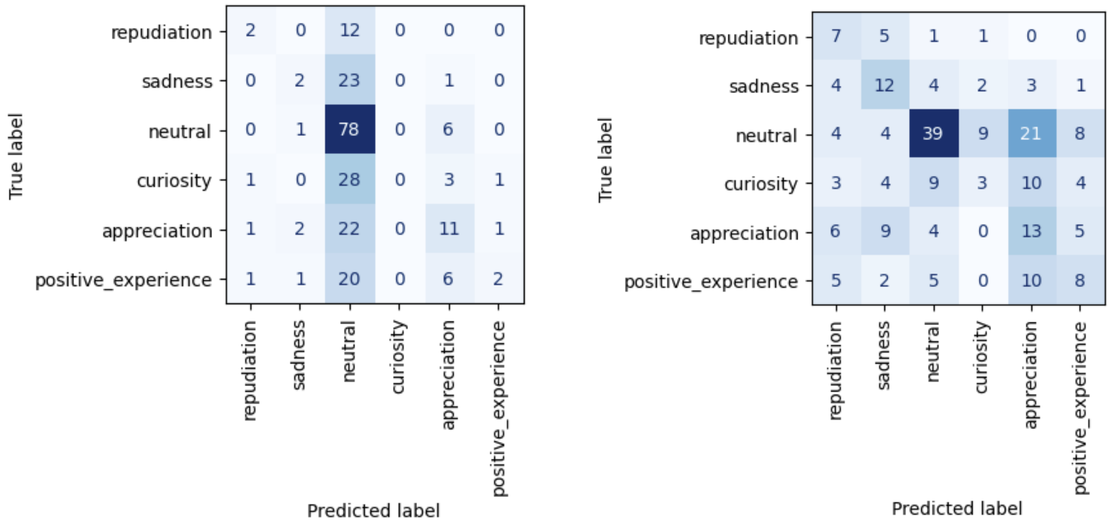
	precision	**recall**	**F1-score**	**support**
repudiation	0.24	0.50	0.33	14
sadness	0.33	0.46	0.39	26
neutral	0.63	0.46	0.53	85
curiosity	0.20	0.09	0.13	33
appreciation	0.23	0.35	0.28	37
positive experience	0.31	0.27	0.29	30
macro avg	0.32	0.35	0.32	225
weighted avg	0.40	0.36	0.37	225

Table 4.22: Metrics per label for the Multilingual BERT model applied in manually annotated web data translated to Italian after re-calibrating the predictions.

Tables 4.21 and 4.22 present the precision, recall, and F1-score metrics per label for the Multilingual BERT model before and after re-calibrating, respectively. The calibration parameters used are shown in Table 4.23. We can see from the parameters that the mean of the predictions on the training dataset for the "neutral" label is much higher than the other means. The model calibration intends to balance that discrepancy. Figure 4.13 illustrates the confusion matrices of both models. It is clear from the confusion matrices that re-calibrating the predictions reduced the bias of the models towards the neutral label.

Label	Repudiation	Sadness	Neutral	Curiosity	Appreciation	Positive experience
y'_{mean}	0.4	0.3	0.7	0.4	0.5	0.4
y'_{stdev}	0.3	0.3	0.2	0.3	0.3	0.3

Table 4.23: Calibration parameters per label for the Multilingual BERT model



(a) Default model

(b) Calibrated model

Figure 4.13: Confusion matrices of the Multilingual BERT model applied on the manually labeled web data translated to Italian before and after re-calibrating the prediction probabilities.

One final experiment was conducted by augmenting the training dataset by adding the WASSA 2021 data and shuffling both datasets together. The results of the retrained models are shown in Tables 4.24 (showing results without re-calibrating) and 4.25 (showing results after re-calibrating). When comparing the results without re-calibrating, as shown in Tables 4.19 and 4.24, it can be observed that the Simple Neural Network with Multilingual USE embeddings and the Language-Agnostic BERT Sentence Encoder models have improved their metrics after being trained with the GoEmotions and WASSA datasets, as opposed to the multilingual BERT model. On the other hand, the Multilingual BERT model trained with only GoEmotions data still performs better than the best version of the other models.

Model	**Accuracy**	**Average F1-score**	**Weighted F1-score**
mUSE	0.41	0.28	0.32
LABSE	0.42	0.24	0.30
mBERT	0.41	0.23	0.31

Table 4.24: Performance metrics of the models trained with WASSA data evaluated on the manually annotated web data translated to Italian before re-calibrating the predictions.

As an end result, the Multilingual BERT calibrated model trained with the GoEmotions dataset showed the best results for Italian web data. It is important to note that such modifications on the model after the first evaluation of the annotated web data might lead to the model

Model	Accuracy	Average F1-score	Weighted F1-score
mUSE	0.36	0.26	0.31
LABSE	0.36	0.34	0.35
mBERT	0.32	0.25	0.30

Table 4.25: Performance metrics of the models trained with WASSA data evaluated on the manually annotated web data translated to Italian after re-calibrating the predictions.

becoming too tailored to this specific data, overfitting to such data. This risk stems from the absence of a test set consisting of manually annotated web data, which could serve as a means of evaluating the model. Hence, it becomes imperative to monitor the model's performance on new, unseen data and ascertain whether its performance metrics continue to align with those observed in this study.

This chapter has described several experiments using different models and sets of labels in order to determine the model with best results, for both English and Italian texts. The BERT-large model for English data and the Multilingual BERT model for Italian data were chosen as the best-fit models for the task of predicting web data between 6 groups of emotion labels. BERT-large has achieved a weighted F1-score of 44% and accuracy of 48% amongst the 6 emotion categories in the manually annotated English web data, while Multilingual BERT achieved a weighted F1-score of 37% and accuracy of 36% over the 6 emotion categories in the Italian translated web data, and a weighted F1-score and accuracy of 43% in the English web data.

Reflecting on our third research question (Is it feasible to employ a language-agnostic model for this task? If so, how does it compare to a single language model?) we can conclude that it is possible and feasible to build language-agnostic models pre-trained on multilingual data, yet fine-tuned solely on English data. Such models perform relatively similarly to single-language models on English evaluation data. However, it is noticeable that there is a decay in model performance when comparing those language-agnostic models between English and Italian test data.

5
Conclusion

In this work, an approach to sentiment classification using finer-grained sentiment groups balancing out positive and negative sentiments was proposed. Six sentiment groups were proposed by grouping the 28 emotion labels presented in the GoEmotions dataset, namely "repudiation", "sadness", "neutral", "curiosity", "appreciation", and "positive experience". As the final goal of this work was to apply it to web data, which is different from the training dataset used, a portion of real-world web data was selected and manually annotated in order to evaluate the models that were considered. By experimenting with several machine learning models including logistic regression, multinomial naïve Bayes, support vector machines, convolutional and recurrent neural networks, and transformers, it was concluded that Transformers models perform usually better than the others tested. The model BERT-large achieved the highest weighted F1-score of 44% and an accuracy of 48% on the web dataset, among the 6 emotion categories.

Another goal of this study was to use language-agnostic models in order to classify texts from non-English websites. To achieve this goal, three language-agnostic models and embeddings (Multilingual BERT, Multilingual Universal Sentence Encoder, and Language-Agnostic BERT Sentence Embeddings) were evaluated on the manually annotated web data translated into Italian, the language of one of the company's clients. It was concluded that although their performances in Italian texts were lower than in English texts, the models achieved feasible results. Multilingual BERT, the model with the highest weighted F1-score on the 6 emotion categories, reached a weighted F1-score of 37% and accuracy of 36% on the collected web data.

Overall, this study was able to research several models, evaluate them in a portion of manually annotated real-world web data, and obtain fairly good results over different emotion groups, finer-grained that “positive”, “negative” and “neutral” categories. It has also been able to identify and evaluate multilingual models in order to obtain the best models that can be applied to multiple languages while still being trained only on English data.

5.1 Future work

Regarding future developments, it is suggested to experiment with the usage of another custom loss function to take into account the difference between positive and negative labels, for example, the Label-Aware Contrastive Loss function [80]. Also, down-sampling the over-represented emotion categories could be researched in order to improve the models’ bias towards such categories. Furthermore, reducing model size could be important to reduce processing time, and thus execution costs, thus the process of knowledge distillation could be implemented and experimented as well.

One other point of improvement is not on the model itself, but on the web scraper that collects the data. Future development is suggested to improve the scraper by better separating real content from noise, and also collecting the text data maintaining uppercase letters and punctuation.

Regarding the implementation of the researched model in production, future works also include actually implementing the model into the data pipeline and evaluating the costs of the execution of the model in a GPU environment, as opposed to a CPU one.

References

[1] Machine Elf 1735, "Robert plutchik's wheel of emotions," February 2011, (accessed: 26.07.2023). [Online]. Available: https://commons.wikimedia.org/w/index.php?curid=13285286

[2] Y.-S. Seo and J.-H. Huh, "Automatic emotion-based music classification for supporting intelligent iot applications," Scientific Figure on ResearchGate, 2019, (accessed: 26.07.2023). [Online]. Available: https://www.researchgate.net/figure/Russells-circumplex-model-The-circumplex-model-is-developed-by-James-Russell-In-the_fig1_330817411

[3] A. S. Cowen and D. Keltner, "Self-report captures 27 distinct categories of emotion bridged by continuous gradients," *Proceedings of the National Academy of Sciences*, vol. 114, no. 38, pp. E7900–E7909, 2017. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.1702247114

[4] Larhmam, "File:svm margin.png - wikimedia commons," https://commons.wikimedia.org/wiki/File:SVM_margin.png, October 2018, (accessed: 08.08.2023).

[5] B. Luaphol, J. Polpinij, and M. Kaenampornpan, "Mining bug report repositories to identify significant information for software bug fixing," *Applied Science and Engineering Progress*, 03 2021.

[6] fdeloche, "Structure of rnn," https://commons.wikimedia.org/wiki/File:Recurrent_neural_network_unfold.svg/, June 2017, (accessed: 08.08.2023).

[7] J. Uszkoreit, "Transformer: A novel neural network architecture for language understanding – google research blog," https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html, August 2017, (accessed: 26.07.2023).

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need." [Online]. Available: http://arxiv.org/abs/1706.03762

[9] Simpson, Ian, "Real-Time Bidding (RTB) Programmatic: One and the Same?" 2021, (accessed: 18.07.2023). [Online]. Available: https://clearcode.cc/blog/difference-between-rtb-programmatic/

[10] D. Alon and J. Ko, "GoEmotions: A Dataset for Fine-Grained Emotion Classification," 2021, (accessed: 18.07.2023). [Online]. Available: https://ai.googleblog.com/2021/10/goemotions-dataset-for-fine-grained.html

[11] D. Demszky, D. Movshovitz-Attias, J. Ko, A. Cowen, G. Nemade, and S. Ravi, "GoEmotions: A dataset of fine-grained emotions." [Online]. Available: http://arxiv.org/abs/2005.00547

[12] P. Ekman, "Are there basic emotions?" *Psychological Review*, vol. 99, no. 3, pp. 550–553, 1992.

[13] R. Plutchik, "A general psychoevolutionary theory of emotion," in *Theories of emotion*. Elsevier, 1980, pp. 3–33.

[14] J. Russell, "A circumplex model of affect," *Journal of Personality and Social Psychology*, vol. 39, pp. 1161–1178, 12 1980.

[15] A. Cowen and D. Keltner, "What the face displays: Mapping 28 emotions conveyed by naturalistic expression," *American Psychologist*, vol. 75, 06 2019.

[16] S. S. T. Hemakala, "Advanced classification method of twitter data using sentiment analysis for airline service," *International Journal of Computer Sciences and Engineering*, vol. 6, pp. 331–335, 7 2018.

[17] T. Dholpuria, Y. Rana, and C. Agrawal, "A sentiment analysis approach through deep learning for a movie review," in *2018 8th International Conference on Communication Systems and Network Technologies (CSNT)*. IEEE, 2018, pp. 173–181.

[18] A. I. Saad, "Opinion mining on us airline twitter data using machine learning techniques," in *2020 16th international computer engineering conference (ICENCO)*. IEEE, 2020, pp. 59–63.

[19] P. Harjule, A. Gurjar, H. Seth, and P. Thakur, "Text classification on twitter data," 02 2020, pp. 160–164.

[20] F. Jemai, M. Hayouni, and S. Baccar, "Sentiment analysis using machine learning algorithms," *2021 International Wireless Communications and Mobile Computing (IWCMC)*, pp. 775–779, 2021.

[21] A. M. Rahat, A. Kahir, and A. K. M. Masum, "Comparison of naive bayes and svm algorithm based on sentiment analysis using review dataset," in *2019 8th International Conference System Modeling and Advancement in Research Trends (SMART)*. IEEE, 2019, pp. 266–270.

[22] A. Tariyal, S. Goyal, and N. Tantububay, "Sentiment analysis of tweets using various machine learning techniques," 12 2018, pp. 1–5.

[23] K. Dhola and M. Saradva, "A comparative evaluation of traditional machine learning and deep learning classification techniques for sentiment analysis," in *2021 11th international conference on cloud computing, data science & engineering (Confluence)*. IEEE, 2021, pp. 932–936.

[24] N. C. Dang, M. N. Moreno-García, and F. De la Prieta, "Sentiment analysis based on deep learning: A comparative study," vol. 9, no. 3, p. 483. [Online]. Available: https://www.mdpi.com/2079-9292/9/3/483

[25] I. E. Fattoh, F. Kamal Alsheref, W. M. Ead, A. M. Youssef *et al.*, "Semantic sentiment classification for covid-19 tweets using universal sentence encoder," *Computational intelligence and neuroscience*, vol. 2022, 2022.

[26] Y. Yang, "Convolutional neural networks with recurrent neural filters," *arXiv preprint arXiv:1808.09315*, 2018.

[27] M. Rhanoui, M. Mikram, S. Yousfi, and S. Barzali, "A cnn-bilstm model for document-level sentiment analysis," *Machine Learning and Knowledge Extraction*, vol. 1, no. 3, pp. 832–847, 2019. [Online]. Available: https://www.mdpi.com/2504-4990/1/3/48

[28] M. S. Hossen, A. H. Jony, T. Tabassum, M. T. Islam, M. M. Rahman, and T. Khatun, "Hotel review analysis for the prediction of business using deep learning approach," in *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)*. IEEE, 2021, pp. 1489–1494.

[29] M. Munikar, S. Shakya, and A. Shrestha, "Fine-grained sentiment classification using bert," in *2019 Artificial Intelligence for Transforming Business and Society (AITB)*, vol. 1, 2019, pp. 1–5.

[30] A. F. Adoma, N.-M. Henry, and W. Chen, "Comparative analyses of bert, roberta, distilbert, and xlnet for text-based emotion recognition," in *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 2020, pp. 117–121.

[31] S. A. G. W. Aqsa Younas, Raheela Nasim and F. Qi, "Sentiment analysis of code-mixed roman urdu-english social media text using deep learning approaches," pp. 66–71, 2020.

[32] S. Smetanin and M. Komarov, "Deep transfer learning baselines for sentiment analysis in russian," *Information Processing & Management*, vol. 58, no. 3, p. 102484, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306457320309730

[33] D. Cortiz, "Exploring transformers models for emotion recognition: a comparision of BERT, DistilBERT, RoBERTa, XLNET and ELECTRA," in *2022 3rd International Conference on Control, Robotics and Intelligent System*. ACM, pp. 230–234. [Online]. Available: https://dl.acm.org/doi/10.1145/3562007.3562051

[34] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[35] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[36] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information." [Online]. Available: http://arxiv.org/abs/1607.04606

[37] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[38] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter." [Online]. Available: http://arxiv.org/abs/1910.01108

[39] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach." [Online]. Available: http://arxiv.org/abs/1907.11692

[40] D. Cer, Y. Yang, S. yi Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, Y.-H. Sung, B. Strope, and R. Kurzweil, "Universal sentence encoder," 2018.

[41] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," 2020.

[42] S. Tafreshi, O. D. Clercq, V. Barriere, S. Buechel, J. Sedoc, and A. Balahur, "WASSA 2021 shared task: Predicting empathy and emotion in reaction to news stories," 2021.

[43] B. Zhao, *Web Scraping*. Cham: Springer International Publishing, 2017, pp. 1–3. [Online]. Available: https://doi.org/10.1007/978-3-319-32001-4_483-1

[44] A. Cowen, D. Sauter, J. L. Tracy, and D. Keltner, "Mapping the passions: Toward a high-dimensional taxonomy of emotional experience and expression," *Psychological Science in the Public Interest*, vol. 20, no. 1, pp. 69–90, 2019.

[45] A. S. Cowen, X. Fang, D. Sauter, and D. Keltner, "What music makes us feel: At least 13 dimensions organize subjective experiences associated with music across different cultures," *Proceedings of the National Academy of Sciences*, vol. 117, no. 4, pp. 1924–1934, 2020.

[46] A. Cowen, P. Laukka, H. Elfenbein, R. Liu, and D. Keltner, "The primacy of categories in the recognition of 12 emotions in speech prosody across two cultures," *Nature Human Behaviour*, vol. 3, p. 1, 04 2019.

[47] A. Cowen, H. Elfenbein, P. Laukka, and D. Keltner, "Mapping 24 emotions conveyed by brief human vocalization," *American Psychologist*, 12 2018.

[48] "1.12. multiclass and multioutput algorithms — scikit-learn 1.3.0 documentation," https://scikit-learn.org/stable/modules/multiclass.html, (accessed: 26.07.2023).

[49] M. Hearst, S. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their Applications*, vol. 13, no. 4, pp. 18–28, 1998.

[50] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[51] E. U. H. Qazi, A. Almorjan, and T. Zia, "A one-dimensional convolutional neural network (1d-cnn) based deep learning system for network intrusion detection," *Applied Sciences*, vol. 12, no. 16, 2022.

[52] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[53] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *CoRR*, vol. abs/1406.1078, 2014. [Online]. Available: http://arxiv.org/abs/1406.1078

[54] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent neural networks for time series forecasting: Current status and future directions," 09 2019.

[55] "neural networks - what exactly are keys, queries, and values in attention mechanisms? - cross validated," https://stats.stackexchange.com/questions/421935/what-exactly-are-keys-queries-and-values-in-attention-mechanisms, (accessed: 09.08.2023).

[56] B. Liu, *Sentiment Analysis and Opinion Mining*, ser. Synthesis Lectures on Human Language Technologies. Springer International Publishing. [Online]. Available: https://link.springer.com/10.1007/978-3-031-02145-9

[57] T. Joachims, "Making large-scale svm learning practical," in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT press, 1999.

[58] J. Shawe-Taylor and N. Cristianini, *Support Vector Machines*. Cambridge University Press, 2000.

[59] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? sentiment classification using machine learning techniques," in *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, 2002.

[60] R. K. Mishra, S. Urolagin, and A. A. Jothi J, "A sentiment analysis-based hotel recommendation using tf-idf approach," in *2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, 2019, pp. 811–815.

[61] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," 2018.

[62] F. Feng, Y. Yang, D. Cer, N. Arivazhagan, and W. Wang, "Language-agnostic bert sentence embedding," 2022.

[63] "Imdb benchmark (sentiment analysis) | papers with code," https://paperswithcode.com/sota/sentiment-analysis-on-imdb, (accessed: 25.07.2023).

[64] X. Wan, "Using bilingual knowledge and ensemble techniques for unsupervised chinese sentiment analysis," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, 2008, pp. 553–561.

[65] K. L. Tan, C. P. Lee, and K. M. Lim, "A Survey of Sentiment Analysis: Approaches, Datasets, and Future Research," *Applied Sciences*, vol. 13, no. 7, 2023.

[66] Y. G. Jung, K. T. Kim, B. Lee, and H. Y. Youn, "Enhanced naive bayes classifier for real-time sentiment analysis with sparkr," in *2016 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2016, pp. 141–146.

[67] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," *CS224N project report, Stanford*, vol. 1, no. 12, p. 2009, 2009.

[68] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[69] "Twitter us airline sentiment | kaggle," https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment?select=Tweets.csv, (accessed: 10.08.2023).

[70] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[71] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1188–1196.

[72] G. M. Raza, Z. S. Butt, S. Latif, and A. Wahid, "Sentiment analysis on covid tweets: an experimental analysis on the impact of count vectorizer and tf-idf on sentiment predictions using deep learning models," in *2021 International Conference on Digital Futures and Transformative Technologies (ICoDT2)*. IEEE, 2021, pp. 1–6.

[73] M. AL-Smadi, M. M. Hammad, S. A. Al-Zboon, S. AL-Tawalbeh, and E. Cambria, "Gated recurrent unit with multilingual universal sentence encoder for arabic aspect-based sentiment analysis," *Knowledge-Based Systems*, vol. 261, p. 107540, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705121008029

[74] M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, M. AL-Smadi, M. Al-Ayyoub, Y. Zhao, B. Qin, O. De Clercq, V. Hoste, M. Apidianaki, X. Tannier, N. Loukachevitch, E. Kotelnikov, N. Bel, S. M. Jiménez-Zafra, and G. Eryiğit, "SemEval-2016 task 5: Aspect based sentiment analysis," in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. San Diego, California: Association for Computational Linguistics, Jun. 2016, pp. 19–30. [Online]. Available: https://aclanthology.org/S16-1002

[75] M. H. Shakeel and A. Karim, "Adapting deep learning for sentiment classification of code-switched informal short text," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, ser. SAC '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 903–906. [Online]. Available: https://doi.org/10.1145/3341105.3374091

[76] A. Rogers, A. Romanov, A. Rumshisky, S. Volkova, M. Gronas, and A. Gribov, "Rusentiment: An enriched sentiment analysis dataset for social media in russian," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, p. 755–763.

[77] O. Y. Koltsova, S. Alexeeva, and S. Kolcov, "An opinion word lexicon and a training dataset for russian sentiment analysis of social media," *Компьютерная лингвистика и интеллектуальные технологии*, vol. 15, p. 277, 2016.

[78] A. Rogers, A. Romanov, A. Rumshisky, S. Volkova, M. Gronas, and A. Gribov, "Rusentiment: An enriched sentiment analysis dataset for social media in russian," in *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, E. M. Bender, L. Derczynski, and P. Isabelle, Eds. Association for Computational Linguistics, 2018, pp. 755–763. [Online]. Available: https://aclanthology.org/C18-1064/

[79] K. R. Scherer and H. G. Wallbott, "Evidence for universality and cultural variation of differential emotion response patterning." *Journal of personality and social psychology*, vol. 66, no. 2, p. 310, 1994.

[80] V. Suresh and D. C. Ong, "Not all negatives are equal: Label-aware contrastive loss for fine-grained text classification," *CoRR*, vol. abs/2109.05427, 2021. [Online]. Available: https://arxiv.org/abs/2109.05427

[81] K. Clark, M. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: pre-training text encoders as discriminators rather than generators," *CoRR*, vol. abs/2003.10555, 2020. [Online]. Available: https://arxiv.org/abs/2003.10555

[82] H. Rashkin, E. M. Smith, M. Li, and Y.-L. Boureau, "Towards empathetic open-domain conversation models: a new benchmark and dataset," in *ACL*, 2019.

[83] S. Mohammad and F. Bravo-Marquez, "WASSA-2017 shared task on emotion intensity," in *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 34–49. [Online]. Available: https://aclanthology.org/W17-5205

[84] ——, "Emotion intensities in tweets," in *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 65–77. [Online]. Available: https://aclanthology.org/S17-1007

[85] A. S. Cowen and D. Keltner, “Self-report captures 27 distinct categories of emotion bridged by continuous gradients,” *Proceedings of the national academy of sciences*, vol. 114, no. 38, pp. E7900–E7909, 2017.

[86] E. Parada-Cabaleiro, G. Costantini, A. Batliner, M. Schmitt, and B. Schuller, “DEMoS: an Italian emotional speech corpus. Elicitation methods, machine learning, and perception,” Feb. 2019. [Online]. Available: https://doi.org/10.1007/s10579-019-09450-y

[87] F. Bianchi, D. Nozza, and D. Hovy, “”FEEL-IT: Emotion and Sentiment Classification for the Italian Language”,” in *Proceedings of the 11th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. Association for Computational Linguistics, 2021.

[88] E. Öhman, M. Pàmies, K. Kajava, and J. Tiedemann, “Xed: A multilingual dataset for sentiment analysis and emotion detection,” *arXiv preprint arXiv:2011.01612*, 2020.

Acknowledgments

I would like to first express my utmost gratitude to the Big Data Management and Analytics Erasmus Mundus consortium for enabling me to have this learning experience. I have grown not only as a researcher and student but also as a person and expanded my horizon culturally and professionally thanks to this opportunity.

My heartfelt gratitude to Giovanni Vedana for supervising my research and following my day-to-day work. Special thanks to Mattia Fosci for believing in me and providing me the opportunity to bring value to Anonymised, while also providing guidance and plenty of insights about the final product. Thanks to Danail Krzhalovski for providing me with more guidance in my project, and to Mustafa Cevik, Sviatoslav Syniak, and Almatkhan Kuanyshkereyev for contributing to the multilingual evaluation in this work.

Furthermore, thanks to Professor Nicolò Navarin for providing valuable time into reviewing and contributing insights in my research.

Finally, and definitely most importantly, a big and warm thank you to my closest family and friends for the emotional support and encouragement. I couldn't have done it without you.