Vrije Universiteit Amsterdam

Avanade

Master Thesis

Dutch sentiment analysis on Twitter

Author: Michelle Rutte (2681730)

1st supervisor: Rob van der Mei
daily supervisor: Bas Vogelzang (Avanade)
2nd reader: Ger Koole

A thesis submitted in fulfillment of the requirements for the Vrije Universiteit Amsterdam Master of Science degree in Business Analytics

August 19, 2022

Preface

The purpose of this thesis is to fulfill the requirements for the Master of Science (MSc) in Business Analytics degree at the Vrije Universiteit in Amsterdam. The Master's in Business Analytics is a two-year program that includes courses to integrate insights from mathematics, computer science, and economics with highly developed communication abilities.

The research conducted in this project was part of an internship at Avanade. Avanade aims to deliver innovative services and solutions to enterprises through the Microsoft ecosystem to their clients [1]. This research focuses on applying sentiment analysis to Dutch tweets related to COVID-19.

I would like to thank Bas Vogelzang and Iman Jabor for allowing me to conduct this research at Avanade and their support along the road. Moreover, I would like to thank Rob van der Mei for his supervision during the project and Ger Koole for being the second reader of my thesis.

Executive summary

Avanade, a leading provider of innovative IT consulting services focused on Microsoft platforms, sees market potential in offering their clients multilingual sentiment analysis. This research is commissioned by Avanade to provide them with the groundwork for Dutch NLP initiatives.

Goal research - The goal of this research is to construct Dutch language models for performing sentiment analysis on COVID-19-related tweets. The Dutch language models BERTje and RobBERT are used for sentiment analysis on Dutch COVID-19-related tweets to answer the research question: "How can Dutch sentiment on the social media platform Twitter related to COVID-19 be detected, and how does the sentiment evolve over time?". Moreover, the performance of these two models is compared to determine which model performs better.

Methodology - The most important aspects of the Methodology used in this research include the labeling experiments, the construction of Dutch language models, and the experiments designed to examine the performance of the models. The first experiment examines the performance of the models on the human-labeled dataset and finds the optimal hyperparameter settings per model. The second experiment investigates the robustness of the models by training the models on a labeled dataset with a lot of bias. Experiment three determines the impact of training models on smaller sets and training sets using sampling techniques for balancing the data. Experiment four includes performing sentiment analysis to examine the overall sentiment in a two-week period, when political decisions are made by the government regarding COVID-19 measures.

Results - The performance of BERTje and RobBERT are tested on a dataset labeled by Avanade colleagues. The first experiment's results indicate that BERTje outperforms RobBERT in classifying the tweets as "Negative", "Neutral", and "Positive", and show that the models' performance improves by using smaller batch sizes. The outcomes of the second experience show that RobBERT outperforms BERTje, which indicates that RobBERT is more robust in predicting the classes if the training set contains a lot of label bias. In experiment three, BERTje outperforms RobBERT when trained on smaller training sets. In addition, RobBERT's performance improves by using sampling techniques. Experiment four shows that RobBERT has more apparent probabilities assigned to the tweets for the three classes. Both models show an overall sentiment change on the days when the Dutch government takes political decisions. Moreover, research of the most negative and positive tweets reveals that these models classify different tweets. Additionally, the most positive tweets classified by RobBERT are more clearly positive than BERTje's.

Conclusion/Recommendation - Responding to the research question, both models can be utilized to detect Dutch sentiment in tweets regarding COVID-19. The sentiment indeed changes over time in the selected period in which the government takes certain measures. Regarding evaluation measures, BERTje is likely the most effective for identifying Dutch COVID-19-related tweets. Referring to the research question, it is advised to also perform this sentiment analysis in a period when more popular measures to reopen society are taken. Avanade can use both the models BERTje and RobBERT constructed in this research to offer Dutch sentiment analysis to customers. However, as obtaining high-quality labeled data is time-consuming, Avanade is advised to conduct further research for learning algorithms, such as Active Learning, to speed up the labeling process.

Contents

1 Introduction **8**
1.1 Context . . . 8
1.1.1 Sentiment Analysis: to get insights from unstructured data . . . 8
1.1.2 Twitter is the preferred social media platform for this project . . . 9
1.2 Problem statement . . . 9
1.3 Research goal . . . 10
1.4 Research question . . . 11
1.5 Organization . . . 12
1.6 Thesis outline . . . 12

2 Literature study **13**
2.1 Natural Language Processing . . . 13
2.1.1 Brief history of NLP . . . 13
2.1.2 Traditional approaches of NLP . . . 14
2.1.3 NLP themes . . . 15
2.1.4 Application of NLP . . . 16
2.2 Sentiment analysis . . . 17
2.2.1 Definition of sentiment . . . 17
2.2.2 Research fields of sentiment analysis . . . 18
2.2.3 Aims of sentiment analysis . . . 19
2.2.4 Importance of sentiment analysis . . . 21
2.2.5 Sentiment analysis on tweets . . . 21
2.3 Pre-trained language models . . . 22
2.3.1 Traditional language model BERT . . . 22
2.3.2 RoBERTa . . . 24
2.3.3 English language model BERTweet . . . 24
2.3.4 Dutch language model BERTje . . . 25
2.3.5 Dutch language model RobBERT . . . 27
2.3.6 Dutch language model RobBERTje . . . 29
2.3.7 Training of language models . . . 32
2.4 Importance of labeled data . . . 33
2.4.1 Active learning . . . 33

3 Methodology **34**
3.1 Data acquisition . . . 34
3.1.1 Twitter scraping . . . 34
3.1.2 Labeling experiment . . . 34
3.2 Preprocessing . . . 37
3.2.1 Fast.ai . . . 37
3.2.2 Tokenization . . . 37
3.2.3 Cleaning the data . . . 40
3.2.4 DataBunch . . . 41
3.3 Dutch language models . . . 42
3.3.1 Experiments . . . 42

3.3.2 Evaluation metrics . . . 44

4 Exploratory data analysis **46**
4.1 Features . . . 46
4.2 Data analysis . . . 46
4.2.1 Human-labeled dataset (baseline) . . . 47
4.2.2 Training set labeled based on polarity extraction for experiment 2 . . . 51
4.2.3 Overview of the training sets for experiments 3 . . . 52
4.2.4 Dataset for experiment 4 . . . 54

5 Experimental results **55**
5.1 Experiment 1 . . . 55
5.1.1 Validation . . . 55
5.1.2 Test . . . 57
5.2 Experiment 2 . . . 58
5.3 Experiment 3 . . . 60
5.3.1 Various sizes of the training set . . . 60
5.3.2 Sampling tweets in training set . . . 60
5.4 Overview of the best results experiment 1-3 . . . 61
5.5 Experiment 4 . . . 62
5.5.1 Characteristics of the dataset . . . 62
5.5.2 The number of classified tweets over time . . . 63
5.5.3 Most negative and positive tweets . . . 65

6 Conclusion **68**

7 Discussion **70**

References **71**

Appendix **97**

List of Figures

1 The stages of analysis in processing natural language [15]. 15
2 Google trends (`www.google.com/trends`) data demonstrating the comparative popularity of the search terms "sentiment analysis" and "customer feedback."[19] 21
3 Encoder-decoder architecture illustration of BERT [43]. 23
4 Fragment from the Dutch section of OSCAR [56] to highlight how different sequences, saved and displayed as distinct rules, are merged within a document [59] 31
5 Representation of the labeling experiment in Azure Machine Learning. 35
6 Distribution of the labels in the collected dataset. 37
7 Most used hashtags and mentioned accounts. 48
8 Distribution of quantity of words in tweets. 49
9 Word clouds. 50
10 Most common words in tweets by frequency count. 51
11 Distribution of the polarity scores from the tweets. 52
12 Distribution of labels across datasets using sampling techniques. 53
13 Data analysis collected dataset for experiment 4 54
14 Test set results of the best performing models of BERTje and RobBERT. 58
15 Test set results of the best performing models of BERTje and RobBERT. 59
16 Distribution of the probabilities to the corresponding classes per model. 63
17 Number of tweets by day from 16 to 30 January. 63
18 The number of tweets per class by BERTje from January 19 to January 24, 2021 . . 64
19 The number of tweets per class by RobBERT from January 19 to January 24, 2021 . 65
20 Labeling experiment 1: overview of Bas's labels. 77
21 Labeling experiment 1: overview of Nathalie's labels. 78
22 Labeling experiment 1: overview of Michael's labels. 78
23 Labeling experiment 2: overview of Michelle's labels. 79
24 Labeling experiment 2: overview of Rob's labels. 79
25 Labeling experiment 2: overview of Iman's labels. 80
26 Labeling experiment 3: overview of Bas's labels. 80
27 Labeling experiment 3: overview of Michelle's labels. 81
28 Labeling experiment 3: overview of Jasper's labels. 81
29 Learning rate versus the loss per weight decay for BERTje with batch size 4. 82
30 Validating results of BERTje with batch size 4 and momentum 0.95-0.85. 82
31 Validating results of BERTje with batch size 4 and momentum 0.99 0.90. 83
32 Learning rate versus the loss per weight decay for BERTje with batch size 8. 83
33 Validating results of BERTje with batch size 8 and momentum 0.95-0.85. 84
34 Validating results of BERTje with batch size 8 and momentum 0.99-0.90. 84
35 Learning rate versus the loss per weight decay for BERTje with batch size 16. 85
36 Validating results of BERTje with batch size 16 and momentum 0.95-0.85. 85
37 Validating results of BERTje with batch size 16 and momentum 0.99-0.90. 86
38 Learning rate versus the loss per weight decay for BERTje with batch size 32. 86
39 Validating results of BERTje with batch size 32 and momentum 0.95-0.85. 87
40 Validating results of BERTje with batch size 32 and momentum 0.99-0.90. 87
41 Learning rate versus the loss per weight decay for RobBERT with batch size 4. 88
42 Validating results of RobBERT with batch size 4 and momentum 0.95-0.85. 88

43 Validating results of RobBERT with batch size 4 and momentum 0.99-0.90. 89
44 Learning rate versus the loss per weight decay for RobBERT with batch size 8. . . . 89
45 Validating results of RobBERT with batch size 8 and momentum 0.95-0.85. 90
46 Validating results of RobBERT with batch size 8 and momentum 0.99-0.90. 90
47 Learning rate versus the loss per weight decay for RobBERT with batch size 16. . . 91
48 Validating results of RobBERT with batch size 16 and momentum 0.95-0.85. 91
49 Validating results of RobBERT with batch size 16 and momentum 0.99-0.90. 92
50 Learning rate versus the loss per weight decay for RobBERT with batch size 32. . . 92
51 Validating results of RobBERT with batch size 32 and momentum 0.95-0.85. 93
52 Validating results of RobBERT with batch size 32 and momentum 0.99-0.90. 93
53 Test set results of the best performing models of BERTje and RobBERT (trained on 300 samples). 95
54 Test set results of the best performing models of BERTje and RobBERT (trained on 600 samples). 95
55 Test set results of the best performing models of BERTje and RobBERT (downsampling neutrals). 96
56 Test set results of the best performing models of BERTje and RobBERT (upsampling positives). 96
57 Test set results of the best performing models of BERTje and RobBERT (combined sample techniques). 97

List of Tables

1 A comparison between BERT Base and BERT Large. 23
2 BERTje tokenizer on the example sentence. 38
3 RobBERT tokenizer on the example sentence. 40
4 Overview over the three datasets used for training, validating and testing. 42
5 Representation of retrieved features from `snscrape`. 46
6 Overview of the number of hashtags, mentions, and emojis in the human-labeled dataset. 47
7 Overview of the the classes in different sets of the polarity-labeled dataset. 52
8 Overview of the training sets for experiment 2. 53
9 Overview of the number of hashtags, mentions, and emojis in the collected dataset for experiment 4. 54
10 Overview of the results per model on the validation set in experiment 1. BS = batch size, LR = learning rate, WD = weight decay, Moms = Momentum, Acc = accuracy, Weighted F1 = Weighted average F1-score, Macro F1 = Macro average F1-score. . . 57
11 Overview results on the test set in experiment 1. 58
12 Overview results on the test set in experiment 2. 59
13 Overview results on the test set using different sample techniques in experiment 3. . 60
14 Overview results on the test set using different sample techniques in experiment 3. . 61
15 Overview of results on the test set of the human-labeled dataset for experiment 1-3. 61
16 Overview the number of predicted classes per model. 62
17 Most negative tweets of BERTje 66
18 Most positive tweets of BERTje 66

19 Most negative tweets of RobBERT 67
20 Most positive tweets of RobBERT 67
21 Overview of the results per model on the validation set in experiment 1. BS = batch size, LR = learning rate, WD = weight decay, Moms = Momentum, Acc = accuracy, F1 = weighted average F1-score, P = weighted average precision, R = weighted average recall. 94

1 Introduction

In this section the context of the study and the company where the study was conducted are explained. In addition, the research problem, the research goal, and the research questions are discussed. Lastly, an outline of the thesis is given.

1.1 Context

People like to express their opinions on social media platforms, especially during a global pandemic like COVID-19. Every so often, new measures are announced by the government mainly to minimize the number of patients in intensive care units. These measures are often leaked to the press several hours before the press conference to see how political decisions fall in society. Since not everyone agrees on the measurements to counteract the number of infections, a relatively large part of the population likes to express their opinions about these measurements on social media. The implementation of these measures takes away freedom from the people. In addition, not all corona measurements provided the desired results in reducing the number of people infected. As a result, several people doubted corona's regulations. Also, other reasons give people certain opinions about the virus and the measures. Some years before the outbreak of COVID-19, several scientists were already concerned about the spread of fake news and disinformation on various social media platforms [2]. The spreading of fake news also affected society, creating distrust towards the government and the Rijksinstituut voor Volksgezondheid en Milieu (RIVM).

1.1.1 Sentiment Analysis: to get insights from unstructured data

Given the concerns mentioned above, it becomes reasonable to assume that social media sentiment will evolve over time. Sentiment analysis can be used to determine if sentiment changes over time. Sentiment analysis is a natural language processing (NLP) technique that determines whether an opinion is positive, negative, or neutral. The term sentiment analysis includes many parts, such as sentiment analysis, opinion extraction, emotion analysis, and review mining [3]. Essential to sentiment analysis is determining how sentiments are expressed in texts. These expressions indicate positive (favorable) or negative (unfavorable) opinions about a particular topic [4].

The developmental history of sentiment analysis is relatively short compared to linguistics and NLP. Sentiment analysis has become very popular in the last twenty years. In addition to being popular in recent years, sentiment analysis has proven its benefits in getting insights from unstructured data, for example, in the healthcare industry and even by predicting specific stock movements. There is often a lot of information on various websites about healthcare, including personal blogs, social media, and reviews on medical issues. Medical experts can use the sentiment extracted from this data to improve medical products and enhance the quality of care. [5]. Additionally, product and service reviews can benefit from sentiment analysis. For example, sentiment analysis was utilized in the research of Batra et al. [6] to forecast the mood of persons who impact stock prices on Twitter. These analyses can assist in forecasting actual stock movement. Batra et al. [6] used a Super Vector Machine (SVM) to determine each tweet's sentiment score. By combining these sentiment scores with additional market information, Batra et al. [6] created an SVM model to forecast the next day's movement of stocks.

As described above, sentiment analysis can be helpful in multiple domains and industries. The Dutch government abolished all coronavirus measures when the virus mutated into the Omicon variant [7], as fewer people were infected and ended up in the intensive care unit (ICU). However, the risk remains that a new mutation could increase the coronavirus breakout figures. Yet, it is also very likely that scientists might identify new viruses in a few years that could similarly affect the entire society. According to RIVM [8], the coronavirus likely evolved in a Chinese market where many different wild animals coexisted. It was not the first time a virus was created by crowding too many animals together. Even in the Netherlands in 2003, there was an avian influenza outbreak, commonly known as Avian influenza [9], which is a contagious disease that affects poultry [10]. The Netherlands is a country with the highest density of chickens in Europe [11]. The avian flu virus has evolved over time, which makes it potentially hazardous if it affects humans. Bird flu returned to the Netherlands even this year, in 2022, causing the worst outbreak in Dutch history. In addition to the avian flu, the Netherlands experienced a Q-fever epidemic from 2007 to 2009, which had arisen from raising dairy sheep and goats [12]. The probability that these viruses will spread globally is growing. As a result, sentiment analysis is still essential for understanding how individuals feel about potential future actions against pandemics. This research on detecting Dutch sentiments on social media may contribute to future pandemics.

1.1.2 Twitter is the preferred social media platform for this project

Dutch people frequently expressed their opinions on social media in opposition to the actions implemented to combat the coronavirus. Social media consists of platforms that are available worldwide, such as Facebook, Twitter, Instagram, etc. Twitter seems particularly interesting for NLP projects because it is a fast responding medium to the news. On Twitter, people can express their opinions and thoughts in tweets. A tweet is a post containing a maximum of 280 characters. The Twitter term “retweet” is identified by the abbreviation “RT”. A retweet is a re-posting of a retweet. Twitter’s Retweet feature helps people quickly share a tweet with all of their followers. Retweeting is sharing another person’s tweet with their own followers. Furthermore, many people use emojis to express their feelings in a tweet. One advantage of Twitter is that users may quickly access content that interests them by following specific pages, such as news programs.

Given the variety of active users on Twitter, it makes sense that a pandemic of the scope of COVID-19 would generate a lot of discussions around the topic. Twitter is one of the most effective ways to ensure that certain news, events, and opinions can go viral. Regarding the COVID-19 pandemic, the public’s trust in the government may be severely impacted if people disseminate false information about COVID-19. On social media, a sizable group of people mistrusting the government can change the tone from positive to negative. Twitter sentiment analysis can monitor opinions on topics such as measures to combat the coronavirus, and one could take timely action before a situation escalates. This project investigates how Dutch sentiment related to COVID-19 can be detected on Twitter.

1.2 Problem statement

Recently, more and more researchers have shown an interest in NLP projects. Most of the research is in English, in contrast to Dutch text data. Particularly interesting for this NLP research are the projects covering sentiment analyses, where various studies cover languages like English, Spanish,

German, and French. However, the Dutch language has less research coverage on the topic of NLP, probably due to its lack in size and the number of people interacting internationally with this language compared to the other languages. Avanade sees an opportunity to offer multilingual sentiment analysis to their customers. Currently, they provide a limited number of NLP projects but are interested in increasing their coverage around the various topics regarding NLP solutions. This research intends to lay the foundation for Dutch NLP projects at Avanade.

Furthermore, this thesis investigates what other studies and academics have accomplished in recent years for extracting sentiment from text data. This investigation covers the available techniques and models for performing sentiment analysis to acquire a deeper understanding of sentiment analysis on Dutch data. The study then examines the findings from earlier research to acquire insight into the implementation of sentiment analysis on Dutch text data. The ultimate objective is to improve the model's performance in classifying sentiment on Dutch tweets by training it on labeled data.

The focus of the project will be on sentiment analysis. The models for sentiment analysis are given a sentence as input and predict a sentiment score based on it. The model categorizes the tweet based on this score. These models need labeled data as input to train the model to be able to classify Dutch sentiment in tweets. There are limited labeled Dutch datasets over the internet available. Data can be labeled by humans or by computers. Computer refers to packages in a programming language (in this project, the focus is on the language Python) that observe the polarity or subjectivity of the given text. The distinction between these two concepts is that polarity extracts sentiment values (positive, neutral, and negative) from a text, whereas subjectivity extracts subjective (based on opinions) and objective (based on facts) values from a text. The problem is that these packages are primarily trained on English text data rather than Dutch text data. The packages will be able to extract some polarity and subjectivity, but not as correctly as they do for English text data. Therefore, a bias is present in a labeled dataset created utilizing packages. Unfortunately, most of the already Dutch labeled datasets available on the internet are labeled using these packages. Avanade is interested in the effect of labeling quality on such an NLP model.

This study employed two labeled data methodologies intending to analyze the impact of quality. At Avanade, a team of volunteers produced a high-quality labeled dataset. As previously stated, the other dataset is labeled using a Python package. Each dataset's results are compared collectively utilizing a variety of models and parameter selections. These datasets are used to train the models. The impact of the accurate labels on the models is then assessed by comparing the outcomes. Hopefully, a clear association between label quality and the model's performance may be observed. In terms of the impact on the quality of a dataset, in the future, Avanade wants to know if it is interesting enough to spend time labeling data and how the time spent on labeling can be made more efficient.

Along with considering how the label accuracy affects the models, training the models on smaller training sets and dealing with unbalanced data are also addressed. Finally, sentiment analysis will be performed over a specified period using the best models from this research.

1.3 Research goal

This research aims to provide a sufficient model for classifying Dutch text on sentiment and identify the change of sentiment over time. In order to reach the goal this project investigates which Dutch

language model is most suitable for NLP projects, which labeling technique works best for the model's performance, the influence of smaller training sets, and the imbalance of labels on the model's performance. Lastly, the best model from the research will be used to construct a Dutch language model for performing sentiment analysis on Covid-19 related tweets.

The project includes recommendations on:

- The models that best classify Dutch sentiment on COVID-19-related tweets.
- The most suitable labeling technique for Dutch NLP projects.
- The influence of the training set size on the model's performance.
- The influence of balancing the training data on the model's performance.
- Performing sentiment analysis over a specific period during the COVID-19 pandemic.

The expected deliverable is to write a well-explained report about the construction of the Dutch language model for performing sentiment analysis on COVID-19-related tweets, the performance of the Dutch language models, and the recommendations.

1.4 Research question

As described in the previous subsections, this project aims to perform sentiment analysis on Dutch tweets related to COVID-19. In addition, the role of creating high-quality labeled datasets and improving the model's performance will be highlighted throughout the research. Therefore, the study aims to answer the following research question: "How can Dutch sentiment on the social media platform Twitter related to COVID-19 be detected, and how does the sentiment evolve over time?". The sub-questions for substantiating the research question are:

1. *What Dutch NLP-based sentiment analysis model is most suitable for Twitter sentiment analysis?*
2. *What are the performance differences between models trained on texts labeled by humans versus those labeled by packages?*
3. *How does training the model on smaller data sets affect its performance?*
4. *What effect do sample methods for data balancing have on the performance of the model?*
5. *How can the change in general sentiment on Twitter related to the pandemic be traced, and do certain events, such as press conferences and demonstrations against the measures, affect general sentiment?*

1.5 Organization

This research is conducted as a collaboration between Vrije Universiteit of Amsterdam and Avanade. Avanade is an international IT consultancy joint venture between Accenture and Microsoft that focuses on consulting services and software development for large enterprises that work with Microsoft technology. Avanade helps companies with their digital transformation. The organization provides an unrivaled combination of insight, innovation, technological expertise, tools, methodologies, and practices. Avanade provides digital innovations in various areas, including infrastructure, security, data analytics, and Enterprise Resource Planning (ERP). The company operates in more than 80 locations in 26 countries. Since 2000, Avanade has had an average year-over-year growth rate of 20%. Currently, Avanade has 12000 current customers, with a total of 4000 customers from 2000 [1].

Avanade has twelve Talent Communities (TC), which can be seen as departments. This thesis project was conducted at the Analytics Talent Community. The Analytics TC has four sub-TCs, including Data Engineering, Advanced Analytics, Analytics Experience, and Analytics Architecture. This project is part of the sub-TC Advanced Analytics. This sub-TC within Avanade strives to create insights from data using data science for their clients.

1.6 Thesis outline

The sections are separated as follows to answer the research question. First, the literature research on related topic is discussed in Section 3. Secondly, the used methodology is discussed in Section 4. Following this, Section 4 explains which data was collected and the data analyses. Subsequently, Section 5 describes the experimental results. After the results, the conclusions are presented in Section 6 with the concluding remarks for answering the research question. Followed by Section 7 addressing the discussions, limitations, recommendations and future work.

2 Literature study

In this section, literature regarding sentiment analysis is discussed. Essential to building a model for classifying Dutch sentiment is to know how previous studies have proceeded with this. This section describes the basics of natural language processing, studies about sentiment analysis, the available pre-trained models, and the importance of labeled data.

2.1 Natural Language Processing

Natural language processing (NLP) refers to the branch of computer science, and is regarded as a domain within artificial intelligence (AI). NLP is a computer-based approach to text analysis based on theories and technologies. NLP uses a set of theoretically motivated computational techniques to analyze and represent natural texts at one or more linguistic analysis levels to achieve human-like language processing for various tasks or applications. Nowadays, NLP continues to be a very active research and development tool [13].

2.1.1 Brief history of NLP

According to the study of Nadkarini et al. [14] NLP started in the 1950s at the intersection of artificial intelligence and linguistics. NLP was originally different from text information retrieval (IR), which used highly scalable, statistically based techniques to efficiently index and search large amounts of text. Over time, the two techniques have pretty much converged. One of the first simpler options was a word-for-word machine translation from Russian to English which initially took a longer until the translation was accurate.

Later, in 1956, Chomosky created a theoretical analysis language grammar that evaluate how difficult a problem is. This analysis influenced the Backus-Naur Form (BNF) notation. The BNF notation is used to identify a context-free grammar and is used to represent the syntax of programming languages. The BNF specification of a language is a collection of derivation rules that together validate the syntactic correctness of program code. In this context, “rules” refers to unbending restrictions rather than the heuristics used by expert systems. Additionally, Chomosky identified even more restrictive “regular” grammars that provide as the foundation for regular expressions used to define text-search patterns. [14].

In 1970, lexer (lexical analyzer) generators and parser generators, such as the lex/yacc pair, started using grammars. A lexer converts text into tokens and the parser validates a set of tokens. Lexer/parser generators produce code and lookup tables that make lexing/parsing decisions by accepting regular-expression and BNF specifications as inputs, respectively. This considerably simplifies the implementation of computer languages. [14].

Natural language’s huge breadth, unrestrictive nature, and ambiguity have led to two problems when utilizing traditional parsing systems that depended solely on symbolic, hand-crafted rules. The goal of NLP is to ultimately extract semantics from text. Syntax is primarily addressed by formal grammars that define the connection between text components, including words like nouns, verbs, and adjectives. By considerably extending sub-categorization and adding new rules and constraints, grammars can be expanded to accommodate natural-language semantics (e.g, the

noun "eat" solely refers to items to be ingested). A set of words can have multiple interpretations. Therefore, the issue is that the number of rules might grow to an unmanageable size and frequently interact in unanticipated ways with more frequent ambiguous parses. The second issue is that handwritten rules handle "ungrammatical" spoken prose and (in medical contexts) the extremely telegraphic prose of in-hospital progress notes rather badly, despite being human-comprehensible [14].

Subsequently, the 1980s led to a crucial realignment, which includes the following elements:

- Simple, robust approaches replaced deep analysis.
- Evaluation became more stricter.
- Probability-based machine learning techniques became common.
- Large, annotated text corpora (which contains the right answers) were used to train machine learning algorithms and served as the accepted standard for evaluation.

This realignment led to the creation of statistical NLP. For instance, by using probabilistic individual rules with associated probabilities that are established by machine learning on annotated corpora, statistical parsing solves the proliferation of parsing rules. As a result, fewer, broader rules replaced a large number of specific rules, and statistical-frequency information is used to resolve ambiguities. The most common situations are used in statistical techniques to learn, which helps them produce good outcomes in practice. The more numerous and representative the data, the better statistical approaches become. Additionally, they succumb to strange or incorrect input more graciously. [14].

2.1.2 Traditional approaches of NLP

Natural language processing research has historically tended to divide language analysis into several stages, matching the theoretical linguistic divisions made between *syntax*, *semantica*, and *pragmatica*. According to a straightforward interpretation, a text's sentences are first examined in terms of their syntax, creating order and structure, which is better suited for exploring its semantics or literal meaning. The meaning of the phrase or text is then ascertained through a stage of pragmatic analysis. This final stage is frequently considered as the one that relates to *discourse*, while the first two typically have to do with sentencing cases. It is well known that in practice it is not that simple to neatly segregate the processing of language into boxes matching to each of the layers. This connection attempt between a stratification distinction (syntax, semantics, and pragmatics) and a separation in granularity (sense versus discourse) can often cause confusion in thinking about the challenges involved in natural language processing.

Nevertheless, the tripartite split into syntax, semantics, and pragmatics only works well as a starting point when processing actual natural language documents is considered. A finer-grained dissection of the process is useful when the current state of the art in combination with the need to deal with real language data is considered, as shown in Figure 1. Tokenization and sentence segmentation are the two critical initial phases in this diagram.

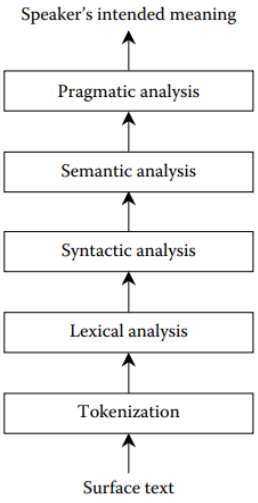


Figure 1: The stages of analysis in processing natural language [15].

2.1.3 NLP themes

It is extremely challenging to create software that reliably ascertains the intended meaning of text or voice data since human language is rife with ambiguity. Metaphors, sarcasm, exceptions to the rules of grammar and usage, are just a few examples of the irregularities in human language, that programmers must teach the software to recognize and understand accurately from the beginning. To help the computer understand the text and speech data it is absorbing, several NLP tasks break down human text and voice data [16]. Explanation on some of these tasks:

- **Speech recognition**, the process of accurately turning audio data into text is known as speech-to-text. Any program that responds to voice commands or questions should use speech recognition. The way individuals speak quickly, slurring words together, with varied emphasis and intonation, in various dialects, and frequently using improper grammar makes speech recognition particularly difficult.
- **Part of speech tagging**, the procedure of identifying the part of speech of a certain word or passage of text based on its use and context is known as grammatical tagging. In the sentences “I can create a paper plane” and “What make of car do you own?,” the word “make” is classified as a verb and a noun, respectively.
- **Word sense disambiguation** is the practice of using semantic analysis to choose the meaning of a word having numerous meanings and discover which interpretation fits the situation the most naturally. Word sense disambiguation, for instance, clarifies the difference between the meanings of the verbs “make” and “make the grade” (achieve) and “make a bet” (place).
- **Named entity recognition**, Using NER, words or phrases are recognized as useful entities. NER identifies “Utrecht” as a place or “Piet” as the name of a guy.

- **Co-reference resolution** is the process of identifying whether and when two words refer to the same thing. The most typical example is figuring out who or what a certain pronoun refers to (e.g., "she" = "Tess"), but it can also require figuring out a metaphor or idiom that is used in the text (e.g., when the Dutch word "beer" refers to a big, hairy person rather than an animal).
- **Sentiment analysis** attempts to extract subjective characteristics from text, such as attitudes, emotions, sarcasm, confusion, and suspicion.
- N**atural language generation** is the process of converting structured information into human language; it's frequently referred to as the opposite of voice recognition or speech-to-text [16].

2.1.4 Application of NLP

Machine Translations, Text Categorization, Information Extraction, Summarization, and virtual agents are examples where NLP is applied [17].

Machine Translations
Making data accessible and available to everyone is a challenge because the majority of the globe is now online. Language is a significant barrier to data accessibility. There are numerous languages, each with unique syntax and sentence structure. Machine translation typically uses a statistical engine like Google Translate to translate words from one language to another. The difficulty with machine translation technology is not in translating words directly but rather in maintaining sentence meaning, grammar, and tenses. To determine the likelihood that anything in Language A correlates to something in Language B, statistical machine learning gathers as much data as it can that appears to be parallel between the two languages. In September 2016, Google launched a new machine translation system based on Deep learning and artificial neural networks. Many techniques have been put out recently to evaluate the quality of machine translations automatically by contrasting hypothesis translations with reference translations [17].

Text Categorization
Systems for categorizing data input a vast volume of data, such as official papers, military casualty reports, market data, news wires, etc., and categorize or index them according to predetermined standards. Email spam filters are another text categorization application. As the first line of defense against unsolicited emails, spam filters are becoming more and more crucial [17].

Information Extraction
Finding interesting terms in text data is the goal of information extraction. Extracting entities like names, places, events, dates, timings, and prices are a powerful technique to summarize the information important to a user's needs for various applications. The automatic recognition of important data in the case of a domain-specific search engine can improve the precision and effectiveness of a directed search. Hidden Markov models (HMMs) are often used to identify the pertinent areas of research publications. These text extracts are utilized to enable searches across certain fields, to effectively present search results, and to link citations to papers [17].

Summarization
In the current digital age, information overload is a genuine problem, and our comprehension of it

has already been surpassed by the reach and accessibility of knowledge and information. Given that this tendency is continuing, it is essential to be able to summarize the data while maintaining its meaning. Text summarization processes massive amounts of digitized text using NLP algorithms to produce summaries and synopses [17].

Virtual agents
Virtual assistants such as Siri (Apple) and Alexa (Amazon) use NLP approaches to give customers the appropriate answer or action. [16].

2.2 Sentiment analysis

In response to an increasing number of informal, opinion-forming texts, such as blog posts and product review sites, a field of sentiment analysis has emerged over the past decade to find out what people think about a particular topic. Researchers from computer science, computational linguistics, data mining, psychology, and sociology collaborate on sentiment analysis by extending traditional fact-based text analysis to enable opinion-based information systems [18].

Nearly as old as spoken communication itself is the interest in other people's opinions [19]. People frequently seek others' perspectives when making daily decisions. For example, people consult political discussion forums when they cast a political vote or searching for reviews before buying a product. Additionally, most people who want to go out for dinner first investigate the best restaurants in a city on various blogs. A new field in text analysis was created using the direct availability of these opinion pieces. Text analysis has expanded its focus from a typically fact- and information-centric view of text to enable sentiment-aware applications as a result of the wide availability of opinionated text. The opinion of internet users about specific products and services is becoming increasingly important to businesses. As a result, sentiment extraction from text has gained more interest in the industry, and academia [18].

Sentiment analysis is one of the fastest developing research topics in computer science, making it hard to keep up with all the activities in the field. Sentiment analysis is well-known for a set of methods, techniques, and tools for detecting and extracting subjective information from languages, such as opinions and perspectives [20]. In recent years, sentiment analysis has expanded from examining online product reviews to social media texts from social media platforms such as Facebook, and Twitter. Sentiment analysis has traditionally focused on opinion polarity, or whether a person has a positive, neutral, or negative opinion on something. In most cases, sentiment analysis is used for product reviews or services whose reviews have been made public on the Internet [19]. In evaluating sentiment extraction/identification, such data is commonly used as a gold standard. The best-known sentiment analysis task is to identify opinions about a particular product across any web content. Today, there are many companies that use sentiment analysis techniques to offer services in the area of brand recognition and market perception [18].

2.2.1 Definition of sentiment

To further explore sentiment analysis and its use, the definition of sentiment will need to be explained first. Determining the objects of study opinions and subjectivity remains one of the challenges of sentiment analysis. Originally, linguists, notably Randolph Quirk, defined subjectivity. Quirk coined

the phrase "private state" as a new definition. A private state is anything that cannot be observed or verified objectively. These internal states encompass a variety of things, such as feelings, beliefs, and speculations. A private state predicts difficulties in analyzing sentiment. Conversations typically entail subjectivity, which is very context-sensitive and particular to each person. In contrast, not every objective sentence is true either [18].

Pang et al. [21] provide the meanings of words closely related to the idea of sentiment to highlight the concept's ambiguity:

- **Opinion** involves a conclusion that has been conceived, but can be challenged
  "each expert seemed to have a different opinion"
- **View** expresses a subjective opinion
  "very assertive in expressing his opinion "
- **Belief** often involves conscious acceptance and intellectual assent
  "a firm beliefs in her party's platform"
- **Conviction** refers to a firmly and seriously held belief.
  "the conviction that animal life is as sacred as human life".
- **Persuasion** implies a belief based on certainty (as by proof) about its truth
  "was of the persuasion that everything changes"
- **Sentiment** implies a fixed opinion that reflects one's feelings
  "her feminist feelings are well known"

Additionally, sentiment has some unique characteristics that can help distinguish it from other qualities. It is typical to desire to categorize material by topic, which may involve dealing with complicated subject taxonomies [18]. Contrarily, the categorization of sentiment typically involves two categories (positive and negative), a range of polarity (such as the star ratings for movies), or even a range in the intensity of the opinion [21]. These classes cover a wide range of subjects, users, and document types. It may appear that managing a small number of classes will be simpler than performing a conventional text analysis. However, that is so far from the truth. [18].

2.2.2 Research fields of sentiment analysis

Computational linguistics, natural language processing, and text mining are closely associated with sentiment analysis as a research area. This field, derived from the study of affective states (psychology) and judgment (evaluation theory), seeks to answer long-studied questions in other domains of discourse by using new tools provided through data mining and computational linguistics. Sentiment analysis is commonly called subjectivity analysis, opinion mining, and valuation extraction, with some connections to affective computing (computer recognition and expression of emotion). This field often studies subjective factors, defined by Wiebe et al. [22] as "linguistic expressions of private states in context," which are mostly single words, phrases, or sentences. There are two types of sentiment in text: "explicit" and "implicit." In explicit, the subjective sentence may directly express an opinion such as "The weather is so beautiful today." In contrast, implicit is where the text indicates an opinion, such as "My phone broke down after only a week." Most research has

focused on explicit sentiment since it is easier to analyze [18].

In addition to the two types of sentiment, sentiment polarity is a unique text feature. Polarity is often divided into positive and negative, but polarity can also be seen as a range. The overall polarity of a document with various opinionated statements would be mixed, which is different from having no polarity at all (being objective). Moreover, a distinction must be established between the polarity of sentiment and its strength. One person is adamant that a product is excellent, another feels it is neither bad nor excellent, and a third has not owned the item for very long may not yet have formed an opinion [18].

Apart from the above components, the purpose of sentiment is also an essential component. The sentiment target can be an object, a concept, a person, or something else. As mentioned earlier, most research is based on product and film reviews, where it is relatively easy to identify the subject in the text. [18].

In contrast to traditional topical analysis, the sentiment statement of the author can be quite important. One of the most crucial complications is quotations. It is essential to comprehend that the sentiment conveyed in the document accurately reflects the author's true intentions. For example, political commentaries and news reports often contain many quotations and opinion citations. Additionally, the quotes may have a complex structure that is very difficult to discern. For instance, a news story about a political debate could include quotes from the participants, commentary from experts, and even the author's stance on the issues [18].

2.2.3 Aims of sentiment analysis

Sentiment analysis involves several separate tasks due to the complexity of the problem, such as underlying concepts and expressions in the text. The complexities are combined to produce some knowledge about the opinions encountered in the text. To better understand the purpose of sentiment analysis, the following paragraphs will provide an overview of the tasks that sentiment analysis encompasses and the tools for each.

The first known task is **sentiment** or **opinion detection**, which is a form of text classification. Opinion detection leans on examining adjectives in sentences, including the polarity a sentence contains. An example would be "What a beautiful picture this is", where the adjective "beautiful" is. The polarity in this sentence can be easily determined by looking at the adjective. A previous study by Hatzivassiloglou et al. [23] mainly investigated the effects of adjectives in sentence objectivity [23]. In addition, the study of Benamara et al. [24] proved that adverbs can be used for a similar purpose [18].

The second known task is **polarity classification**. This classification aims to classify the opinion as one of two contrasting sentiment polarities or determine its position on the continuum between these two polarities [21]. Polarity classification is the binary classification task for labeling an opinion document as generally positive or negative. This classification works by obtaining a text's polarity scores and classifying them based on those scores. Polarity classification is most commonly performed in product review studies, where the definitions of "positive" and "negative" are clearly identifiable, making classification very successful. In contrast, with polarity classification, it is difficult to recognize whether the news is "good" or just "bad". A news item may include "bad"

news without using subjective terms. These classes often appear interchangeably when a document exhibits positive and negative sentiments. For this reason, the most important task is determining the sentiment of the document [18].

To distinguish between different mixtures of the two opposites, polarity classification uses a multi-point scale (such as the number of stars for a movie review). The task of distinguishing involves the multi-class text categorization problem. In contrast to topic-based multi-class classification problems, where the vocabularies for each class are different (or overlap a little), the vocabularies of the positive, neutral, and negative classes can certainly be very similar and differ in just a few keywords. When a document has a mixed opinion, the corresponding class is basically a combination of positive and negative. In text analysis, negations are often seen as unimportant in the sentence, but negations play a crucial role in sentiment. The concept of negations is important because negations can turn an original positive term into a negative one, and vice versa [18].

The two tasks described above can be performed at different levels, namely at the term, phrase, sentence, or document level. The output of such a level is known to be used as input to the higher layers. One example is that sentiment analysis can be applied to sentences and then use the information to evaluate these sentences. For different levels, different techniques are appropriate for usage. Several techniques use n-gram classifiers or lexicons and usually work at the term level. On the other hand, Part-of-Speech (POS) tagging, is correctly utilized for sentence and phrase analysis. In POS, the first action is to assign tags to each token in the text with a corresponding POS tag, such as noun, verb, preposition, etc., based on its definition and context [25]. Different heuristics are frequently employed to generalize sentiment to the document level [18].

Finally, a third task additional to sentiment detection is the discovery of the target of the opinion. The domain of analysis very much affects the difficulty of this task. As stated earlier, it is safe to assume that product discussions are usually about the specified product. According to Liu's research [26], the features are defined as components or attributes of an object, which is a definition most commonly used in practice. The breakdown of the discussion into characteristics allows for a more accurate analysis of sentiment, as well as a more detailed summary of the findings.

Is some cases, a sentimental sentence includes more than one target, which is typically the case in comparative sentences. A subjective comparative sentence arranges objects in order of their preference. An example is "these flowers are much prettier than my garden". Comparative adjectives and adverbs (more, less, better, longer), superlative adjectives (most, least, best), and such other words as "same", "different", "win", "prefer", etc. [26], can be successfully used to identify these sentences. Finally, when the sentences are collected, the objects can be put in an order that is most representative of their merits, as outlined in the text [18].

One of the peculiarities of sentiment is that, while the idea of positive and negative opinion is a general one, the ways in which these opinions are expressed vary greatly throughout the spectrum of topical fields. For this reason, topic-specific and cross-topic sentiment analysis is studied to enhance performance in a particular area. In these two analyses, combining general knowledge about the expression of sentiment and topic-specific expertise is a crucial point. The idea of cross-topic analysis is to use the knowledge gathered about one domain in another domain, for example, from using a labeled dataset to a dataset that is not labeled [27] [28] [18].

2.2.4 Importance of sentiment analysis

The survey of Mäntylä et al. [19] from 2018, describes what a considerable increase they saw in the number of articles focused on sentiment analysis and opinion analysis over the last few years. At the time, there were almost 7000 articles published on this topic, and even more striking and interesting, 99% of the articles appeared after 2004. This shows how fast sentiment analysis has grown in research fields. In the Google search engine, they tracked the increase in searches with the search term "sentiment analysis". Figure 2 shows the increase in the relative popularity of the search term "sentiment analysis" compared to the search term "customer feedback." [19]

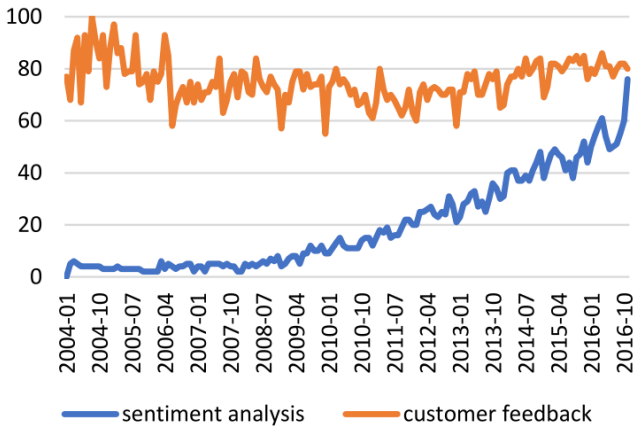


Figure 2: Google trends (`www.google.com/trends`) data demonstrating the comparative popularity of the search terms "sentiment analysis" and "customer feedback."[19]

In addition, the study of Mäntylä et al. [19] discovers that the first academic studies measuring public opinion were opinions during and after World War II, and their motivation is very politic oriented. Modern sentiment analysis really broke out in the mid-2000s. At the time, the focus was mainly on the product reviews available on the Web [29]. However, sentiment analysis has expanded into numerous other areas, such as the prediction of financial markets [30] and the reactions to terrorist attacks [31]. In addition, the overlapping studies of sentiment analysis and natural language processing tackled many problems that contribute to sentiment analysis's applicability, such as irony detection [32] and multilingual support [33]. Additionally, for emotions, attempts to simply detect polarity transitioned to more complex nuances of emotions and distinguishing negative emotions such as anger and sadness [34] [19].

2.2.5 Sentiment analysis on tweets

Sentiment analysis is widely used for product and service reviews, as well as on tweets. This type of analysis of tweets can help guide marketing campaigns to share consumers' opinions about brands and products [35], outbreaks of harassment [36], events that create uncertainty [37], and acceptance or rejection of politicians [38], all in an electronic word-of-mouth manner such as tweets. Sentiment analysis aims to find and identify opinions, emotions, and attitudes in source materials such as

documents, short texts, and sentences. In fields like this, one often encounters large text corpora and "formal language." Nevertheless, considering sentiment analysis on tweets, the concerns are various, mainly due to:

- The number of spelling errors and jargon in tweets is considerably higher than in other domains since Twitter users use a specific cultural vocabulary and many different electronic devices, such as cell phones and tablets.
- Twitter users post on a variety of subjects, as opposed to other sites that are focused on specific topics.

Analyzing the sentiment of tweets is considered a classification problem, and as with essential documents, the sentiment of tweets can be expressed in several ways [39]. Tweets can be specifically classified according to the existence of sentiment. When a message contains sentiment, one can consider such a message as polar (categorized as positive or negative) and otherwise as neutral [40].

2.3 Pre-trained language models

In this section, first the traditional language models. Following that, the research of a language model for tweets is studied to comprehend the procedures involved in creating a model based on tweets. Subsequently, several papers discussing methods for Dutch language models are explained in chronological sequence. Lastly, an approach of a training procedure and the hyperparameters for language models will be covered. These models will be used to decide how to construct a model for detecting Dutch sentiment on Twitter.

2.3.1 Traditional language model BERT

One of the most popular transformer-based large-scaled pre-trained language models is BERT (Bidirectional Encoder Representation from Transformers), which was created by Devlin et al. [41]. BERT has helped improve state-of-the-art performance on many NLP tasks. BERT is a neural network-based model intended to pre-train deep bidirectional representations from an unlabeled text by simultaneously conditioning on both left and right context across all layers. Additionally, the model has a clear conceptual foundation and strong empirical support.

According to Devlin et al. [41], the primary shortcoming of typical language models is that they are unidirectional. BERT avoids this unidirectionality constraint by utilizing a "masked language model" (MLM) pre-training goal. The masked language model masks some of the input tokens at random, intending to predict the original vocabulary id of the masked word based solely on its context. In contrast to the left-to-right language model pre-training, the MLM aim permits the representation to combine the left and the right context, which allows us to pre-train a deep bidirectional Transformer. In addition to the masked language model, a "next sentence prediction" task is employed by Devlin et al. to jointly pre-train text-pair representations.

The BERT framework consists of two steps: pre-training and fine-tuning. The model is pre-trained using unlabeled data across a variety of pre-training tasks. In addition, pre-trained BERT was created by Devlin et al. [41] in a way that the model can be fine-tuned with just one additional output layer allowing for state-of-the-art models to be built without making significant task-specific

architecture changes for a variety of tasks, including question answering and language inference. The pre-trained parameters are employed to fine-tune the BERT model, and labeled data from downstream tasks is used to fine-tune each parameter. Despite being initialized with the same pre-trained parameters, each downstream task has its own fine-tuned models [41].

The Transformer model of Vaswani et al.[42] served as the foundation for BERT's model design, which is built on a multi-layer bidirectional Transformer encoder. The Transformer completely relies on attention mechanisms, with no repetition or convolutions. The architecture of the Transformer consists of two parts: the encoder and the decoder. The implementation of BERT uses only the encoder. The encoder consists of a stack of six identical layers, with two sub-layers in each layer. An example of encoder-decoder architecture is shown in Figure 3. In this example the sentence is translated from English to German.

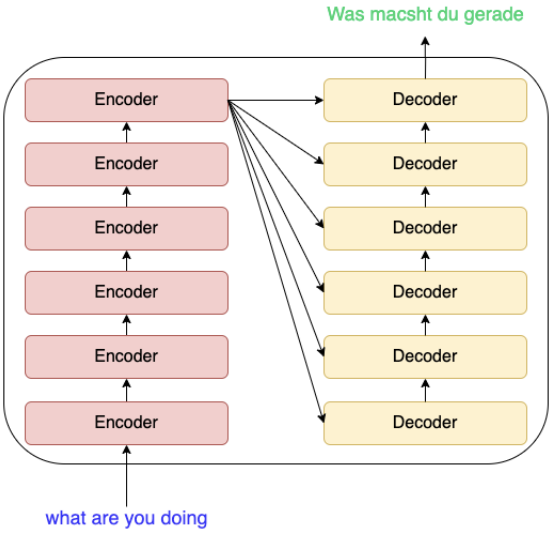


Figure 3: Encoder-decoder architecture illustration of BERT [43].

As explained above, BERT is simply a stack of encoders. BERT comes in two different model sizes: $\text{BERT}_{\text{base}}$ and $\text{BERT}_{\text{large}}$. The differences in sizes are feasible in Table 1.

Table 1: A comparison between BERT Base and BERT Large.

	$\textbf{BERT}_{\textbf{base}}$	$\textbf{BERT}_{\textbf{large}}$
Layers	12	24
Hidden size	768	1024
Heads	12	16
Parameters	110M	340M

The BERT language model and other versions of BERT have successfully contributed to new state-of-the-art performance results for several NLP tasks, including sentiment analysis. The success of the models is primarily tied to regular English domains such as Wikipedia, news, and books [44].

2.3.2 RoBERTa

In 2021, Liu et al. [45] were convinced that BERT was significantly undertrained and propose an improved recipe for training BERT models, named RoBERTa. RoBERTa stands for a robustly optimized BERT pretraining approach, which is a replication study of BERT. The architecture includes 12 self-attributing layers with 12 heads [41] with 117M trainable parameters.

RoBERTa can match or exceed the performance of all the post-BERT methods. The modifications of Liu et al. [45] are simple, it includes:

1. training the model longer, with bigger batches, over more data;
2. removing the next sentence prediction objective;
3. training on longer sequences;
4. dynamically changing the masking pattern applied to the training data.

The main difference is that the masking procedure during the model training was done statically at BERT and dynamically at RoBERTa. The initial BERT implementation only used a static mask because masking was done only once during data preprocessing. The training data was copied ten times, resulting in each sequence being masked ten different ways throughout the duration of the 40 training epochs. This was done to prevent utilizing the same mask for every training instance in every epoch. As a result, each training session was observed four times while wearing the same mask. In this study, the static masking method was compared with the dynamic masking method. Liu et al. [45] generate the masking pattern whenever a sequence is fed to the model. This becomes more important when pretraining across a larger dataset or with more stages.

The research of RoBERTa has led to the conclusion that training the model longer, with larger batches over more data, removing the next sentence prediction objective, training on longer sequences, and dynamically changing the masking pattern applied to the training data can all significantly improve performance.

2.3.3 English language model BERTweet

For this research, it is critical to understand how prior studies dealt with Twitter data in order to determine the approach of constructing a Dutch language model on COVID-19-related tweets. The preprocessing steps that were carried out for BERTweet will be explained in the following paragraphs, along with the data that was used.

Since 2020, BERTweet has been the first public largescale pre-trained language model for English Tweets, which was created by Nguyen et al.[44]. The architecture of BERTweet and $\text{BERT}_{\text{base}}$ is identical, and the model was trained using RoBERTa. In the study of constructing BERTweet, Nguyen et al. described that Twitter is one of the most prominent microblogging platforms where

people share real-time information about all their experienced topics and events. In addition, it is clarified that the huge and abundant tweet data has shown to be a popular and real-time source of information in various important analytical tasks. Applying existing language models trained on traditional texts to tweets remains a challenge due to the typical short length of tweets and frequent use of informal grammar and irregular vocabulary. BERTweet is trained on an 80GB corpus of 850 million English tweets. The pre-training data also included COVID-19-related tweets.

The data is composed of tweets of at least ten and up to 64-word tokens. Nguyen et al. [44] used the language identification component of the Python library `fastText` to identify only the English tweets [46]. This research [44] proves that the performance of the BERTweet model improved when the tweets were better pre-processed. The first pre-processing step was using a specific TweetTokenizer from the NLTK toolkit [47]. The second pre-processing stage also included transforming the emojis in the tweets into strings and normalizing them by changing user mentions and web/URL links into the distinctive tokens @USER and HTTP URL, respectively. Nguyen et al. [44] then segmented all 850M Tweets with subword units using a vocabulary of 64K subword types. Resulted in tweets containing an average of 25 subword tokens [48] [44].

2.3.4 Dutch language model BERTje

Pretrained language models are often trained on English texts. However, the research of de Vries et al. [49] is an example of how the BERT architecture can be used on a different language, in this case Dutch. The Dutch language model BERTje uses the same architecture and parameters as the pre-trained language model BERT. The paper of de Vries et al. [49] compared BERTje with the multilingual BERT model (M-BERT). The multilingual BERT model is based on all Wikipedia pages of 104 languages, including Dutch, while BERTje is based on a large and diverse dataset of 2.4 billion Dutch tokens.

For the pre-training data and parameters of BERTje, de Vries et al. [49] chose to train a single Dutch BERT-based model that is architecturally equivalent to the $\text{BERT}_{\text{base}}$ model with 12 transformer blocks [41] to facilitate comparison. Nevertheless, the pre-training data were clearly different, and further adjustments were made to the pre-training data generation based on later derivations of the BERT. In addition, de Vries et al. [49] strived to collect a data set of comparable size and diversity as used for the initial English BERT model. For pre-training, they combined the following several corpora of high-quality Dutch text [49]:

- Books: a collection of contemporary and historical fiction novels (4.4GB)
- TwNC (Ordelman et al., 2007): a Multifaceted Dutch News Corpus (2.4GB)
- SoNaR-500 (Oostdijk et al., 2013): a multi-genre reference corpus (2.2GB)
- Web news: all articles of 4 Dutch news websites from January 1, 2015 to October 1, 2019 (1.6GB)
- Wikipedia: the October 2019 dump (1.5GB)

De Vries et al. [49] constructed a similar WordPiece vocabulary with a vocabulary size of 30K tokens as in BERT. The learning approach for a WordPiece tokenizer is as follows. First, it pre-tokenizes the text into words (by dividing it based on punctuation and whitespace), and then, it tokenizes each word into individual wordpieces. WordPiece is a greedy longest-match-first approach for single-word tokenization, repeatedly choosing the longest prefix of the remaining text that fits a vocabulary token [50].

Subsequently, De Vries et al. [49] decided to construct this vocabulary using a SentencePiece model on the raw pre-trainng dataset. SentencePiece is a straightforward and language-independent text tokenizer and detokenizer designed primarily for Neural Network-based text production systems where the vocabulary size is defined before the training of the Neural Network [51]. The advantage of this technique is that this tokenizer does not presume to use spaces to separate words in the entered text. The use of spaces to separate words, however, varies by language. Therefore, using language-specific pre-tokenizers such as SentencePiece is a possible approach. The space is included in the list of characters to use since this method handles the input as a raw input stream. The right vocabulary is then created using the Byte Pair Encoding (BPE) or unigram technique [48] [52]. SentencePiece makes it possible to construct a fully end-to-end system that is independent of any language-specific processing [51]. To comply with the original BERT concept, the generated vocabulary is translated to WordPiece format after being created using a SentencePiece.

According to the use of the BPE and unigram technique in SentencePiece, BPE is a straightforward data compression method that expands its vocabulary by replacing the most often occurring pair of bytes in a sequence with a single, unused byte iteratively. The BPE algorithm was first presented as a text compression method, but it is also ideally suited as a tokenizer for language models. BPE divides words into a series of two-word units that frequently appear in a reference corpus, which is the corpus used to train the model. The algorithm then goes over each word individually, counting the number of occurrences of token pairs. After collecting all of the frequencies for each pair, the ones that appear the most frequently will be chosen. As a result, a vocabulary can also consist of words with just one or two letters [52] [53].

Unlike BPE or WordPiece, Unigram starts with a superset of the final vocabulary and pruning each symbol to create a smaller vocabulary. For example, the base vocabulary may include all pre-tokenized words as well as the most prevalent substrings. Although Unigram is used in conjunction with SentencePiece, it is not directly used for any of the models in the transformers. The Unigram algorithm, using the current vocabulary and a unigram language model, defines a loss (typically described as the log-likelihood) over the training data at each step. The method then computes how much the aggregate loss would increase if each symbol in the vocabulary were eliminated. The symbols with the smallest loss increment (i.e. those that have the least impact on the overall loss across the training data), are then removed by Unigram to the extent of p (where p is often 10 or 20 percent). This procedure continues until the vocabulary has expanded to the size desired. The base characters are always maintained by the Unigram method, allowing any word to be tokenized. The primary distinction between the Unigram method with the BPE and WordPiece algorithms is that Unigram is not based on merging criteria. Unigram saves the likelihood of each token in the training corpus in addition to the vocabulary, allowing the probability of each conceivable tokenization to be calculated after training. In practice, the algorithm just chooses the most likely tokenization, but it

also provides the option to sample several tokenizations based on their probabilities [52] [53].

After constructing the vocabulary, BERTje was pre-trained for 1 million iterations. De Vries et al. [49] wanted to measure the influence of the number of iterations on the performance of the downstream tasks. They also evaluated the fine-tuning performance at the 850k iterations checkpoint. The model has been refined for different NLP cases to assess the effectiveness of BERTje for use on downstream tasks. As a train set, they used 80% of the resulting documents. The remaining data consists of 10% of the validation set and 10% for the test set. The described downstream tasks, such as NER, POS-tagging, and SRL, represent low-level linguistic information. Therefore, to test BERTje on a higher level task, BERTje is also applied for sentiment analysis. For this analysis, de Vries et al. [49] used the 110K Dutch Book Reviews Dataset [54]. This dataset consists of a balanced collection of positive and negative reviews deployed for a binary sentiment classification task [49].

Three models are fine-tuned for each of the previously described tasks: the multilingual BERT base, BERTje at the 850K checkpoint (BERTje850k), and the fully trained BERTje model (1M checkpoint). De Vries et al. [49] trained these three models for four epochs on the train set for each task with the same hyperparameters. The results indicated that the model's performance on the validation data deteriorated due to the too-long training. In addition, no increase in performance was seen after the fourth epoch [49].

The outcomes of the sentiment analysis are only highlighted. The results of de Vries et al. [49] show that with the task sentiment analysis on the 110K Dutch Book Reviews Dataset, BERTje has an accuracy of 93.8% without hyperparameter tuning. De Vries et al. [49] concluded that they have successfully created, refined, and evaluated the Dutch language BERT-based model. BERTje consistently outperforms the multilingual BERT model on word-level NLP tasks. Compared to the multilingual BERT model, which incidentally also performs well on Dutch NLP tasks, the results show that de Vries et al. [49] would prefer a monolingual model [49].

2.3.5 Dutch language model RobBERT

As described earlier, multilingual BERT performs well on many NLP tasks. However, several recent studies have shown that a monolingual BERT model is preferred. Therefore, training a Dutch BERT model has great potential for a wide range of Dutch NLP tasks. In addition to BERTje, Pieter Delobelle et al. [55] created a Dutch language model with the use RoBERTa and called the model RobBERT. Therefore, the architecture of RobBERT is identical to the RoBERTa's base model.

For the pre-training phase, two different versions of RobBERT were trained. For RobBERT version 1 (RobBERT v1), Pieter Delobelle et al. [55] chose to replace the pre-training corpus with a Dutch corpus, and they replaced both the corpus and the tokenizer with Dutch versions for RobBERT version 2 (RobBERT v2). Using these two versions can enable the importance of having a language-specific tokenizer. Pieter Delobelle et al. [55] use the OSCAR corpus's Dutch section to pre-train the model. This large multilingual corpus was obtained by language classification in the Common Crawl corpus [56]. The Dutch corpus is 39 GB in size, containing 6.6 billion words distributed over 126 million lines of text, where each line can have multiple sentences. This corpus is considerably larger than the corpora employed for comparable Dutch BERT models. For example,

BERTje is trained on a 12GB corpus, and BERT-NL on a SoNaR-500 corpus (about 2.2GB) [55].

RobBERT v2 utilizes a Dutch tokenizer instead of a standard byte pair encoding (BPE) tokenizer from RoBERTa. Developing the vocabulary of the tokenizer was done using the Dutch part of the OSCAR corpus [56] with the same byte-level BPE algorithm as RoBERTa [45]. As explained in the previous section on BERTje, the tokenizer gradually constructs a vocabulary by replacing the most frequent consecutive tokens with a new merged token. The vocabulary in the tokenizer is limited to 40K words. This tokenizer of RobBERT v2 contains 10K fewer words than the RobBERT v1 due to additional tokens, including a non-negligible number of Unicode tokens that are not used in English. Misclassified sentences probably cause this non-negligible number of Unicodes during the creation of the OSCAR corpus [56] [55].

The RobBERT model has been evaluated on multiple downstream Dutch NLP tasks. To test the task text classification, researchers assessed the model on sentiment analysis and predicting demonstrative and relative pronouns. Predicting demonstrative and relative pronouns is beneficial for evaluating zero-shot prediction capabilities (i.e., only the pre-trained model without fine-tuning). To find out how well the RobBERT performs on smaller data sets, Pieter Delobelle et al. [55] used subsets of the data for both classification tasks.

For this project, the focus is only on the evaluation of the sentiment analysis part described in the paper by Pieter Delobelle et al. [55]. The study compares the high-level sentiment analysis with BERT-NL [57], and BERTje [49]. For the seniment analysis task, the Dutch Book Reviews dataset (DBRD) was used. The dataset contains reviews that are labeled as positive and negative. This dataset consists of 118,516 reviews, of which only 22,252 of these reviews were labeled. Of this dataset, 90% was used for the train set and 10% for the test set.

In the research of Van der Burgh et al. [54], the DBRD was released, and the researchers described the performance of a ULMFiT (Universal Language Model Fine-tuning for Text Classification) model on this dataset. Van der Burgh et al.[54] fine-tuned the ULMFiT model with the unlabeled reviews before training the classifier. For Pieter Delobelle et al. [55] of the RobBERT model, it was unclear whether the other BERT models used the unlabeled reviews for further pre-training [58] or whether they only used the labeled data to fine-tune the pre-trained model. Pieter Delobelle et al. [55] chose to use only the labeled data for fine-tuning, which would allow for better future performance for the RobBERT model if one did additional pre-training on unlabeled in-domain sequences. Additionally, it was unclear how other researchers had developed models that could handle reviews longer than the allowed number of tokens. The book reviews typically have a length of 547 tokens, with 40% of the documents exceeding the model's maximum length [55]. Pieter Delobelle et al. [55] decided to just utilize the final tokens of a review as input because most reviews include a summary of the entire review. In comparison to choosing the first quantity of tokens, RobBERT's training performance was also improved.

The results showed that RobBERT performed better than BERT-NL and BERTje. Additionally, the two versions of RobBERT perform better than the state-of-the-art ULMFiT model. However, the difference is only statistically significant for RobBERT v2 [55]. In future studies, RobBERT can be used in new settings. For example, RobBERT could be employed in a model that relies on a BERT-like transformer stack for the encoder and a generative model as the decoder. In addition,

RobBERT can serve as the basis for multiple Dutch downstream NLP tasks. Since RobBERT's state-of-the-art performance is adequate on both small and large datasets, it may be helpful to refine the model on new datasets to improve its performance.

To conclude, Pieter Delobelle et al. [55] arrived at a new language model for Dutch-based on RobBERTa, called RobBERT. The results showed that RobBERT outperforms previous approaches and other BERT-based language models for several different Dutch language tasks. Most notable is that RobBERT performed significantly better than other BERT-like (an umbrella term for BERT and similar optimized BERT models [59]) models on smaller data sets, meaning that RobBERT is a valuable resource for a wide range of application areas. Pieter Delobelle et al. [55] expect that RobBERT can serve as a basis for fine-tuning other tasks. They also assume that RobBERT can contribute to developing new models to improve results for Dutch NLP tasks [55].

2.3.6 Dutch language model RobBERTje

Delobelle et al. [59], the same researchers who created the RobBERT model [55], have been developing different methods to distill language models into smaller models to increase efficiency with a slight trade-off in achievements. In the paper of Delobelle et al. [59], they created several distilled versions of the state-of-the-art Dutch RobBERT, called RobBERTje. The smaller models differ in their distillation corpus, meaning whether the corpus is shuffled or not and whether the data is merged with subsequent sentences. Delobelle et al. [59] believe that the performance of the models using shuffled versus non-shuffled data sets is comparable on most tasks, and randomly combining consecutive sentences in a corpus leads models to train faster and perform better on tasks with long sequences.

Continuing the success of distilling the knowledge from neural network models, many types of distillation have been used to extract optimal parameters or the ability of larger language models into smaller models. Distilling the models requires fewer resources and time to run, even though the models are less accurate. The distillation process can provide a beneficial trade-off between performance and ease of use in deployment. In this study, Delobelle et al. [59] perform different distillations using a small unlabeled Dutch dataset and fine-tune them to various language tasks to find the optimal processing of the dataset and target architecture hyperparametrizations. The paper focuses on evaluating data processing for distillation, replicating studies of distillation architectures, and constructing more lightweight versions of RobBERT to provide more efficient fine-tuning and power-efficient inferencing of Dutch downstream language tasks [59].

Distillation is known for a technique that allows a more straightforward model called "the student" to learn from a more complex model called "the teacher." This technique was first called model compression, which viewed a large ensemble model as "a teacher" to label a large unlabeled dataset from which the student model can learn. The advantage of this technique is that the student gets access to a larger dataset, which leads to somewhat noisy labels due to errors in the teacher model. Compared to the teacher model, the student model can be used in many more situations, as the model is smaller and therefore faster and requires fewer resources, but at the cost of lower accuracy. Subsequently, the model compression technique extended to neural networks in knowledge distillation. This distillation takes advantage of the fact that the neural network generally predicts the probability for each possible label by generating the class probabilities using the softmax output layer. Originally, researchers devised this method to compress ensemble models into simple neural

networks, even though the technique is used for various similar distillations. Recently, distillation was also used, for example, to distill neural networks into similar networks with fewer layers and neurons or with more efficient basic operators. In addition, it is intended as a technique for discovering suitable learner architectures [59].

Distilling language models have received much attention in recent years, as expanding large-scale pre-trained language models with lots of parameters scales exponentially. The distilled techniques are reasonably easy to implement on the BERT-like models and less time-consuming to train and perform inference. The main idea behind distillation techniques is to make a model much smaller without sacrificing too much accuracy for a particular NLP task. The training of BERT-like models is often divided into pre-training and small specific fine-tuning. The distillation technique can be used after both training phases [59].

In the study on RobBERTje [59], Delobelle et al. pointed out that few previous studies have been interested in evaluating the transfer dataset. However, the study of Wouts et al. [60] describes that there has been some disagreement as to whether training on the non-shuffled or shuffled versions of the OSCAR training dataset affects the performance of a pre-trained BERT-like model positively or negatively. Delobelle et al. [59] conducted several distillation experiments by distilling several smaller models from the Dutch RobBERT model. These researchers decided that the distilled RobBERT models should also use smaller versions of the RoBERTa architecture [45] and the OSCAR corpus [56] for its architecture and transfer dataset. Delobelle et al. [59] set up experiments for examining the influence of order and the length of the transfer dataset and replicated studies of the DistilBERT and Bort architectures. According to the study by Wynter et al.[61], the Bort model has an optimal hyperparameterization, where inference rate, parameter size, and error rate are balanced. Compared to $\text{BERT}_{\text{large}}$, which contains 340M parameters as can be seen in 1, the student Bort has only 56M parameters [61] [59].

Three experiments have been conducted to test the distilled models. The first experiment investigates the influence of a shuffled training corpus. The OSCAR corpus is originally a non-shuffled version. Although, in the last years, a shuffled form has also been made available. According to the study by Woutsen et al. [61], using a non-shuffled version, the model would be able to learn dependencies that involve multiple sequences. The sequence itself would also not need to be of interest for pre-training since each input sequence is accessed independently. Some researchers believe that it is possible that not shuffling the dataset could impair training performance due to the lack of diversity in each batch. To delve more into this, Delobelle et al. [59] set up an experiment that distilled two models, one based on a shuffled and one based on a non-shuffled version. The models have help from the DistilBERT regime, in which only the nature of the transfer dataset was different. DistiBERT is a distilled model (student) based on the BERT model (teacher), where the student has the same general architecture as the teacher [62] [59].

The second experiment examines the length of the training sequences. The Dutch OSCAR corpus contains mainly short lines, often under 40 tokens, and does not consist of multiple sentences. However, because the OSCAR corpus grabs the document's beginning and end, these short sequences are created using line breaks and can be merged into a valid longer text sequence. A new transfer dataset was created by combining two consecutive lines from the same document into one training sequence with a probability of $p = 0.5$. A fragment from the Dutch section of OSCAR [56] that

is merged can be seen in Figure 4. Therefore, since the OSCAR corpus contains reasonably short sentences, the following positions do not have as many training examples as earlier positions, which may affect tasks that contain essential information at the end of the input positions. Because of this, Delobelle et al. [59] think that merging successive sequences from the same documents could theoretically enhance the model's performance on downstream tasks, including long lines processing. Merging sequences can make the dataset more compact in fewer sequences, which can be an advantage for training time since it is reduced. However, the disadvantage is that there is relatively less training data left for the initial input positions of the model compared to the original version in which a non-merged and non-shuffled corpus. Finally, Delobelle et al. created a new dataset from the unshuffled Dutch OSCAR dataset by randomly merging a sequence with its next sequence with a 50% probability if they come from the same document. This dataset is much smaller than the original corpus with generally longer sentences [59].

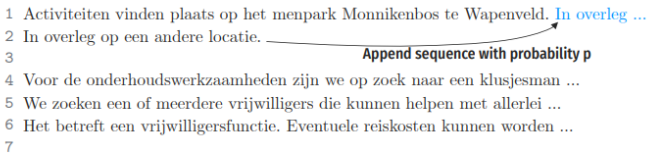


Figure 4: Fragment from the Dutch section of OSCAR [56] to highlight how different sequences, saved and displayed as distinct rules, are merged within a document [59]

The third experiment investigates which distillation architecture hyperparametrization works best for the distilled model. In this experiment, Delobelle et al. [59] replicate the study of Wynter et al. [61] using the same learner architecture with the same hyperparameters and test if these parametrizations are still optimal when used on this Dutch RobBERT model. The merged subsequences transfer dataset was employed to distill this Dutch Bort model from its RobBERT teacher.

Delobbe et al. [59] evaluated the performance of the distilled models on six downstream tasks, of which only the sentiment analysis is covered. The Dutch Book Reviews dataset (DBRD) is used to evaluate the sentiment analysis task. As mentioned earlier, this dataset consists of binary classification (negative or positive reviews). Delobelle et al. [55] confirm that when the model uses the last 512 tokens of a sequence, the performance of RobBERT's evaluation improves.

After fine-tuning the distillation, Delobelle et al. [59] created four distilled model variants for five downstream tasks, including sentiment analysis. For the experimental setup, five models were trained with random hyperparameters for each task, leading to 100 fine-tuned models. Delobbe et al. [59] select the best performing model on the validation set and evaluate it on the test set. From the results, the non-shuffled, the shuffled, the merged, and the Bort distillations, these four distillations achieved an accuracy of around 90% on the sentiment analysis task. The Bort distillation performed slightly less, and the accuracy lies in a confidence interval of 89.3 ± 1.3, while the merged one manages an accuracy between 92.9 ± 1.1. According to the study by Delobbe et al. [55] the accuracy of the RobBERT v2 model lies in the interval 94.4 ± 1.0, which is still the best performance for the task sentiment analysis. In addition to RobBERT also, BERTje also outperformed RobBERTje.

Delobbe et al. [59] concluded that in creating the RobBERTje models, they discovered that the influence of using a shuffled dataset is small for distillation. Moreover, randomly merging successive sequences of the unshuffled dataset improves the performance of the distilled language model for tasks with longer input sentences. In addition, Delobbe et al. [59] found it interesting that all distilled RobBERTje models show less stereotype bias than their teacher RobBERT. From the results, one can consider that the new distilled RobBERTje models can be used to make many NLP tasks more efficient and still achieve almost the state-of-the-art results. Delobbe et al. [59] are convinced this will lower the threshold of fine-tuning Dutch BERT-like models thanks to lower computational requirements and faster inference times. In addition, they count on the Dutch NLP community's benefits in using RobBERTje to fine-tune the model for their downstream task with less computational power and storage.

2.3.7 Training of language models

The models described in the previous sections are neural network-based models. The used hyperparameters correspond to these of a neural network. The most common hyperparameters mentioned in the studies above are batch size, learning rate, number of epochs, and momentum.

The batch size, or the number of data required to train a single forward and backward pass, is one of the most crucial hyperparameters. The network may take too long to reach convergence if this hyperparameter is set too high (no more gain in accuracy). In contrast, if the batch size is too small, the network would fluctuate without obtaining an adequate level of performance. Additionally, the kind of dataset can affect the batch size, particularly the complex medical dataset. [63].

According to the study of Leslie Smith [64], the efficient training procedure for a neural network combines a rising cyclical learning rate (allowing an initial low learning rate to start convergence) and a decreasing cyclical momentum (allowing the learning rate to increase in the early to middle stages of training). The learning rate and the momentum are tightly connected. The optimal learning rate depends on momentum, which in turn depends on the learning rate. The most crucial hyperparameter for these models to tune is the learning rate, and therefore momentum is as important as the learning rate. Similar to learning rates, setting momentum as high as possible while preventing training instabilities is beneficial.

Overfitting may happen if the learning rate is too low. Large learning rates make training more consistent, but if they become excessive, training will diverge. It is, therefore, conceivable to detect learning rates that converge or diverge using a grid search [64].

In addition, weight decay is a crucial hyperparameter for preventing overfitting. Weight decay is a type of regularization that plays a vital part in training, and its value must therefore be properly established [64]. According to the study of Leslie Smith [64], reasonable values for the weight decay to test are 10^{-3}, 10^{-4}, and 10^{-5}.

2.4 Importance of labeled data

In the field of machine learning, data has the greatest importance. In supervised learning, a machine learning model is trained based on sample data labeled according to a particular target concept. As a result, one ultimately wants to transition from a model to a learned function that can predict the unseen labels. The machine learning model's performance depends on the quality of the labeled training data [65]. To keep deep learning-based detection models up to date, huge volumes of new labeled training data are required, which can be costly and time-consuming. In order to successfully train these models, there is frequently relatively little data available regarding new malware or assaults [66].

2.4.1 Active learning

Finding a dataset can be difficult sometimes, but it becomes even more difficult if one wants to utilize a model that needs labeled data. An intriguing technique that predetermines which texts are pertinent for the model to label would be essential to learning models. In the field of machine learning and artificial intelligence, there is a learning algorithm that fulfills this task. Active Learning is also known as "query learning" and occasionally "optimal experimental design" in the statistics literature. The fundamental premise is that if the learning algorithm is permitted to choose the data from which it learns, it will perform better with less training. For any supervised learning system to work successfully, it frequently needs to be trained on hundreds (or even thousands) of labeled examples. This makes it a desired quality for learning algorithms to have. The acquisition of labeled instances in many complicated supervised learning tasks is exceedingly challenging, time-consuming, or expensive [67].

Active learning systems make an effort to get around the labeling bottleneck by posing queries as unlabeled instances that must be classified by an oracle (e.g., a human annotator). This reduces the cost of getting labeled data since the active learner strives to attain high accuracy with the fewest number of labeled instances possible. In many contemporary machine learning issues, where data may be plentiful, but labels may be hard to come by or expensive to acquire, active learning is well-motivated [67].

3 Methodology

The procedures for developing classification models to identify Dutch sentiment will be covered in this section. First, the data acquisition will be discussed. The second subsection describes the pre-processing methods for the data. Finally, the models that were employed will then be covered with the conducted experiments and the evaluation measures that were used to validate the results.

3.1 Data acquisition

The data collection process will be broken down into two parts: tweet scraping and labeling. The primary data consists of tweets that are expressed in words, and the sentiment labels could also be represented in words ("Positive", "Neutral", "Negative") or numbers (0, 1, 2). Section 4: *Exploratory data analysis* will focus on a closer examination of the data.

3.1.1 Twitter scraping

Twitter data can be collected using various methods, such as the Twitter API and other Python tools. In this study, tweets were gathered using the library `snscrape` [68] in Python 3.9.7. `snscrape` is a scraper tool that can provide social service scans. Various social networking services can be retrieved, including Twitter [69]. Snscrape was developed as a result of it being discovered that the free version of the Twitter API had too many restrictions. As a result, snscrape was chosen to be used in this project.

Periodically, tweets from each day in a predetermined date range from December 1, 2020, to February 28, 2021, were retrieved. This time period was chosen because several political decisions concerning COVID-19 regulation were made in the Netherlands. According to the Dutch COVID-19 schedule, the first lockdown was implemented in December 2020. It happened just before the holidays, which was bad timing for many people as they were prepared to buy gifts and eager to spend time with friends and family. The curfew was also introduced in January, as was the start of the vaccinations. The lockdown had been extended by February 2021, and more emphasis was being devoted to the socio-economic effects.

The goal of this project is to compile tweets about COVID-19 and to perform sentiment analysis to COVID-19 related tweets. Finding the ideal keyword for obtaining Dutch tweets required some exploration. The keyword "the coronavirus" was used to gather the majority of Dutch tweets. Additionally, the Python `langdetect` language-detection library was employed to guarantee that only Dutch tweets were collected. This package uses a detection algorithm to determine the language of each tweet. From the dataset, the non-Dutch tweets were taken out. Given that the dataset will only contain unique tweets, the retweets are additionally extracted from the data.

3.1.2 Labeling experiment

To arrive at a model for sentiment analysis on corona related tweets, the goal is to further train the chosen model on COVID-19 related tweets. To further train a classification model, labeled data is required. Hardly any Dutch-labeled datasets were available on the Internet. The accessible datasets included some bias since they had been labeled by packages trained on English literature.

In order to provide a dataset of a reasonable quality that can be utilized to further train the selected model, it was decided to classify a dataset by humans. Azure Machine Learning was used for accessible labeling. This platform enables data scientists and developers to build, deploy and manage high-quality models faster and confidently, as well as edit and manage data [70]. The data was tagged by a total of seven individuals, including six Avanade employees and one external volunteer. The experiment of labeling went as follows. Three participants were given 300 identical tweets to categorize as "Positive", "Neutral", "Negative", and "Don't Know". The "Don't know" class was for the tweets containing too many spelling errors or where the context could not be extracted from the tweet. Figure 5 provides a visual representation of the labeling experiment. The goal was to label purely on sentiment which is a fairly difficult task. The following example was given to the participants to get an idea of how they could best label the data on sentiment:

1. "My iPhone broke this week."
2. "I am devastated that my iPhone broke this week."

In these two tweets, there is a noticeable difference between the sentiment that each statement conveys. People may interpret the first tweet negatively because they can relate to how frustrating it is when a phone breaks. However, as this tweet doesn't genuinely express a strong opinion, it would likely be categorized as neutral. The second tweet does contain a clear negative sentiment through the words "I am devastated".

Figure 5: Representation of the labeling experiment in Azure Machine Learning.

The duration of the experiment varied from experiment to experiment since some tweets were more difficult to identify or contained more content. The experiment lasted an hour or a little longer for most of the participants. The participants had more than a week to identify the given dataset. The labeling might then be divided into daily segments of 10 minutes to reduce participant fatigue. After the experiments, the tweets that were labeled as "Don't Know" were examined to determine if they should be removed from the dataset. The dataset was cleaned up of the don't knows that were obviously vague or ambiguous. Additionally, it was debatable if some tweets with the label "Don't know" were relevant to the model. The ultimate decision on whether they should be excluded from the dataset or not was made with Bas Vogelzang. The tweets listed below are samples of tweets that were excluded from the dataset for a variety of reasons:

- **Too many ambiguities and or too many spelling errors, causing the tweet to lose its valuable information.**

  - *"Reddy de mey geen toelating van zijn Chinese communistische leider zijn pvdachterlijken -clan en bang dat de al die militairen hem als hét corona-virus -syndroom zouden ontmaskeren"*

- **A more common type of tweet. For instance, if multiple users utilize the same statement from a news article or press conference, these tweets may be too similar and should be removed to avoid overfitting.**

  - *"De vanmorgen gevonden nieuwe Nederlandse mutatie van het Corona virus is 50x minder besmettelijk dan de gewone variant. Het blijkt dat de virusdeeltjes eerst eindeloos met elkaar overleggen voordat ze iemand besmetten'. #denhaag #vvd #pvda #cda #d66 #groenlinks #avondklok #sp "*
  - *"Er is een Nederlandse variant van het Corona virus ontdekt. Het is 50x minder besmettelijk dan de gewone variant. Het blijkt dat de virus deeltjes eindeloos overleggen voor ze iemand besmetten."*
  - *"Er blijkt nu ook een NL variant te zijn: 'De vanmorgen gevonden nieuwe Nederlandse mutatie van het Corona virus is 50x minder besmettelijk dan de gewone variant. Het blijkt dat de virusdeeltjes eerst eindeloos met elkaar overleggen voordat ze iemand besmetten'."*
  - *"😂gehoord...'De vanmorgen gevonden nieuwe Nederlandse mutatie van het Corona virus is 50x minder besmettelijk dan de gewone variant. Het blijkt dat de virusdeeltjes eerst eindeloos met elkaar overleggen voordat ze iemand besmetten'. #coronadebat #avondklok"*

- **Even with language recognition, tweets occasionally appeared to be written in a mixture of languages, such as half Dutch and half English, German, African, etc.**

  - *Die departement van basiese onderwys het Vrydagoggend aangekondig die skoolkalender vir 2021 word weens die corona-virus aangepas.✍️: @earlseptember en @luks_byron*

As it can be difficult to identify tweets by sentiment, each experiment was completed by three volunteers to ensure the correctness of the labels. The labels would be more biased if only one individual applied labels to the tweets rather than three. Ultimately, the label with the highest number of clicks was selected. Three labeling experiments in total were carried out. A total of 300 tweets were identified by two groups of three respondents, and 308 tweets were labeled by a third group of three respondents. The results of the first two experiments showed that more tweets than anticipated were labeled as "Don't know". Therefore, the third experiment included 308 tweets to compensate for some of the deleted tweets. Bas and Michelle again participated in the third experiment because not enough Avanade employees volunteered for the labeling experiment. Appendix A contains the person-specific summary of each tagged dataset. The resulting dataset contained 72 positive tweets, 507 neutral tweets, and 299 negative tweets (see Figure 6). In the rest of the report, this dataset is referred to as human-labeled dataset.

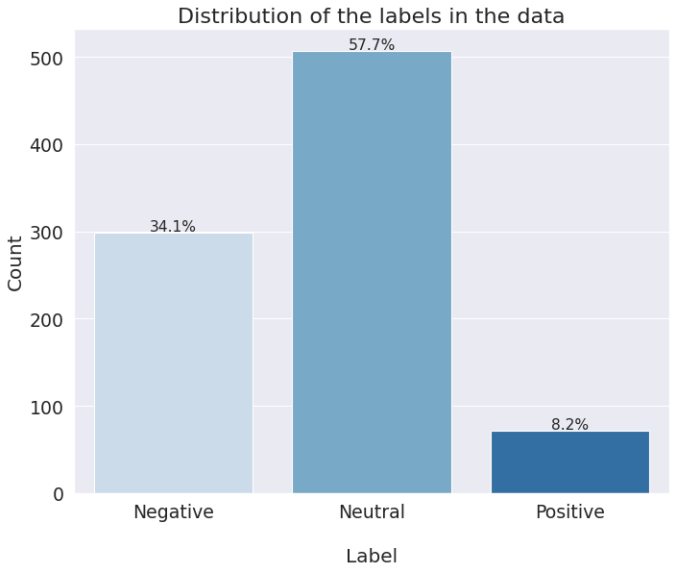


Figure 6: Distribution of the labels in the collected dataset.

3.2 Preprocessing

The preparation tasks will be covered in this section. First, the library `fast.ai` will be described, which was used to prepare the data and create the model. Then, as part of the preprocessing steps, the details about tokenization, data cleaning, and the DataBunch API will be addressed. Finally, the Dutch language models will be described along with the experiments and evaluation measures.

3.2.1 Fast.ai

`fast.ai` was used for the preprocessing steps and building the model. `fast.ai` is a deep learning library that offers researchers high-level components that may be combined to create novel approaches as well as practitioners high-level components that can rapidly and easily provide state-of-the-art outcomes in common deep learning domains. It seeks to accomplish both without making significant concessions in terms of use, adaptability, or performance [71].

3.2.2 Tokenization

In the majority of text processing projects, tokenization comes first. How a token is defined has a significant impact on how well a model performs since a token acts as an atomic unit that embeds text's contextual information [72].

BERTje
As described in the literature study, De Vries et al. [49] built a WordPiece vocabulary with a vocabulary size of 30K words. Moreover, this vocabulary was constructed using a SentencePiece model. Given that this method handles the input as a raw input stream, the spaces between words

are included in the list of characters to utilize. Subsequently, the right vocabulary is created using the Byte Pair Encoding (BPE) or unigram technique.

As explained above and in Section 2.3.4 on BERTje of the literature, BERTje's tokenizer does not divide the sentence based on white spaces, hence the words do not contain any additional symbols to represent a white space. As an illustration of how BERTje's tokenizer works, consider the following sentence (ignore the spelling errors): *"hoe langer ik na het corona virus kijk, hoe meer het op een bitterbal begin te lijken."*. The outcome of the tokenization is shown in Table 2. The tokenizer split the word "corona" into "cor" and "##ona", which indicates that "cor" is the start of the word and "##ona" is the completion of the word. Different tokenizers utilize various special symbols to represent the subwords. BERTje's tokenizer uses "##" to represent the second subword. Additionally, [CLS] is a special symbol added before each input example, and [SEP] is a special separator token [41].

Table 2: BERTje tokenizer on the example sentence.

Vocabulary	Given tokens
[CLS]	1
hoe	13324
langer	14844
ik	13604
na	15885
het	13261
cor	10380
##ona	27457
virus	21693
kijk	14242
,	11
hoe	13324
meer	15473
het	13261
op	16804
een	11130
bitter	9608
##bal	24162
begin	8890
te	19883
lijken	15081
.	13
[SEP]	2

RobBERT v2
As mentioned in the literature study, RobBERT v2 uses a Dutch tokenizer with the same traditional byte pair encoding (BPE) algorithm as the RoBERTa's tokenizer [55]. The BPE algorithm was described in Section 2.3.4 on BERTje of the literature.

The tokenizer of RobBERT v2 has a 40 thousand word vocabulary. The beginning and end tokens for each sentence are {'<s>': 0} and {'</s>': 2}, respectively. For words with a space before them, the vocabulary offers unique tokens. Consider the same example sentence from the dataset as for BERTje's tokenizer: *"hoe langer ik na het corona virus kijk, hoe meer het op een bitterbal begin te lijken."*. The tokens for this sentence are provided in Table 3. The table demonstrates how, for instance, the tokenizer interprets the word "langer", which has a space before it in the sample text, as the word "Ġlanger". The letter Ġ denotes a space before a word. Furthermore, the table shows that when the tokenizer does not have a term in the vocabulary, the word is divided up. For example, "Ġcorona" is not in the vocabulary, but "Ġcor" and "ona" are, therefore the tokenizer generates two distinct tokens for the word corona.

Table 3: RobBERT tokenizer on the example sentence.

Vocabulary	Given tokens
<s>	0
hoe	6490
Ġlanger	946
Ġik	29
Ġna	106
Ġhet	10
Ġcor	8912
ona	5603
Ġvirus	9404
Ġkijk	1413
,	6
Ġhoe	132
Ġmeer	54
Ġhet	10
Ġop	13
Ġeen	9
Ġbitter	14506
bal	3135
Ġbegin	706
Ġte	14
Ġlijken	2602
.	4
</s>	2

3.2.3 Cleaning the data

Before using a transfer model, it is crucial to clean the data properly. In contrast to other text data, Twitter data is generally rather untidy because tweets frequently contain jargon, grammatical errors, and people creating their own words. The cleaning steps for tweets are outlined below:

- Removing hyperlinks (HTTPURL) and mentions (@USER)
- Lower-case all words
- Converting the emojis

It was unclear in several studies what researchers did with emojis ("😀") and emoticons (":)") in tweets when they utilized a BERT-like model. As described in the literature study for the model

BERTweet, Nguyen et al. [44] converted the emojis into strings with the package `emoji` and their model produced positive outcome. Therefore, it was decided to turn the emojis in the tweets into strings as well. Then, the strings were translated to Dutch using the package `deep-translator` since the converted strings were in English. Underscores were present in the transformed strings but have been eliminated. An example is this emoji: 😳, which is converted in the string: "flushed_face", and after translating and removing the underscore, the string resulted in: "blozend gezicht" (Dutch). Furthermore, the emoticons were kept in the dataset.

Due to the context-related nature of BERT-like models, the stop words were left in the dataset. Additionally, all punctuation was preserved in the tweets because RobBERT v2 understands every punctuation mark, whereas BERTje knows practically all of them.

3.2.4 DataBunch

To ensure that the model accurately analyzes the data, the texts should first be preprocessed, and the data can then be fed into the model in an efficient manner. The `fast.ai` DataBunch API provides a straightforward approach to not only load the data but also shuffle and test it without requiring a lot of setting in order for the model to load the data appropriately. The API specifies input and output standards for many sorts of data. One advantage of the DataBunch API is that it accepts a wide range of data types and produces the same kinds of data, making the model highly adaptable in terms of the types of data and formats it can handle. [73].

Splitting the data
As previously mentioned, `fast.ai`'s DataBunch enables users to divide the data into a train set, validation set, and test set. In this project, it was chosen to create the test set beforehand, because this collection of tweets will be established and used subsequently for each experiment as test set, which is explained in more detail in a subsection later. The DataBunch divided the train and validation sets, and the test set was added to the DataBunch. The test data consists 88 tweets, which is around 10% of the entire dataset. This means that the remained data contains of 790 tweets. For creating a validation set of around the same size as the test set, the split for the validation set was set to 11% (which is almost 10% of the whole dataset). A split of 10% of the data for the validation and test set was decided upon due to the short size of the dataset and the model's preference for many training samples.

The different sets should consist fairly the same distribution in labels. Especially, a requirement for all the sets is that these contain a sufficient number of positively labeled tweets, since there are few positive tweets compared to neutral and negative tweets. As noted in the labeling experiment in subsection 3.1.2, there are 72 positive tweets in total, accounting for 8% of the whole data set. Approximately 8% of the tweets in the training set, validation set, and test set should be positively identified in order to get reliable test results. The results of the model's performance on the test set could be slightly better when there are fewer than seven tweets in the test set, which reflects a slightly skewed outcome. An overview of the train, validation and test set is shown in Table 4.

Table 4: Overview over the three datasets used for training, validating and testing.

	Train	**Validation**	**Test**	**Total**
Number of positives	56	9	7	72
Number of neutrals	404	50	53	507
Number of negatives	244	27	28	299
Size	704	86	88	878

3.3 Dutch language models

This section discusses the models provided in this study, the general structure of these models, the experimental setup, the used hyperparameters for each model, and the evaluation metrics. Based on the literature study, two kinds of Dutch language models will be used for sentiment analysis in this research: BERTje, based on BERT's architecture [49], and RobBERT v2, the Dutch version of the robustly optimized RoBERTa [55]. The remainder of the report uses the term RobBERT to refer to RobBERT v2. These two machine learning techniques perform well on Dutch sentiment analysis tasks. The models can classify texts as either positive or negative. This project aims to further train the models on a human-labeled dataset and fine-tune these models also to classify neutral tweets. Both models were created using the `fast.ai` package version 1.0.61. The models only require the tweets and their labels as features. Both models generate probabilities for the three classes, and the highest probability determines a tweet's class.

As described in the literature study, BERTje was the first created model for analyzing Dutch texts. BERTje is a single Dutch BERT-based model with 12 transformer blocks, structurally equivalent to the $\text{BERT}_{\text{base}}$ architecture. The characteristics of $\text{BERT}_{\text{base}}$ are; 12 layers, a hidden size of equal 768, 12-heads, and 110 M trainable parameters [74].

The second model examined for this study is RobBERT. As described in the literature review, the RobBERT model is a Dutch language model for many NLP tasks, including sentiment analysis. In this project, RobBERT's architecture consists of 12 self-attention layers with 12 heads consisting of four layers. The model can be divided into 14 blocks:

- 1 Embedding layer
- 12 Transformer layers
- 1 Classifier layer

3.3.1 Experiments

For this research, four experiments are set up. At each experiment, the outcomes of the models in terms of evaluation metrics on the validation set and test set will be presented to assess the performance of the models. These evaluation metrics will be discussed in greater detail in the following subsection. In addition, the comparison of the loss function on the train and validation set and the confusion matrices for each experiment will be presented and discussed in Section 5:

Experimental results.

For all the following experiments, the hidden dropout probability was set to 0.05, the number of hidden layers was set to 768, and the maximum sequence length was 256. The first three experiments use the test set generated from the human-labeled dataset to compare the outcomes. The different experimental conditions varied on four levels:

1. **BERTje versus RobBERT v2**: The created labeled-human dataset will be used to further train and fine-tune the two models. The models' performance is improved by considering and adjusting additional hyperparameters. In this experiment, the hyperparameters: batch sizes, learning rate, weight decay, and momentum are tuned.
   - **batch size -** The batch sizes $\in \{4, 8, 16, 32\}$ are examined per model. As the batches get bigger, the model's quality could deteriorate, which would ultimately affect how well it generalizes to new data. Therefore, in general, the batch size is one of those hyperparameters that must be checked and changed based on how the model performs during training.
   - **momentum -** According to the study of Smith [64] explained in the literature, setting momentum as high as possible while preventing training instabilities is beneficial. Therefore, it was decided to evaluate each model using the batch size and the cyclic momentum of 0.95 to 0.85 and 0.99 to 0.90.
   - **learning rate and weight decay -** A grid search will be used to find the optimal values for the learning rate and the weight decay. As mentioned earlier, the package `fast.ai` was used for building the model. The package has a function for finding an appropriate learning rate. The function accomplishes this by choosing a very low learning rate initially, training one mini-batch at this learning rate, and calculating the loss. The following mini-batch is trained at a progressively higher learning rate, and this procedure is repeated until a learning rate is attained at which the model clearly diverges [75]. In this project, it was decided to select values for the weight decay that had a few more steps between them than the values mentioned above. Therefore, a modest grid search was performed with the values 10^{-2}, 10^{-4}, and 10^{-6} for weight decays.

   This experiment will evaluate which hyperparameter setting for BERTje and RobBERT performs best on the human-labeled dataset. For the upcoming experiments, these two models are used.

2. **Different labeling techniques**: As mentioned earlier, the data can be labeled using people and, for example, packages. In this experiment, the same dataset obtained for the human-labeled dataset will be labeled using a package named `TextBlob` by extracting the polarity score of the tweets. In the rest of this thesis, this dataset is referred to as a polarity-labeled dataset. The selected models from experiment 1 will be trained and fine-tuned on this dataset. The models are tested on the same test set labeled by humans and used in experiment 1. This experiment's results will indicate how the transfer model will deal with a labeled dataset that is not correctly labeled (attached some bias). Based on this, it will become clear if the quality of labeling makes much difference to the model's performance.

3. **Downsampling versus upsampling**: This experiment is divided into two sub-experiments:

- *Sup-experiment 1:* BERTje and RobBERT with the best hyperparameter setting from experiment 1 are trained on 300 and 600 training samples to investigate the effect of the training size on the performance of the models. The distribution of classes remained the same as in the human-labeled dataset.
- *Sup-experiment 2:* The options for downsampling (decrease the majority class) and upsampling (increase the minority class) have been implemented since the training data showed an imbalance between the classes. The downsampling works in one mode: "Dropout Neutrals". The tweets from the majority class (the neutrals tweets) are dropped out with a probability of around 25.75% (from 404 to 300 tweets). As an alternative to downsampling, upsampling was used to generate more training samples for the positive class since these tweets are in the minority. By resampling tweets from the dataset, the number of tweets with positive labels has increased approximately 78.57% (from 53 to 100 tweets). In the end, the downsampling and upsampling techniques were combined to produce a dataset with a dropout of neutrals with a probability of 25.75% and an increase in positives with a probability of around 78.57%. As a result, the dataset is more balanced across the two classes, positive and negative. The sampling techniques in this experiment will provide information on how the models react to a more balanced dataset.

4. **Sentiment analysis**: In addition to optimizing the model's performance and the quality of the labeled dataset, it is also crucial that the models can provide insight into overall sentiment over time. This experiment aims to determine whether the general sentiment has significantly changed over time due to specific political choices made in the Netherlands about COVID-19. The corona timeline was also crucial in understanding why the mood might suddenly change. From experiments 1 to 3, the best model for BERTje and RobBERT with the optimal hyperparameter values is chosen. For the next period, these two models will be utilized to perform sentiment analysis:

    - **January 16, 2021 to January 30, 2021**
      On January 20, 2021 the government announced that the implementation of the curfew would take place on January 23. On January 23, the government actually instituted the curfew to curb the number of corona infections [76]. For this study it was decided to collect tweets from a week before and after January 23. The dataset was gathered using the same keyword as the human-labeled dataset: "the coronavirus."

3.3.2 Evaluation metrics

The most popular evaluation criteria for sentiment analysis (which is basically a classification problem) are accuracy, precision, recall, and F1-score. The equations below represent the formulas for these measurements. The letters TP, TN, FP, and FN in the equations stand for true positive, true negative, false positive, and false negative, respectively.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

$$\text{F1-score} = \frac{2 \times \text{ Precision } \times \text{ Recall}}{\text{Precision } + \text{ Recall}} \tag{4}$$

The accuracy metric, in general, calculates the proportion of accurate predictions to the total number of occurrences examined. Additionally, precision is utilized to quantify the positive patterns in a positive class that is accurately predicted from the total positive patterns predicted. Furthermore, recall measures the proportion of correctly classified positive patterns. Lastly, the harmonic mean is represented by the F1-score metric between values for recall and precision [77].

For this project, the weighted average F1-score is taken into account since the focus is on a multi-classification problem, and the collected data showed some label imbalance [77]. In addition to the weighted average F1-score, the macro average F1-score and the micro average F1-score were considered. Macro averaging gives each class equal weight, but micro averaging gives each per-document categorization decision identical weight. Large classes predominate over small classes in micro averaging because the F1-score measure overlooks true negatives, and the quantity of true positives mostly determines its magnitude. The macro average F1-score helps to get an idea of its effectiveness in small classes [78]. Therefore, it was decided to take, in addition to the weighted average F1-score, also the macro average F1-score into account. Furthermore, only the weighted average will be provided for precision and recall. Moreover, the confusion matrices of the models on the validation and test sets will also be provided since a confusion matrix is a foundation for each of these metrics.

4 Exploratory data analysis

This section provides the data analyses of the tweets. First, the features of the datasets using the package `snscrape` will be discussed. The second subsection describes the different datasets per experiment and covers some data analyses.

4.1 Features

As mentioned in the methodology, the package `snscrape` was used for collecting the data. `snscrape` retrieves 28 features. Some of these features contain the word "NaN" (Not a Number) for each datapoint and are therefore not considered. The features that consist of information are explained in Table 5.

Table 5: Representation of retrieved features from `snscrape`.

Information	**Explanation/features**
user	information about user: name, number of followers, profile, location, url to profile image, characters of their tweets (e.g. number of likes and attached media), protected page
urls	link of tweet: *url*, link to source: *source* and *sourceUrl*, links attached: *outlinks*, *tcooutlinks* (tco links: shortened links from Twitter), *media*
date	date and time of the tweet
place	*coordinates*, *place*
id	*id* (tweet id), *conversationId*, *inReplyToTweetId*
content	*content*, *renderedContent* (same text as in content)
counts	*replyCount*, *retweetCount*, *likeCount*, *quoteCount*
language	*lang*
source	*sourceLabel* (such as Twitter Web App or Tweetdeck)
mentions	*mentionedUsers*
symbols	used *hashtags* (#) and *cashtags* ($)

For the experiments, the only feature from Table 5 that is given to the model is *content*. In addition, the feature *date* was crucial for experiment 4, where sentiment analysis was performed over a period of time.

4.2 Data analysis

As discussed in Section 3: *Methodology*, the dataset resulted in 878 tweets after the labeling experiments. This dataset will be used in various ways for the experiments. In experiments 1 and 3, this dataset is employed for training, and in experiment 2, the same tweets in the training set are labeled based on polarity extraction and these models are then tested on the human-labeled test set.

The following training and datasets are created for the experiments:

- **Experiment 1**
  - Human-labeled dataset, which can be seen as the baseline dataset.
- **Experiment 2**
  - Training set of tweets labeled based on polarity extraction
- **Experiment 3**
  - Training set of 300 samples
  - Training set of 600 samples
  - Training set: downsampling neutrals
  - Training set: upsampling positives
  - Training set: downsampling neutrals and upsampling positives combined
- **Experiment 4**
  - Dataset of retrieved tweets from January 16, 2021 to January 30, 2021

First, for the human-labeled dataset, some overall data analyses for the unique tweets will be provided. Second, the labeling technique based on the polarity extraction and the analyses of these polarity scores will be discussed. Next, the summaries of the created training sets for experiment 3 and some analyses will be delivered. Lastly, some analyses of the retrieved dataset for experiment 4 will be given.

4.2.1 Human-labeled dataset (baseline)

The methodology already covered the amount of samples per class for this dataset. The following aspects of this dataset are also examined in order to generate insights: the number of frequently used twitter components, the most used hashtags and mentions, the distribution of lengths in tweets, the difference in words across the three classes, and lastly the most common consecutive words in tweets.

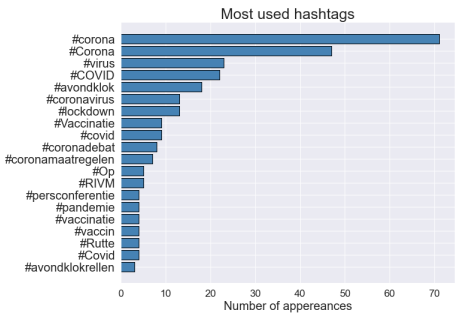
People on Twitter often combine different components to make the message of their tweets more effective. For example, emojis, mentioned accounts, and hashtags all accomplish this. A summary of the numbers of these components are shown in Table 6.

Table 6: Overview of the number of hashtags, mentions, and emojis in the human-labeled dataset.

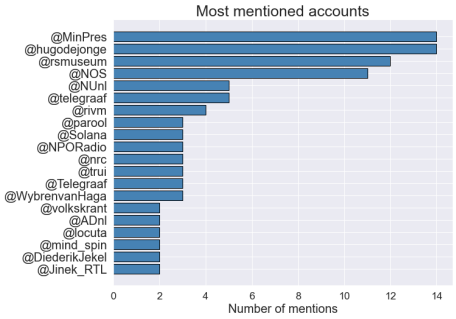
	Human-labeled dataset
Number of hashtags	640
Number of mentions	294
Number of emojis	80

Before eliminating the mentions to clean the data, the most commonly used hashtags and mentions in the dataset were identified (see Figure 7). Most hashtags, as shown in Figure 7b, refer to the coronavirus and governmental measures like the lockdown and curfew. In addition, the most

commonly mentioned account is @MinPres, which stands for minister-president, as shown in Figure 7b on the right. Mark Rutte is the owner of this Twitter account. Moreover, newspapers, news channels, RIVM, and Dutch talk shows are the next most frequently mentioned sources.



(a) Most used hashtags.

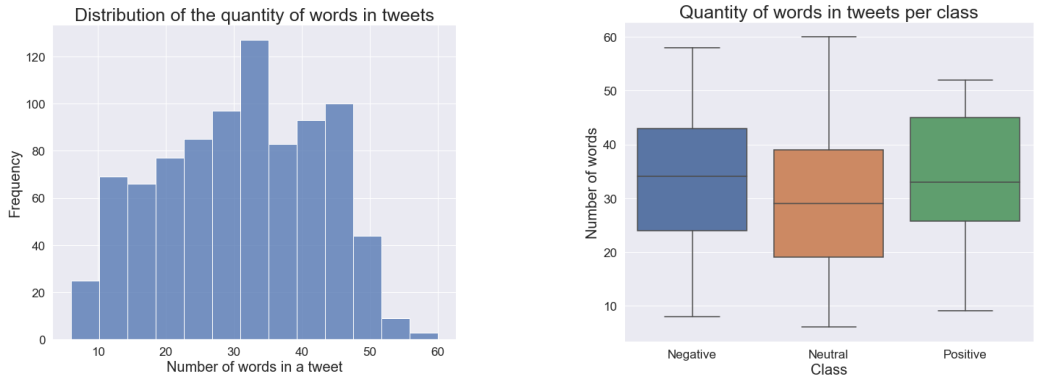


(b) Most mentioned accounts.

Figure 7: Most used hashtags and mentioned accounts.

For diving deeper into the lengths of the content in tweets, these lengths are analyzed. As described in the introduction, a tweet can contain 280 characters. When a tweet contains more characters, the tweet breaks down in the middle, which can be recognized by "(1/2)". Most tweets are complete except for four tweets. These four tweets only contain the first half, but the clue of the tweet still makes sense, and therefore these tweets were kept in the dataset.

In this analysis, the emojis were not converted into strings and counted as one word. Figure 8 shows the distribution in the length of tweets. The majority of tweets, as shown in Figure 8a, are between 30 and 35 words long. The variation in the length of tweets ranges from six to sixty words. In addition, the distribution of the number of words in tweets by class may be observed in Figure 8b. The tweets labeled negatively (median 34) tend to be slightly longer than the tweets labeled as positive (median 33) and neutral (median 29). Moreover, the variation in the length of tweets is fairly even across the labels.



(a) Length of tweets

(b) Length of tweets per class

Figure 8: Distribution of quantity of words in tweets.

Some examples from the dataset are given first to explore the difference in tweets across the classes. The following list of samples will enable a better understanding of the kind of tweets that are classified as positive, neutral, and negative:

- **Positive**:
  - *Ik blijf nog even thuis en hou mij aan de vervelende maar noodzakelijke maatregelen om het Corona-virus in NL te beteugelen. Lijkt mij verstandig. Ik wens iedereen wijsheid en geluk* 🍀
  - *Yes! Krijg vandaag mijn eerste inenting ter bescherming tegen het Corona virus, omdat ik dus in de zorg werk. Moet dan wel even een stukje reizen, maar so be it! #vaccination #Vaccinatie*
- **Neutral**:
  - *Niemand heeft ooit 'het corona virus' gezien.*
  - *Het corona-virus zal niet verdwijnen en daarom moeten naar andere oplossingen zoeken dan lockdowns en avondklokken*
  - *Nederland leeft 1 jaar met het corona virus..*😳
- **Negative**:
  - *De #lockdown heeft allemaal niets met een #corona virus te maken maar met de rechtszaken van clubjes als #Urgenda en #milieudefensie die de hele wereld terug willen sturen naar de middeleeuwen. Het is wachten op de grote #reset21 die het #WEF al heeft aangekondigd.*
  - *Vandaag na alle corona shit weer een k** dag. Mijn moeder (ook gepakt door het corona virus) wederom opgenomen in het ziekenhuis. Weer met verminderde nierfunctie. Kan door corona komen, door hartfalen. Ze drinkt ook te weinig, zo eerlijk ben ik ook. Maar het blijft k.u.t.*

– *Rutte moet niet klagen over de #avondklok zijn hele slappe en verkeerde aanpak van het #Corona virus heeft langer geduurd dan strikt noodzakelijk. Rutte gaf voorkeur aan economie draaiend te houden ipv het welzijn. Meteen langer/strengere lockdown en we waren er vanaf zie China!*

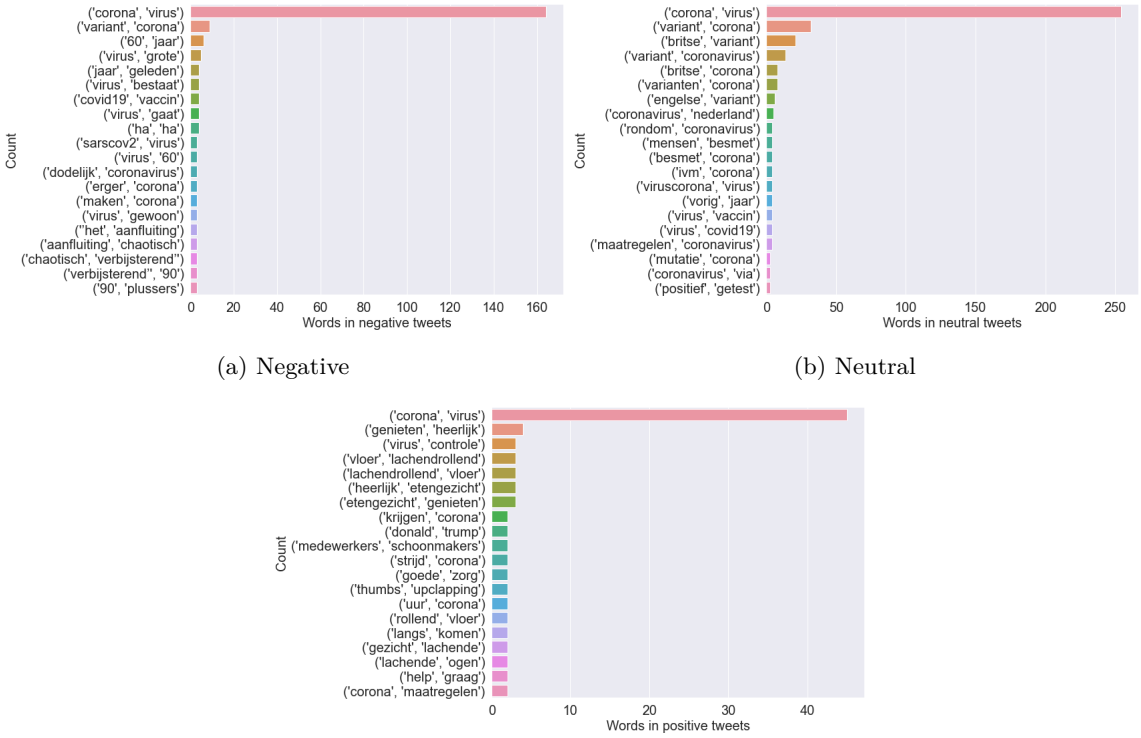
Secondly, a word cloud was created for the positively and negatively labeled tweets to see if there is a significant difference in frequent words. In Figure 9a, the following words in negative tweets stand out: “lockdown”, “avondklok” (“curfew”), “besmet” (“infected”), “maatregelen” (“measures”), “chaotisch” (“chaotic”), “verbijsterend” (“bewildering”), “aanfluiting” (“mockery”), “helaas” (“unfortunately”), “gevaarlijk” (“dangerous”), “RIVM”, and, “Plussers” (“older people”). In addition for Figure 9b , the words: “goed” (“good”), “strijd” (“fight”), “dank” (“thanks”), “geduld” (“patience”), “hulpverleners” (“aid workers”), and “zelfbeheersing” (“self-control”), stand out in the positive tweets. In conclusion, the most common terms associated with positive and negative sentiment change significantly between the word clouds (coronavirus excluded).

(a) Negative labeled tweets

(b) Positive labeled tweets

Figure 9: Word clouds.

Lastly, the frequent 20 2-bigrams are examined for the human-labeled dataset, which can be seen in Figure 10. Figure 10a shows that the terms in the bigrams for the negative tweets are quite similar to the words that appear most frequently in Figure 9a’s word cloud. In addition, the bigrams in Figure 10b for the neutral tweets are all extremely related corona. Moreover, in Figure 10c, the following 2-bigrams for the three classes stand out the most if the bigram (“corona”, “virus”) is ignored: (“vloer”, “lachendrollend”), (“etengezicht”, “genieten”), and (“goede”, “zorg”).



(a) Negative

(b) Neutral

(c) Positive

Figure 10: Most common words in tweets by frequency count.

4.2.2 Training set labeled based on polarity extraction for experiment 2

As described in the experiments in Section 3.3.1, the same tweets that humans labeled are labeled by the package `TextBlob`. This package was used to extract a polarity score from the tweets. In addition, this package is based on English texts. Therefore, the (cleaned) tweets are first translated to English with the package `deep-translator`. The polarity score for each tweet was established after translation. These scores range between -1 and 1. The following thresholds are used to categorize each tweet into the "Positive", "Neutral" and "Negative" classes:

$$sentiment(score) = \begin{cases} Negative, & \text{if } score < 0. \\ Neutral, & \text{if } score = 0. \\ Positive, & \text{if } score > 0. \end{cases} \tag{5}$$

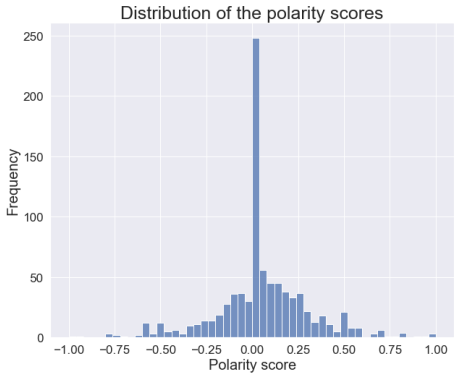
As explained in the problem statement (Section 1.2), there is some bias attached to the labels using `TextBlob`. In addition to being trained on English material, this package ignores the context of a tweet. Here's an illustration: "I tested positive for corona." The package assigns this tweet a polarity score that is relatively high, leading to the labeling of this tweet as positive while perhaps the user intended it to be negative. According to the sentiment expressed in this tweet, it should be classified as neutral.

The same tweets from the train set of the human-labeled dataset are also present in the train set of the polarity-labeled dataset. An overview of the number of classes in the train set is shown in Table 7. It is noticeable that this train set contains significantly more tweets with positive labels than the test set of the human-labeled dataset.

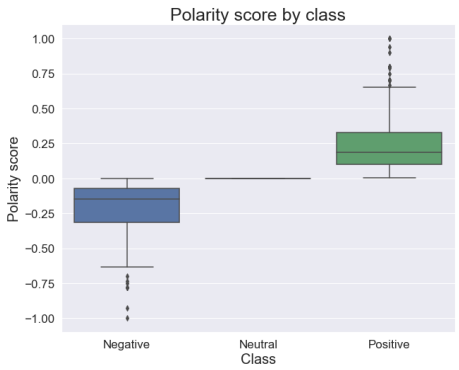
Table 7: Overview of the the classes in different sets of the polarity-labeled dataset.

	Train
Number of positives	334
Number of neutrals	176
Number of negatives	194
Size	704

To investigate the polarity scores, the distribution of these scores of the tweets are displayed in Figure 11a. It is remarkable that most tweets have positive polarity scores and that most polarity scores fall between 0 and approximately 0.04. In addition, Figure 11b depicts the distribution of scores by class to provide additional insight into the polarity scores. The median polarity score by class is very reasonable (positive: 0.19, neutral: 0.00, negative: -0.15). However, the medians of the positive and negative classes are quite close to zero. Moreover, there is slightly more variation in the polarity score of the negative tweets than the positive ones.



(a) Distribution of the polarity scores.



(b) Distribution of polarity scores by class

Figure 11: Distribution of the polarity scores from the tweets.

4.2.3 Overview of the training sets for experiments 3

The training sets for experiment 3 were produced using the human-labeled dataset, as explained in the Methodology's experiments. In experiment 3, there were two sub-experiments set up. For the first sub-experiment, the best-performing model from experiments 1 and 2 was trained on 300 and 600 tweets to examine the impact of training size on model performance.

For sub-experiment 1, the datasets have approximately the same label distribution as the human-labeled dataset. The label distribution, as illustrated in Figure 6 of the Methodology, was as follows: 34.1% of tweets were negative, 57.7 % were neutral, and 8.2 % were positive. Table 8 displays a summary of the total number of tweets for each class in the two training sets.

Table 8: Overview of the training sets for experiment 2.

	Training set 1	**Training set 2**
Number of positives	25	49
Number of neutrals	173	346
Number of negatives	102	205
Size	300	600

For the second sub-experiment, different sample techniques were used to balance the data across the labels with the goal of improving the model's performance. As mentioned in the Methodology, the training set consists of 56 positives, 404 neutrals, and 244 negatives. In the first training set, the neutrals were down-sampled from 404 tweets to 300 tweets. For the second training set, the positive tweets were up-samples from 56 to 100. The third training set consists of 100 positive tweets, 300 neutrals, and 299 negatives. Figure 12 depicts the overviews of the number of tweets per class in the training sets with different sampling techniques. It is noticeable that the datasets are still not very balanced, but the combined technique is more balanced.

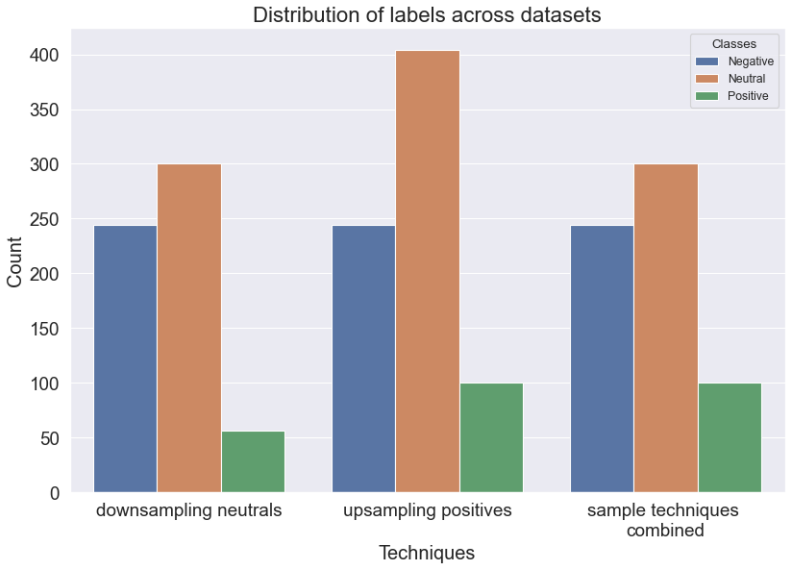


Figure 12: Distribution of labels across datasets using sampling techniques.

4.2.4 Dataset for experiment 4

The dataset is from the dates 16 January 2021 until 30 January 2021. In this period, the curfew was implemented in the Netherlands.

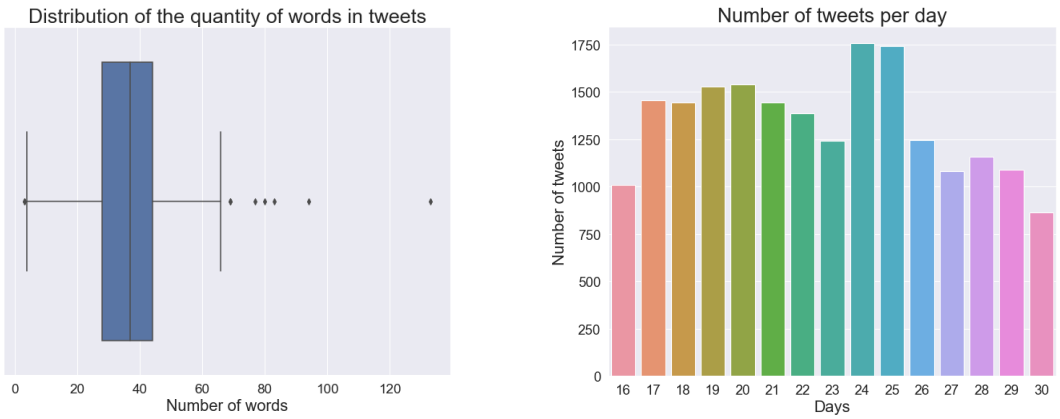
As discussed in 4.2.1, people use components to make the message of their tweets more effective with using hashtags, mentioned accounts, and emojis. Table 9 shows a summary of these components in the collected dataset for this experiment.

Table 9: Overview of the number of hashtags, mentions, and emojis in the collected dataset for experiment 4.

	Dataset
Number of hashtags	16740
Number of mentions	24905
Number of emojis	2385

Also, in this analysis, the emojis count as one word. Figure 13a shows the distribution in the length of tweets. The variation in the length of tweets ranges from 3 to 133 words. The median word length is equal to 37. In comparison with the human-labeled dataset, this dataset contains relatively more longer tweets.

In addition, the number of tweets per day is visualized in Figure 13b. This figure shows that on days 24 and 25 the most people tweeted using the keywords “het coronavirus”. This could be as a result of the fact that the implementation of the curfew was announced at the press conference on January 23, 2021.



(a) Length of tweets

(b) Number of tweets per day

Figure 13: Data analysis collected dataset for experiment 4

5 Experimental results

In this section, the results of the experiments will be discussed. First, the results of experiments 1,2 and 3 will be provided. Then, an overview of the results of these experiments is given. Lastly, the sentiment analysis on the periods January 16, 2021, to January 30, 2021, will be performed and described. The models in all the experiments are executed using the GPU of Google Colab.

5.1 Experiment 1

The first experiment aims to discover which model performs the best on the human-labeled dataset in identifying the classifications "Negative", "Neutral", and "Positive". The performance of the model will be enhanced by selectively choosing the hyperparameters for the batch size, learning rate, weight decay, and momentum. For each model, the batch sizes $\in \{4, 8, 16, 32\}$ are examined. Each batch size delivers eight possibilities to investigate for the two models, BERTje and RobBERT. The batch sizes were given to the DataBunch of each model.

As described in Section 3.3.1 of the Methodology, it was decided to test each model by batch size and by the following cyclic momentum's: 0.95-0.85 and 0.99-0.90. In addition, a modest grid search was performed to find the optimal learning rates and simultaneously the optimal values for the weight decay between the values 10^{-2}, 10^{-4}, and 10^{-6}.

After determining the optimal values for the learning rate and the weight decay, the models were trained for a maximum of 20 epochs. The small dataset may cause the model to overfit quickly. Therefore, callbacks were implemented in the model. The running was terminated after the model failed to improve the validation loss after five continuous epochs. The number five was chosen since, in some instances, the running had already been stopped after three or four epochs. As a result, the model's training was inadequate to produce accurate results on the validation set.

The results will be provided using the evaluation metrics described in Section 3.3.2. Based on the validation results, the models with the optimal values for the hyperparameters will be performed on the test set. The test results are used to select the model that performs the best on the human-labeled dataset.

5.1.1 Validation

As described above, first the optimal learning rate and weight decay were found using a grid search. Then, the model was executed on the data for a maximum of 20 epochs. Subsequently, the model was evaluated on the validation set. Appendix B consists of the visualizations of the grid search for finding optimal values for the learning rate and weight decay, the train versus the validation loss, and the confusion matrix of each model on the validation set. Additionally, an overview of the weighted average precision and recall are given in Table 21 in Appendix B.

The performance of the models with the additional hyperparameters on the validation set is summarized in Table 10. From this table, it is noticeable that BERTje (bs=16, moms=0.95-0.85)

has the highest accuracy on the validation set. However, in most instances where accuracy is comparatively higher, the model classifies most tweets as neutral. As shown in Figure 36b of Appendix B, this model can predict the majority of the neutral tweets correctly but also classifies almost all negative and positive tweets as neutral. As described in the Methodology, the data consists of many more neutrals than negative and positive tweets. As a result, the model will achieve high accuracy if it identifies every tweet as neutral, which can be a misleading result. Therefore, the accuracy seems not the best performance metric for choosing the best combination of hyperparameters for BERTje and RobBERT since this can lead to incorrect conclusions.

The weighted average F1-Score and the macro average F1-score were used to determine the best-performing model due to the imbalances in the dataset. From Table 10, it can be seen that in terms of the weighted F1-score for BERTje, that BERTje (bs=4, moms=0.99-0.90) achieves the highest weighted average and macro average F1-score. In Figure 31b, it can be observed that this model correctly classifies 15 of the 27 negative tweets, 31 of the 50 neutral tweets, and 1 of the 9 positive tweets. As previously stated, high-accuracy models are good at predicting neutrals, but it is also essential that the model recognize the other two classifications especially the positive ones.

In addition, RobBERT (bs=4, moms=0.99-0.90) and RobBERT (bs=8, moms=0.95-0.85) have both the highest accuracy and weighted average F1-score, respectively 0.52 and 0.53. Moreover, RobBERT (bs=4, moms=0.99-0.90) has a higher macro average F1-score of 0.45, which indicates that this model's performance is better on small classes like the negative and positive tweets. As shown in Figure 43b, RobBERT (bs=4, moms=0.99-0.90) correctly classifies 12 of the 27 negative tweets, 30 of the 50 neutral tweets, and 3 of the 9 positive tweets. Comparatively, RobBERT (bs=8, moms=0.95-0.85) correctly classifies 10 of the 27 negative tweets, 33 of the 50 neutral tweets, and 2 of the 9 positive tweets (see Figure 45b). Therefore, it can be concluded that RobBERT (bs=4, moms=0.99-0.90) performs better in terms of accurately categorizing on the smaller classes.

Regarding the train and validation losses of BERTje (bs=4, moms=0.99-0.90) and RobBERT (bs=4, moms=0.99-0.90), Figure 31a for BERTje shows that the validation loss corresponds to the training loss for at least 2112 processed batches (twelve epochs) before the model overfits. In addition, Figure 43a for RobBERT illustrates that before the model overfits, the validation loss tracks the training loss for at least 528 processed batches (three epochs). Therefore RobBERT overfits significantly quicker than BERTje, but requires less training epochs than BERTje to achieve about the same performance.

As a concluding observation, the confusion matrices in Appendix B show that nearly no model of BERTje successfully predicted a positive tweet. In addition to BERTje (bs=4, moms=0.99-0.90) and RobBERT (bs=4, moms=0.99-0.90), the models that were able to classify at least one positive tweet are: RobBERT (bs=4, moms=0.95-0.85), RobBERT (bs=8, moms=0.95-0.85), RobBERT (bs=8, moms=0.99-0.90), RobBERT (bs=16, moms=0.99-0.90), and RobBERT (bs=32, moms=0.95-0.85). This would indicate that RobBERT is better at classifying positive tweets.

Table 10: Overview of the results per model on the validation set in experiment 1. BS = batch size, LR = learning rate, WD = weight decay, Moms = Momentum, Acc = accuracy, Weighted F1 = Weighted average F1-score, Macro F1 = Macro average F1-score.

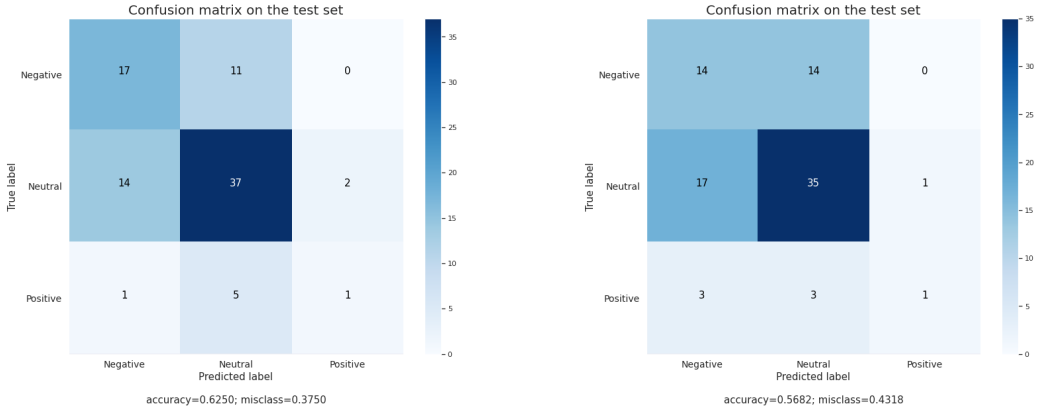
	Hyper parameters					Metrics		
Model	**BS**	**LR**	**WD**	**Moms**	**Epochs**	**Acc**	**Weighted F1**	**Macro F1**
BERTje	4	8×10^{-6}	10^{-4}	0.95-0.85	15	0.55	0.52	0.37
BERTje	4	8×10^{-6}	10^{-4}	0.99-0.90	17	0.55	**0.54**	**0.43**
BERTje	8	10^{-5}	10^{-2}	0.95-0.85	16	0.52	0.51	0.36
BERTje	8	10^{-5}	10^{-2}	0.99-0.90	15	0.52	0.51	0.36
BERTje	16	9×10^{-6}	10^{-6}	0.95-0.85	15	**0.59**	0.53	0.35
BERTje	16	9×10^{-6}	10^{-6}	0.99-0.90	17	0.50	0.48	0.35
BERTje	32	6×10^{-6}	10^{-2}	0.95-0.85	14	0.55	0.52	0.36
BERTje	32	6×10^{-6}	10^{-2}	0.99-0.90	16	0.49	0.47	0.34
RobBERT	4	4×10^{-5}	10^{-2}	0.95-0.85	9	0.50	0.53	0.43
RobBERT	4	4×10^{-5}	10^{-2}	0.99-0.90	8	**0.52**	**0.53**	**0.45**
RobBERT	8	5×10^{-5}	10^{-2}	0.95-0.85	9	**0.52**	**0.53**	0.41
RobBERT	8	5×10^{-5}	10^{-2}	0.99-0.90	7	0.50	0.52	0.39
RobBERT	16	8×10^{-5}	10^{-4}	0.95-0.85	7	0.50	0.48	0.34
RobBERT	16	8×10^{-5}	10^{-4}	0.99-0.90	11	0.45	0.43	0.42
RobBERT	32	10^{-4}	10^{-4}	0.95-0.85	11	0.48	0.49	0.38
RobBERT	32	10^{-4}	10^{-4}	0.99-0.90	10	0.45	0.43	0.32

5.1.2 Test

As mentioned above, the models BERTje (bs=4, moms=0.99-0.90) and RobBERT (bs=4, moms=0.99-0.90) perform best on the validation set. These models have been executed on the test set and the results are shown in Table 11. This table shows that BERTje outperforms the RobBERT in all the evaluation metrics. From the confusion matrices in Figure 14, it can be observed that BERTje can classify three more negative tweets and two more neutral tweets correctly than RobBERT on the test set. As a result, BERTje (bs=4, moms=0.99-0.9) performs better on the test set of the human labeled dataset.

Table 11: Overview results on the test set in experiment 1.

Model	Accuracy	Weighted avg F1	Macro avg F1
BERTje	0.62	0.62	0.49
RobBERT	0.57	0.56	0.45



(a) BERTje on the test set.

(b) RobBERT on the test set.

Figure 14: Test set results of the best performing models of BERTje and RobBERT.

5.2 Experiment 2

As described in the methodology, in this experiment, BERTje and RobBERT are trained on the polarity-labeled dataset. According to the results of experiment 1, the following models with the settings for the hyperparameters perform the best on the human-labeled dataset:

- **BERTje**
  - batch size of 4
  - learning rate of 8×10^{-6}
  - weight decay of 10^{-4}
  - momentum of 0.99-0.90
  - 17 epochs
- **RobBERT**
  - batch size of 4
  - learning rate of 4×10^{-5}
  - weight decay of 10^{-2}
  - momentum of 0.99-0.90

– 8 epochs

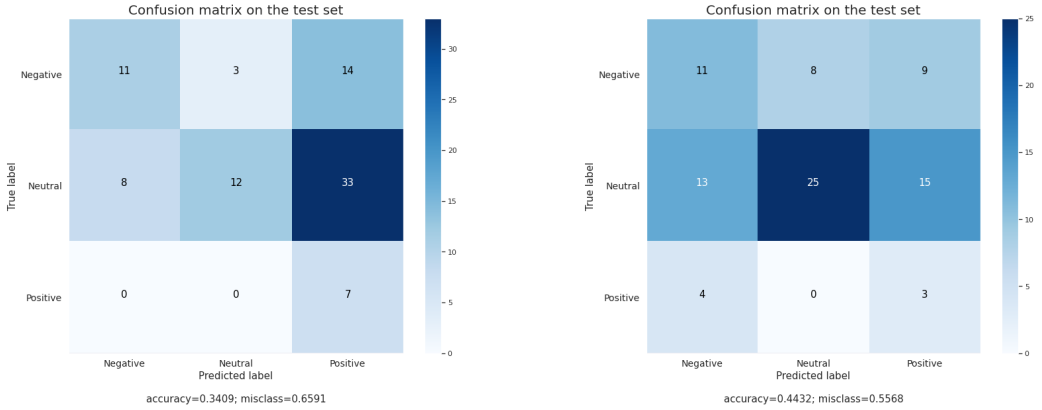
In this experiment, the models' robustness was evaluated. These two models were trained on the polarity-labeled dataset. The settings for the hyperparameters used by the models are those mentioned above. After training, the models were directly evaluated on the test set of the human-labeled dataset.

The results of these models on the test set are shown in Table 12. From this Table, it can be seen that the results for both models in all evaluation metrics clearly perform less well in comparison of the test results in experiment 1. However, RobBERT outperforms BERTje in all evaluation metrics. This may indicate that RobBERT will probably perform better if the model is trained on a biased labeled dataset than BERTje and makes RobBERT more robust in this case.

Table 12: Overview results on the test set in experiment 2.

Model	**Accuracy**	**Weighted avg F1**	**Macro avg F1**
BERTje	0.34	0.38	0.35
RobBERT	0.44	0.49	0.38

As described in Section 4 *Explanatory Data Analysis*, the dataset of tweets labeled based on the polarity scores contains a lot more positive labeled tweets than the human-labeled dataset. As a result, the models have seen significantly more tweets with positive labels during training. It is noticeable in the confusion matrices in Figure 15 that BERTje classifies all positive tweets correctly, whereas RobBERT three of them. According to the results of experiment 1, BERTje could only classify one positive tweet on the test set, while here it could categorize seven.



(a) BERTje. (b) RobBERT.

Figure 15: Test set results of the best performing models of BERTje and RobBERT.

5.3 Experiment 3

Experiment 3 is divided in two sub-experiments. As explained in the Methodology, the models were trained on training sets of varying sizes and of different sampling techniques to examine how this affects performance. Also in this experiment, the models with the best performing setting of hyper parameters from experiment 1 were used. Appendix C provides the confusion matrices of the results on the test sets for both sub-experiments.

5.3.1 Various sizes of the training set

For the first sub-experiment, the models are trained on training sets of 300 and 600 samples. The results of these models on the test set of the human-labeled dataset are shown in Table 13. From this table, it can be seen that for both models that the difference in training sizes do have an impact on the model's performance. Both models perform better when the training size increased.

Table 13: Overview results on the test set using different sample techniques in experiment 3.

Training size	Model	Accuracy	Weighted avg F1	Macro avg F1
300	BERTje	0.58	0.55	0.37
	RobBERT	0.51	0.50	0.34
600	BERTje	0.62	0.60	0.42
	RobBERT	0.60	0.58	0.40

5.3.2 Sampling tweets in training set

For the second sub-experiment, the models are first trained on the training set where the neutral tweets are downsampled from 404 tweets to 300 tweets. Second, the models are trained using the training set of positive tweets that were upsampled from 56 to 100. Lastly, the models are trained on the combined sample techniques, where the neutrals are downsampled and the positives upsampled. In this experiment, the models with the best performing hyperparameter settings from experiment 1 were also employed.

The results of the models on the test set are shown in Table 14. From Table 14, it can be seen that when RobBERT is trained on the combined sample techniques it even better performs than in experiment 1. BERTje's performance did not improve using the sampling methods.

Table 14: Overview results on the test set using different sample techniques in experiment 3.

Technique	Model	Accuracy	Weighted avg F1	Macro avg F1
Downsampling	BERTje	0.59	0.58	0.40
	RobBERT	0.53	0.52	0.36
Upsampling	BERTje	0.61	0.58	0.39
	RobBERT	0.55	0.53	0.36
Combined	BERTje	0.61	0.59	0.40
	RobBERT	0.59	0.57	0.45

5.4 Overview of the best results experiment 1-3

Finally, this section will provide an overview of the results from experiments 1 to 3 in the evaluation metrics: accuracy, weighted average F1-score, and macro average F1-score. In experiment 4, a new set of tweets will be classified using the best models. The project aims to identify a model that can reasonably predict across all classes. As a result, the macro F1-score evaluation metric will be the main focus in choosing the best models for experiment 4.

An overview of the best results on the test set of the human-labeled dataset for experiments 1 to 3 is shown in Table 15. From this Table, it is noticeable that BERTje from experiment 1 performs best on the test set compared to experiments 2 and 3. Therefore, BERTje trained on the human-labeled dataset from experiment 1 is chosen for experiment 4. Additionally, it can be seen that employing the downsampling and upsampling strategy for the train set (experiment 3: sub-experiment 2) improves RobBERT's performance compared to experiment 1 in terms of accuracy and weighted average F1-score. As a result, RobBERT trained on the dataset that combines downsampling neutrals and upsampling positives is selected for experiment 4 as well.

Table 15: Overview of results on the test set of the human-labeled dataset for experiment 1-3.

Experiment	Model	Accuracy	Weighted avg F1	Macro avg F1
1	BERTje	0.62	0.62	0.49
	RobBERT	0.57	0.56	0.45
2	BERTje	0.34	0.38	0.35
	RobBERT	0.44	0.49	0.38
3: sup-exp 1	BERTje	0.62	0.60	0.42
	RobBERT	0.60	0.58	0.40
3: sup-exp 2	BERTje	0.61	0.58	0.39
	RobBERT	0.59	0.57	0.45

5.5 Experiment 4

This section provides the sentiment analyses performed on the collected dataset from January 16 to January 30, 2021. The best performing models for BERTje and RobBERT from previous experiments are used for these analyses. The results of the two models on this dataset will be compared in this section. As described in the Methodology, these models give as output a probability per class. The tweets are classified depending on which class has the highest probability. In this section, the characteristics of the dataset are analyzed first. In addition, the number of classified tweets of the given period is examined. Lastly, the most negative and positive tweets based on the highest probabilities are investigated.

5.5.1 Characteristics of the dataset

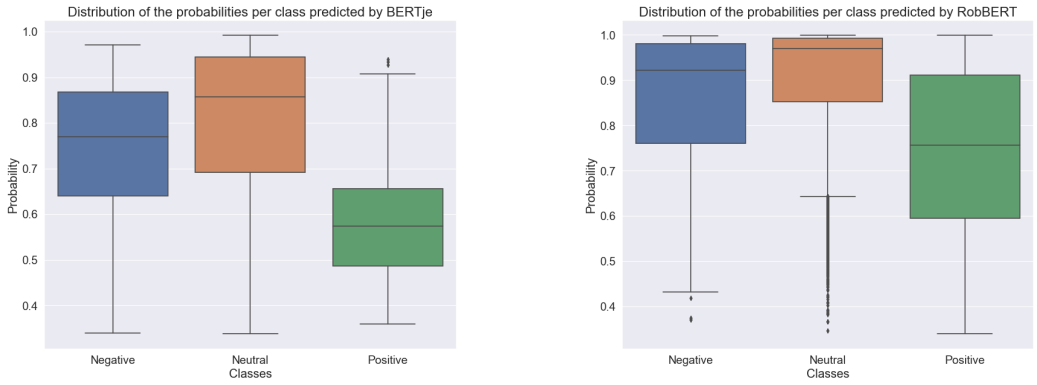
For the characteristics of the collected data, an overview of the number of predicted classes of each model is provided first. Following this, the probabilities of the tweets that the models assigned are examined.

Table 16 shows the number of predicted classes per model on the dataset. As can be seen in this table, BERTje predicted more tweets as negative than RobBERT, whereas RobBERT classified more neutrals and positives.

Table 16: Overview the number of predicted classes per model.

	BERTje	**RobBERT**
Number of negatives	9183	7202
Number of neutrals	10440	12297
Number of positives	354	478
Total	19977	19977

To have a better understanding of the predicted classes and the probabilities of the two models assigned to the tweets, the probabilities for each class are examined. As explained in the Methodology, the models provide a range of probabilities. An illustration of such a range is [0.990, 0.008, 0.002], where the probabilities are for the classes negative, neutral, and positive, respectively. For this analysis, the data were filtered by class and the corresponding probability of this class. Similar to the previous example, for the negative class, only the probability of 0.990 was taken into account. Figure 16 shows the distribution of the corresponding probabilities by class per model. As can be seen in this figure, the predicted probabilities by RobBERT (median of the classes: 0.94, 0.98, 0.73, respectively) tend to be higher per class than the probabilities by BERTje (median of the classes: 0.77, 0.86, 0.58, respectively). This indicates that the tweets classified by RobBERT often have a more apparent probability of being negative, neutral, or positive. Moreover, there is a lot more variation in the probabilities of the positive class by RobBERT in comparison with the probabilities of the positive class by BERTje.

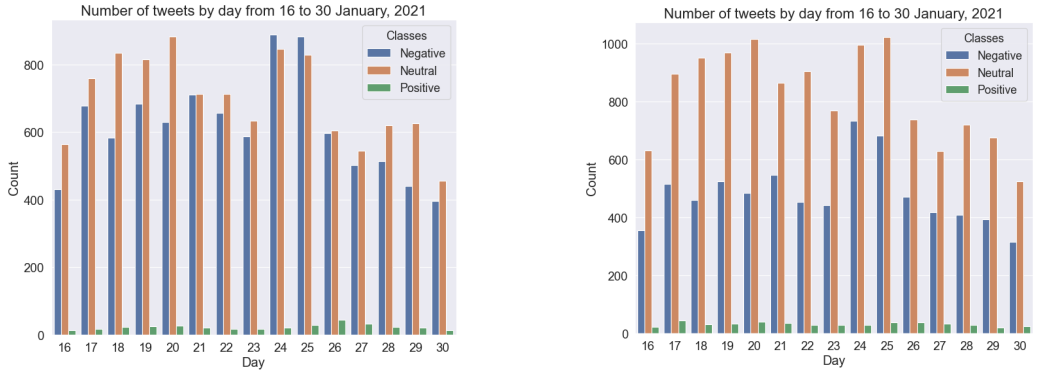


(a) BERTje. (b) RobBERT.

Figure 16: Distribution of the probabilities to the corresponding classes per model.

5.5.2 The number of classified tweets over time

In order to provide more information for the number of tweets in sentiment, Figure 13b displays the daily number of tweets from January 16 to January 30, 2021. According to the RIVM, on January 20, 2021, at 7:00 PM, it was announced that the curfew would be implemented on January 23 [76]. The Figures 17a and 17b show that the negative sentiment is starting to rise from day 21 and is dropping reasonably again on day 22. From days 24 and 25, the number of negative tweets increases dramatically and reaches the maximum number of tweets in the two weeks in both figures.



(a) BERTje. (b) RobBERT.

Figure 17: Number of tweets by day from 16 to 30 January.

To better understand the period indicated above, the number of tweets categorized by the models was analyzed further. Figure 18 displays the number of tweets classified by BERTje from January 19 to January 24, 2021, by class, per day, and per hour. Additionally, Figure 19 gives insight into RobBERT's hourly and daily tweet volume by the class from January 19 to January 24, 2021. In

general, both figures follow quite similar daily trends, indicating that the models classify most tweets similarly. It is noteworthy that during this time, RobBERT detected more neutral tweets than BERTje did each day, despite BERTje consistently identifying more negative tweets.

For both figures, one would expect an increase in negative tweets on day 20 or possibly as early as day 19 (if the press conference information is leaked earlier). However, it is noticeable, as with BERTje and with RobBERT, that on day 24, the number of negative tweets is almost the same every hour. Additionally, it is apparent that on January 24, compared to previous days, a disproportionately higher number of negative tweets were posted, which may have resulted from the curfew's implementation.

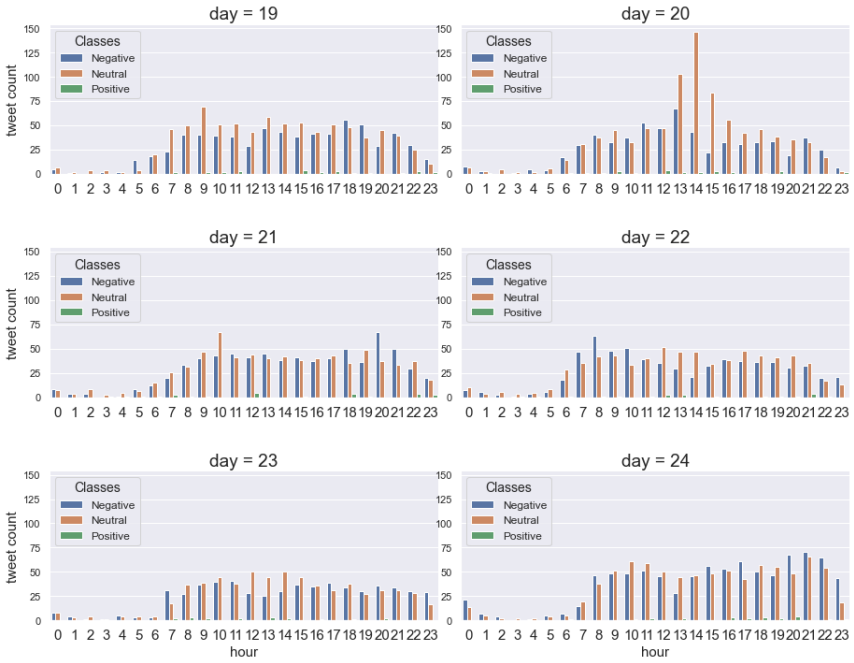


Figure 18: The number of tweets per class by BERTje from January 19 to January 24, 2021

Figure 19: The number of tweets per class by RobBERT from January 19 to January 24, 2021

5.5.3 Most negative and positive tweets

In this analysis, it is examined which tweets had the highest probability of the negative tweets and the positive tweets. Furthermore, this analysis determines which terms characterize the most negative and positive tweets. Unique tweets are considered, not the replies.

Table 17 lists the most negative tweets according to BERTje, including the associated probabilities. The table shows that BERTje correctly classified the three tweets as negative. Additionally, Table 18 shows the most positive tweets classified by BERTje. From this table, it can be seen that the first tweet seems to be negative, which indicates that it is not well predicted. However, the second and third tweets in this table were correctly predicted as positive.

Table 17: Most negative tweets of BERTje

Tweets	**Probability**
Wakker worden mensen! Alles heeft een reden...en Corona is het niet! 1 van de redenen waarom bijv het referenda ook is afgenomen....	0.9685
Bedenk wel : Niets is wat het lijkt.....ook de corona crisis niet !!	0.9684
Vandaag regent het maar op drukke momenten (steeds meer door toename bevolking en corona) is dit een levensgevaarlijke route ! Breedte fietspad voldoet niet aan normen. Overheid kijkt weg. Nog meer mensen aantrekken dus niet zo verstandig.	0.9683

Table 18: Most positive tweets of BERTje

Tweets	**Probability**
RIVM meldt vandaag voor het land 39 overleden coronapatiënten, onder wie twee coronapatiënten uit Schiedam en een coronapatiënt uit Vlaardingen te betreuren zijn @rivm #coronavirus	0.9391
Corona heeft meer dan ooit laten zien hoe belangrijk de zorgsector is. Elke dag zetten duizenden zorg- en welzijnshelden zich in voor een ander. Vandaag vieren we daarom De dag van het respect voor de zorg met de boodschap: 'Jullie zijn fantastisch!' #respectvoordezorg	0.8844
Mijn ouders zijn gisteren ingeënt tegen Corona. Het einde is stilaan in zicht! 😊	0.8825

The most negative tweets classified by RoBERT are seen in Table 19. The first tweet in this table is meant to be negative, but it must also be amusing. Furthermore, the second and third tweets are much more obviously negative. In addition, Table 20 shows the most positive tweets classified by RobBERT. According to the tweets in this table, RobBERT accurately predicted these tweets to be positive.

Table 19: Most negative tweets of RobBERT

Tweets	**Probability**
Als er geen Corona was, hadden wij wel een nieuwjaarsduik in het openluchtzwembad willen nemen. Probleem is... Dat zwembad is er nog steeds niet!! #almere #makealmeregreatagain	0.99829
Wij maken ons druk over de Engels en Haga's. In de VS krijgt een wappie gewoon de ruimte om miljoenen mensen op de mouw te spelden dat door een foutje van Fauci het Corona virus is ontstaan.... 🙄	0.99822
We gaan komend jaar meemaken dat er veel kinderen en jongeren met leerachterstand bij komen. Wie zegt dat dit niet gebeurd mag wat mij betreft zelf terug naar school. Gevolgen van probleem zoals Corona virus worden echt zichtbaar als het voorbij is.	0.99803

Table 20: Most positive tweets of RobBERT

Tweets	**Probability**
Laten we vandaag ook niet onze zorgverleners, ambulance medewerkers, schoonmakers en andere helden vergeten, die iedere dag vechten tegen #corona virus. Dank jullie wel 👍 👋 En laten wij helpen: hou afstand, verminder bezoek, werk thuis, het is aan ons om de pandemie te stoppen	0.99868
#coronavirus daagt me uit om nog creatiever te zijn/worden! Wat doet het corona virus met jullie?	0.99835
Vandaag op de GGD locatie op Schiphol de 1e vaccinatie tegen Covid-19 ontvangen. Over 3 weken de 2e. 💪 Alles goed gegaan en alles is uitstekend geregeld op de locatie. Samen kunnen we het corona-virus verslaan. Prikken is voor je eigen gezondheid en die van andere! #ikprikhetwel	0.99700

6 Conclusion

This conclusion summarizes the findings of the four experiments performed for this thesis research. Moreover, these experiments used the Dutch language models BERTje and RobBERT to discover sentiment among COVID-19-related tweets. Comparing the outcomes of the four experiments assists in answering the main research question: "How can Dutch sentiment on the social media platform Twitter related to COVID-19 be detected, and how does the sentiment evolve over time?" Additionally, the model's effectiveness in terms of the quality of the labeled dataset needs to be determined. The concluding remarks will also dive into the performance of those models on smaller training sets and training sets using sampling methods. Finally, comparing the outcomes from the sentiment analysis on both models will conclude this research.

Four experiments were conducted in this research to answer the research question and the sub-questions. In the first experiment, the two Dutch language models, BERTje and RobBERT, were constructed. This experiment concluded that both models performed better using small batch sizes in this research. In addition, the models also performed better when using the cyclical momentum of 0.99-0.90. As a result, these models used these values for the hyperparameters and the obtained learning rate and weight decay in experiments 2 to 4. The main finding of this experiment was that BERTje performs slightly better on the human-labeled dataset than RobBERT. However, RobBERT could still correctly predict positive tweets with the majority of combinations of hyperparameter settings. In contrast, BERTje could only predict one positive tweet with one hyperparameter setting.

In experiment 2, the models were trained on the polarity-labeled dataset with much bias in the labels. According to the results of experiment 2, RobBERT outperformed BERTje when trained on this polarity-labeled dataset. This result would suggest that RobBERT is more robust in using biased labels as model inputs. Furthermore, both models predicted all of the positive tweets on the produced test set of the human-labeled dataset, which was most likely due to the higher number of positive instances in the training set of the polarity-labeled dataset. In conclusion, both models underperformed when trained on a training set with biased labels.

In experiment 3, the first sub-experiment revealed that BERTje outperformed RobBERT in training the models on smaller sets. However, the literature on the construction of RobBERT described that RobBERT performed exceptionally well on smaller datasets. The results of this experiment also showed that both models' performance improved as training sizes increased. The second sub-experiment found that combining the techniques of downsampling neutral and upsampling positive tweets increases RobBERT's performance on the test set compared to experiment 1. Moreover, none of the sample methods enhanced BERTje's performance. Based on experiments 1, 2, and 3, BERTje trained on the human-labeled dataset was chosen for performing sentiment analysis in experiment 4. In addition, RobBERT trained on the human-labeled using downsampling and upsampling methods was also used for experiment 4.

Lastly, in experiment 4, sentiment analysis was performed using the best-performing models of BERTje and RobBERT on a collected Twitter dataset from January 16 to January 30, 2021. This experiment concluded that BERTje predicted more tweets as negative than RobBERT. Furthermore, RobBERT classified more neutrals as positives. Additionally, RobBERT more often assigned a higher probability to tweets to be negative, neutral, or positive. Both models showed, as expected

that sentiment changed after the introduction of the curfew. This outcome would imply that the measurement impacted the overall mood on Twitter.

In conclusion, this thesis covers Dutch sentiment analysis on COVID-19-related tweets. Referring to the research question, BERTje and RobBERT are two Dutch language models that can detect Dutch sentiment in tweets regarding COVID-19. BERTje is probably the most suitable for detecting sentiment among Dutch COVID-19-related tweets, according to the terms and standards of the evaluation measures provided in this study. This research showcases the current capabilities of what Dutch sentiment analysis can extract from tweets. Moreover, it brings Avanade a step closer to determining an approach for building a model for the task of Dutch sentiment analysis and widening their assortment on future endeavors for related NLP projects.

7 Discussion

Following the examination of the research's findings in the previous section, this section will discuss the limitations and potential improvements for future work in the field of Dutch sentiment analysis concerning tweets about COVID-19.

In this research, sentiment analysis is performed with the help of the models BERTje and RobBERT on Twitter data sets. Both models performed reasonably well on tweets. Tweets, in general, are far more difficult to classify than, for instance, texts that contain almost no spelling errors. On the COVID-19-related tweets for this project, BERTje performed slightly better than RobBERT in terms of the evaluation metrics. Additionally, RobBERT showed to be more robust when trained on a labeled dataset that contains bias. The literature claims that RobBERT is currently the best-performing Dutch language model. However, it seems that not much research has been done so far for using RobBERT to perform sentiment analysis on Dutch tweets. For performing Dutch sentiment analysis on Twitter, it would be recommended to further examine RobBERT. Although, given the results of this project, additional research on BERTje would also be advised.

The biggest limitation of this research was the availability of labeled data. In this project, labeling experiments were conducted by colleagues of Avanade. The disadvantage of labeling data by humans is that it is time-consuming. As mentioned in the conclusion, the quality of the labeled data and the training size do have an impact on the model's performance. Therefore, a recommendation for Avanade would be to investigate such learning algorithms as Active Learning for helping the labeling task, as mentioned in the literature. In addition, a model's performance might be enhanced by a learning algorithm that can help determine which texts are relevant to the model. Moreover, the algorithm would assist speed up data labeling, making labeling experiments less time-consuming. Furthermore, data augmentation could be considered for generating more data points.

The tweets were obtained when there was not much optimism about the coronavirus, which affected people's lives and resulted in a minority of positive tweets in the dataset. As a result, the data in this project showed an imbalance in the classes. From the experiments, it was demonstrated that RobBERT's performance slightly improved by implementing sampling techniques. As a result, it would be recommended to investigate other data balancing techniques to improve the model's performance.

The change in general sentiment on Twitter can be traced using sentiment analysis. In this project, a labeled data set of two weeks in January 2021 was created to investigate if certain events would affect the overall sentiment on Twitter. However, the sentiment analysis was only performed during a period when the government took unpopular measures fighting the outbreak numbers of Corona patients. It would be good to repeat this research in a period where the government takes measures to reopen society.

References

[1] Avanade. *About Avanade*. https://www.avanade.com/en-us/about-avanade.

[2] José Van Dijck and Donya Alinejad. "Social media and trust in scientific expertise: Debating the Covid-19 pandemic in the Netherlands". In: *Social Media+ Society* 6.4 (2020), p. 2056305120981057.

[3] Bing Liu. "Sentiment analysis and opinion mining". In: *Synthesis lectures on human language technologies* 5.1 (2012), pp. 1–167.

[4] Tetsuya Nasukawa and Jeonghee Yi. "Sentiment analysis: Capturing favorability using natural language processing". In: *Proceedings of the 2nd international conference on Knowledge capture*. 2003, pp. 70–77.

[5] Laith Abualigah et al. "Sentiment analysis in healthcare: a brief review". In: *Recent Advances in NLP: The Case of Arabic Language* (2020), pp. 129–141.

[6] Rakhi Batra and Sher Muhammad Daudpota. "Integrating StockTwits with sentiment analysis for better prediction of stock price movement". In: *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*. IEEE. 2018, pp. 1–5.

[7] Rijksoverheid. *Verdere versoepelingen coronamaatregelen*. https://www.rijksoverheid.nl/actueel/nieuws/2022/03/15/verdere-versoepelingen-coronamaatregelen.

[8] Talha Khan Burki. "Coronavirus in China". In: *The Lancet. Respiratory Medicine* 8.3 (2020), p. 238.

[9] MCM De Jong et al. "Intra-and interspecies transmission of H7N7 highly pathogenic avian influenza virus during the avian influenza epidemic in The Netherlands in 2003". In: *Revue scientifique et technique/Office International des Epizooties* 28.1 (2009), pp. 333–340.

[10] Rijksoverheid. *Vogelgriep*. https://www.rivm.nl/aviare-influenza.

[11] Marius Gilbert et al. "Income disparities and the global distribution of intensively farmed chicken and pigs". In: *PLoS One* 10.7 (2015), e0133381.

[12] HIJ Roest et al. "The Q fever epidemic in The Netherlands: history, onset, response and reflection". In: *Epidemiology & Infection* 139.1 (2011), pp. 1–12.

[13] Elizabeth D Liddy. "Natural language processing". In: (2001).

[14] Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman. "Natural language processing: an introduction". In: *Journal of the American Medical Informatics Association* 18.5 (2011), pp. 544–551.

[15] Nitin Indurkhya and Fred J Damerau. *Handbook of natural language processing, Chapter: Classical Approaches to Natural Language Processing*. Chapman and Hall/CRC, 2010, pp. 3–7.

[16] IBM. *Natural Language Processing (NLP)*. `https://www.ibm.com/cloud/learn/natural-language-processing`.

[17] Diksha Khurana et al. "Natural language processing: State of the art, current trends and challenges". In: *Multimedia Tools and Applications* (2022), pp. 1–32.

[18] Yelena Mejova. "Sentiment analysis: An overview". In: *University of Iowa, Computer Science Department* (2009).

[19] Mika V Mäntylä, Daniel Graziotin, and Miikka Kuutila. "The evolution of sentiment analysis—A review of research topics, venues, and top cited papers". In: *Computer Science Review* 27 (2018), pp. 16–32.

[20] Nitin Indurkhya and Fred J Damerau. *Handbook of natural language processing, Chapter: Sentiment Analysis and Subjectivity*. Chapman and Hall/CRC, 2010, pp. 627–666.

[21] Bo Pang, Lillian Lee, et al. "Opinion mining and sentiment analysis". In: *Foundations and Trends® in information retrieval* 2.1–2 (2008), pp. 1–135.

[22] Janyce Wiebe et al. "Learning subjective language". In: *Computational linguistics* 30.3 (2004), pp. 277–308.

[23] Vasileios Hatzivassiloglou and Janyce Wiebe. "Effects of adjective orientation and gradability on sentence subjectivity". In: *COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics*. 2000.

[24] Farah Benamara et al. "Sentiment analysis: Adjectives and adverbs are better than adjectives alone." In: *ICWSM* 7 (2007), pp. 203–206.

[25] AJP Manoj Prasad Jayaweera and Naomal GJ Dias. "Hidden markov model based part of speech tagger for sinhala language". In: *arXiv preprint arXiv:1407.2989* (2014).

[26] Bing Liu. "Opinion mining and sentiment analysis". In: *Web Data Mining*. Springer, 2011, pp. 459–526.

[27] John Blitzer, Mark Dredze, and Fernando Pereira. "Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification". In: *Proceedings of the 45th annual meeting of the association of computational linguistics*. 2007, pp. 440–447.

[28] Kamal Nigam and Matthew Hurst. "Towards a robust metric of opinion". In: *AAAI spring symposium on exploring attitude and affect in text*. Vol. 598603. American Association for Artificial Intelligence Menlo Park, CA, USA. 2004.

[29] Kushal Dave, Steve Lawrence, and David M Pennock. "Mining the peanut gallery: Opinion extraction and semantic classification of product reviews". In: *Proceedings of the 12th international conference on World Wide Web*. 2003, pp. 519–528.

[30] Arman Khadjeh Nassirtoussi et al. "Text mining for market prediction: A systematic review". In: *Expert Systems with Applications* 41.16 (2014), pp. 7653–7670.

[31] Pete Burnap et al. "Tweeting the terror: modelling the social media reaction to the Woolwich terrorist attack". In: *Social Network Analysis and Mining* 4.1 (2014), pp. 1–14.

[32] Antonio Reyes and Paolo Rosso. "On the difficulty of automatically detecting irony: beyond a simple case of negation". In: *Knowledge and Information Systems* 40.3 (2014), pp. 595–614.

[33] Alexander Hogenboom et al. "Multi-lingual support for lexicon-based sentiment analysis guided by semantics". In: *Decision support systems* 62 (2014), pp. 43–53.

[34] Erik Cambria et al. "An ELM-based model for affective analogical reasoning". In: *Neurocomputing* 149 (2015), pp. 443–455.

[35] Bernard J Jansen et al. "Twitter power: Tweets as electronic word of mouth". In: *Journal of the American society for information science and technology* 60.11 (2009), pp. 2169–2188.

[36] Jun-Ming Xu et al. "Learning from bullying traces in social media". In: *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*. 2012, pp. 656–666.

[37] Marc Cheong and Vincent Lee. "A microblogging-based approach to terrorism informatics: Exploration and chronicling civilian sentiment and response to terrorism events via Twitter". In: *Information Systems Frontiers* 13.1 (2011), pp. 45–59.

[38] Nicholas A Diakopoulos and David A Shamma. "Characterizing debate performance via aggregated twitter sentiment". In: *Proceedings of the SIGCHI conference on human factors in computing systems*. 2010, pp. 1195–1198.

[39] Andrés Montoyo, Patricio Martínez-Barco, and Alexandra Balahur. "Subjectivity and sentiment analysis: An overview of the current state of the area and envisaged developments". In: *Decision Support Systems* 53.4 (2012), pp. 675–679.

[40] Luiz FS Coletta et al. "Combining classification and clustering for tweet sentiment analysis". In: *2014 Brazilian conference on intelligent systems*. IEEE. 2014, pp. 210–215.

[41] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[42] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[43] Nagesh Singh Chauhan. *Google BERT: Understanding the Architecture*. `https://www.theaidream.com/post/google-bert-understanding-the-architecture`.

[44] Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. "BERTweet: A pre-trained language model for English Tweets". In: *arXiv preprint arXiv:2005.10200* (2020).

[45] Yinhan Liu et al. "Roberta: A robustly optimized bert pretraining approach". In: *arXiv preprint arXiv:1907.11692* (2019).

[46] Armand Joulin et al. "Bag of tricks for efficient text classification". In: *arXiv preprint arXiv:1607.01759* (2016).

[47] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.", 2009.

[48] Rico Sennrich, Barry Haddow, and Alexandra Birch. "Neural machine translation of rare words with subword units". In: *arXiv preprint arXiv:1508.07909* (2015).

[49] Wietse de Vries et al. "Bertje: A dutch bert model". In: *arXiv preprint arXiv:1912.09582* (2019).

[50] Xinying Song et al. "Fast wordpiece tokenization". In: *arXiv preprint arXiv:2012.15524* (2020).

[51] Taku Kudo and John Richardson. "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing". In: *arXiv preprint arXiv:1808.06226* (2018).

[52] Hugging Face. *Summary of the tokenizers.* `https://huggingface.co/docs/transformers/tokenizer_summary`.

[53] Kaj Bostrom and Greg Durrett. "Byte pair encoding is suboptimal for language model pretraining". In: *arXiv preprint arXiv:2004.03720* (2020).

[54] Benjamin Van der Burgh and Suzan Verberne. "The merits of Universal Language Model Fine-tuning for Small Datasets–a case with Dutch book reviews". In: *arXiv preprint arXiv:1910.00896* (2019).

[55] Pieter Delobelle, Thomas Winters, and Bettina Berendt. "Robbert: a dutch roberta-based language model". In: *arXiv preprint arXiv:2001.06286* (2020).

[56] Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. "Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures". In: *7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7)*. Leibniz-Institut für Deutsche Sprache. 2019.

[57] A Brandsen et al. "BERT-NL a set of language models pre-trained on the Dutch SoNaR corpus". In: *Dutch-Belgian Information Retrieval Conference (DIR 2019)*. [Sl: sn]. 2019.

[58] Chi Sun et al. "How to fine-tune bert for text classification?" In: *China national conference on Chinese computational linguistics*. Springer. 2019, pp. 194–206.

[59] Pieter Delobelle, Thomas Winters, and Bettina Berendt. "RobBERTje: a Distilled Dutch BERT Model". In: *31st Meeting of Computational Linguistics in The Netherlands (CLIN 31), Date: 2021/07/09-2021/07/09, Location: Ghent, Belgium*. 2021, pp. 1–6.

[60] Joppe Valentijn Wouts. "Text-based classification of interviews for mental health–juxtaposing the state of the art". In: *arXiv preprint arXiv:2008.01543* (2020).

[61] Adrian de Wynter and Daniel J Perry. "Optimal subarchitecture extraction for bert". In: *arXiv preprint arXiv:2010.10499* (2020).

[62] Victor Sanh et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *arXiv preprint arXiv:1910.01108* (2019).

[63] Ibrahem Kandel and Mauro Castelli. "The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset". In: *ICT express* 6.4 (2020), pp. 312–315.

[64] Leslie N Smith. "A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay". In: *arXiv preprint arXiv:1803.09820* (2018).

[65] Todd Kulesza et al. "Structured labeling for facilitating concept evolution in machine learning". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2014, pp. 3075–3084.

[66] Ankush Singla, Elisa Bertino, and Dinesh Verma. "Overcoming the lack of labeled data: Training intrusion detection models using transfer learning". In: *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*. IEEE. 2019, pp. 69–74.

[67] Burr Settles. "Active learning literature survey". In: (2009).

[68] JustAnotherArchivist. *snscrape*. `https://github.com/JustAnotherArchivist/snscrape`.

[69] Luciana Abednego, Cecilia E Nugraheni, and Ame Fedora. "Forex Sentiment Analysis with Python". In: *International Journal of Advanced Research in Economics and Finance* 4.1 (2022), pp. 46–55.

[70] Microsoft. *Azure Machine Learning, een service op bedrijfsniveau gebruiken voor de end-to-end machine learning-levenscyclus*. `https://azure.microsoft.com/nl-nl/services/machine-learning/`.

[71] Jeremy Howard. *Welcome to fastai*. `https://github.com/fastai/fastai`.

[72] Jonathan J Webster and Chunyu Kit. "Tokenization as the initial phase in NLP". In: *COLING 1992 volume 4: The 14th international conference on computational linguistics*. 1992.

[73] Christopher Brousseau, Justin Johnson, and Curtis Thacker. "Machine Learning Based Chat Analysis". In: *Code4Lib Journal* 50 (2021).

[74] Mayara Khadhraoui et al. "Survey of BERT-Base Models for Scientific Text Classification: COVID-19 Case Study". In: *Applied Sciences* 12.6 (2022), p. 2891.

[75] Akash Shastri Towards Data Science. *Why I use Fastai and you should too.* `https://towardsdatascience.com/why-i-use-fastai-and-you-should-too-a421f6c99508`.

[76] Rijksoverheid. *Corona virus, Januari 2021: Invoering avondklok en start vaccinatie*. `https://www.rijksoverheid.nl/onderwerpen/coronavirus-tijdlijn/januari-2021-invoering-avondklok-en-start-vaccinatie`.

[77] Mohammad Hossin and Md Nasir Sulaiman. "A review on evaluation metrics for data classification evaluations". In: *International journal of data mining & knowledge management process* 5.2 (2015), p. 1.

[78] Christopher D Manning. *Introduction to information retrieval*. Syngress Publishing, 2008.

Appendix

This is the appendix contains part A, B, and C.

Appendix A

Appendix A contains the representations of the labeling experiments in Azure Machine Learning.

Labeling experiment 1

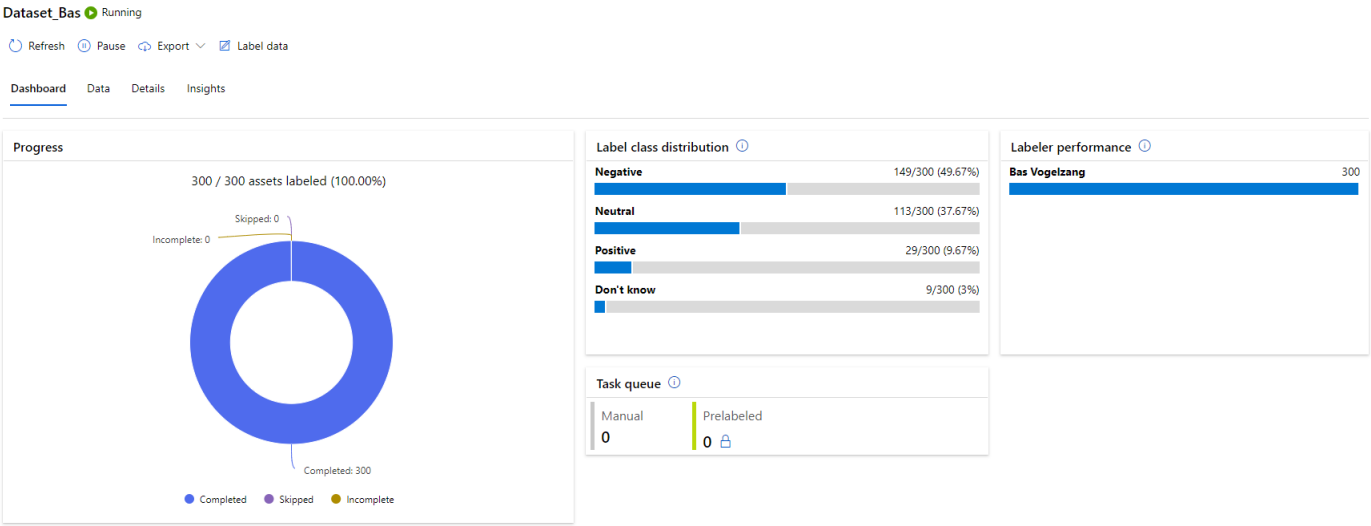


Figure 20: Labeling experiment 1: overview of Bas's labels.

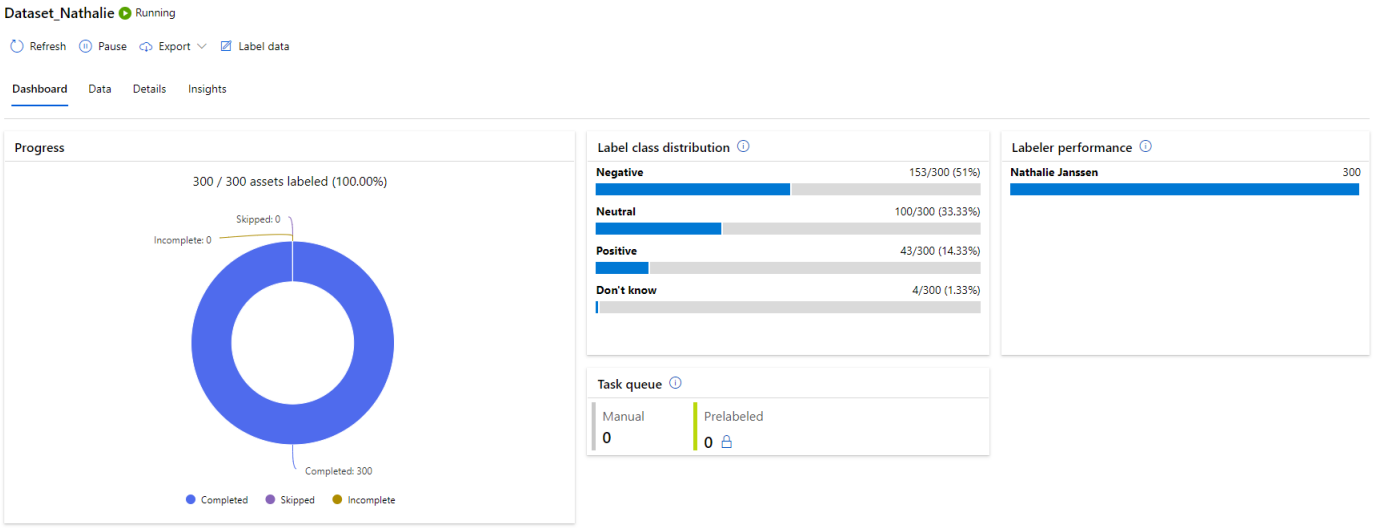


Figure 21: Labeling experiment 1: overview of Nathalie's labels.

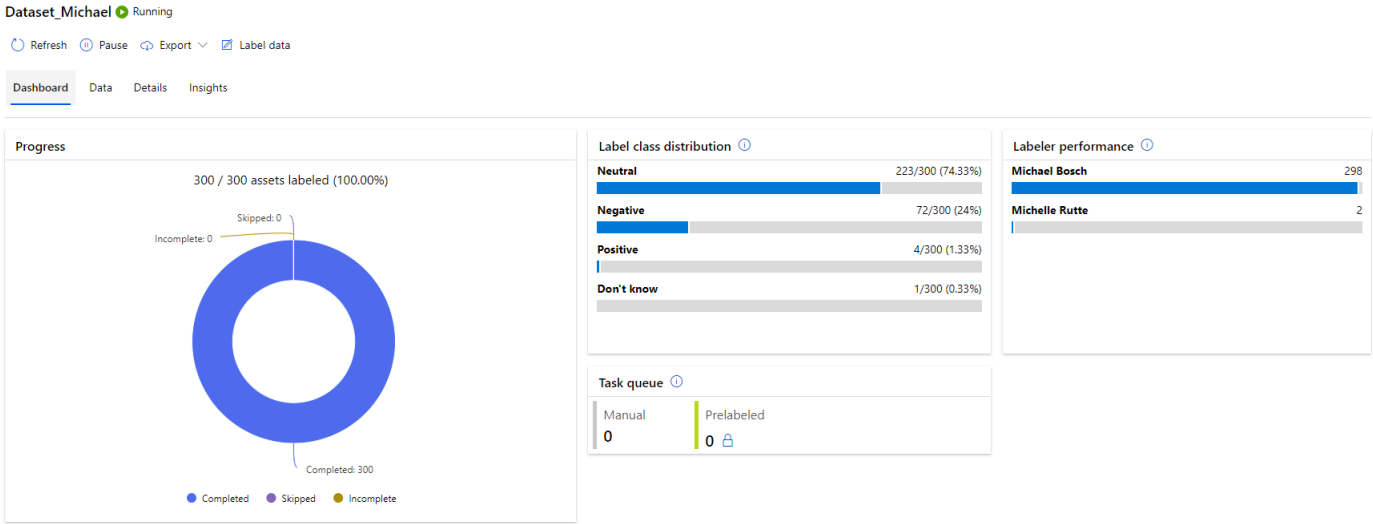


Figure 22: Labeling experiment 1: overview of Michael's labels.

Labeling experiment 2

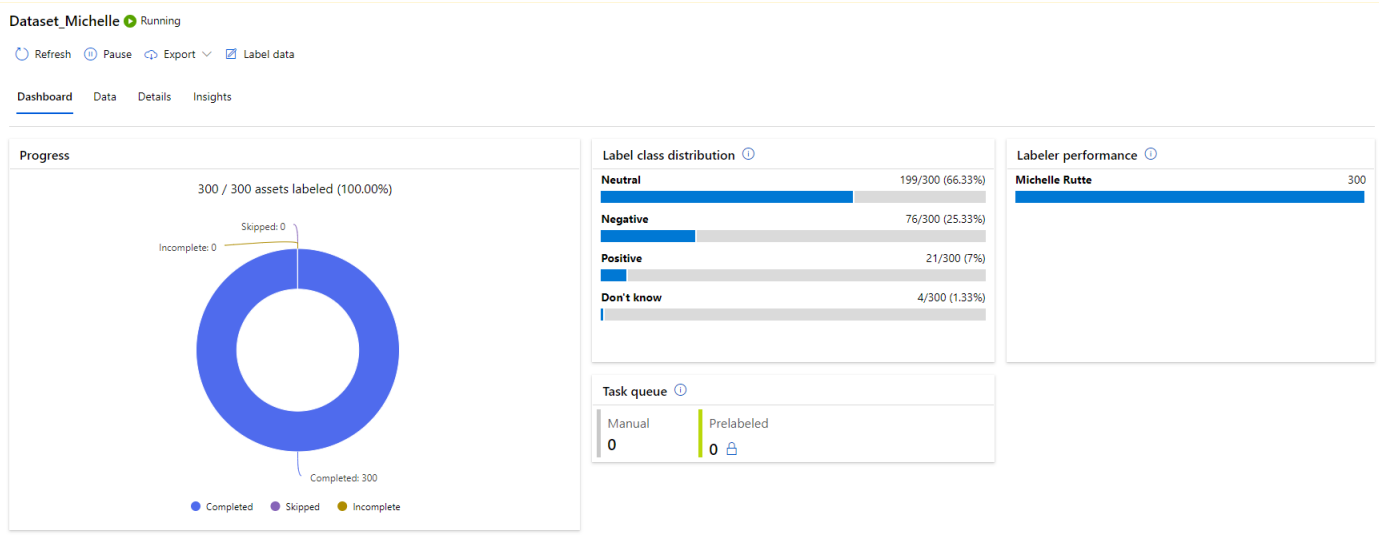


Figure 23: Labeling experiment 2: overview of Michelle's labels.

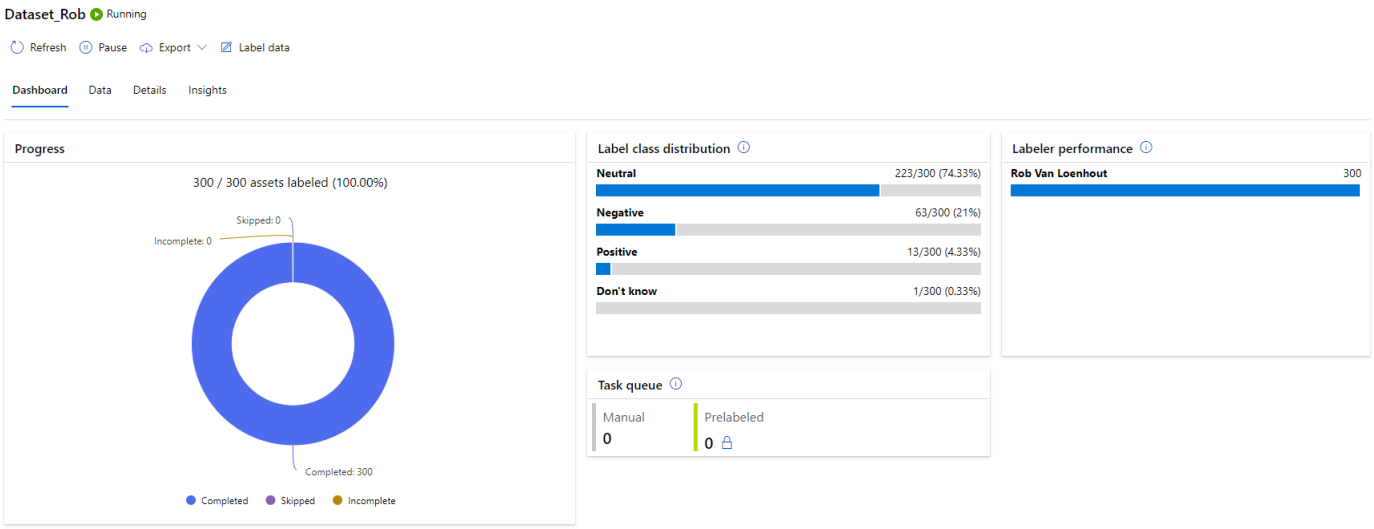


Figure 24: Labeling experiment 2: overview of Rob's labels.

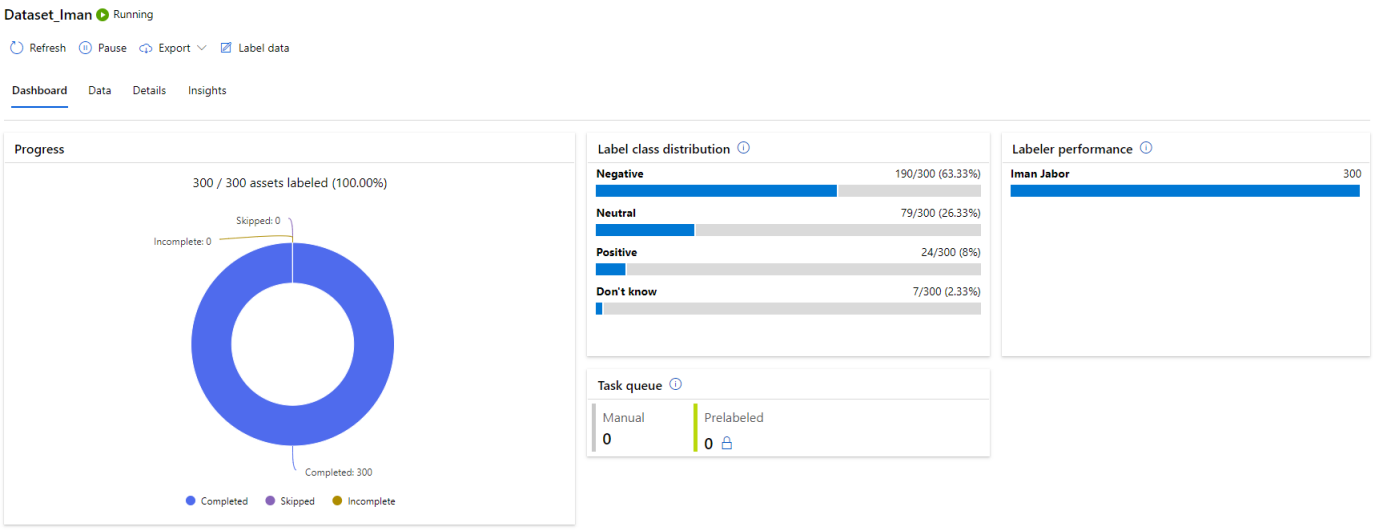


Figure 25: Labeling experiment 2: overview of Iman's labels.

Labeling experiment 3

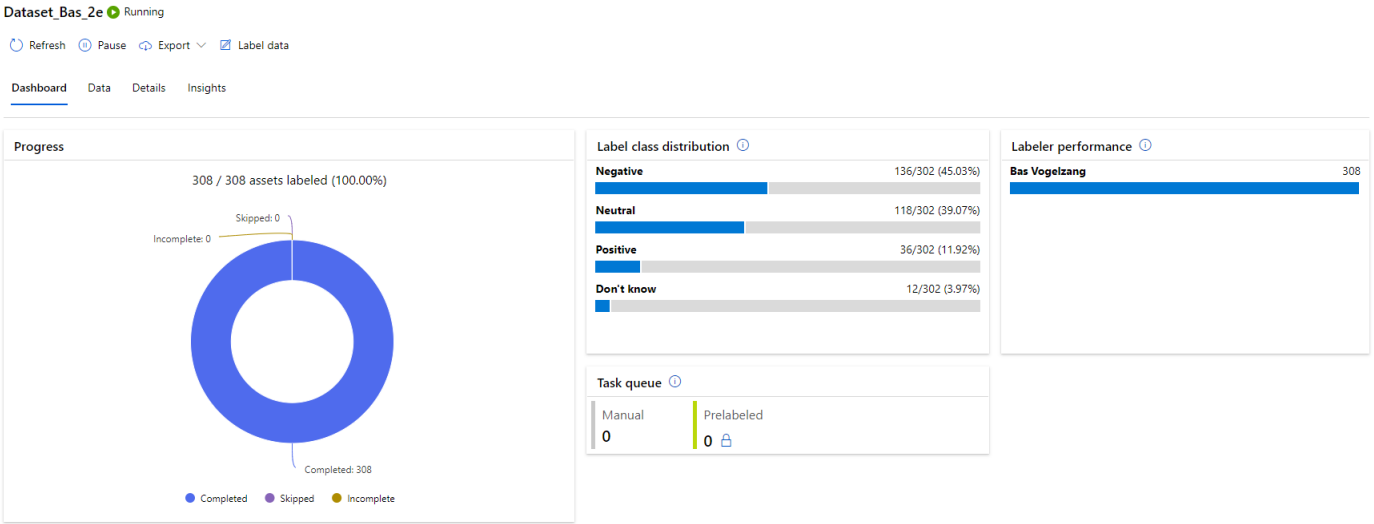


Figure 26: Labeling experiment 3: overview of Bas's labels.

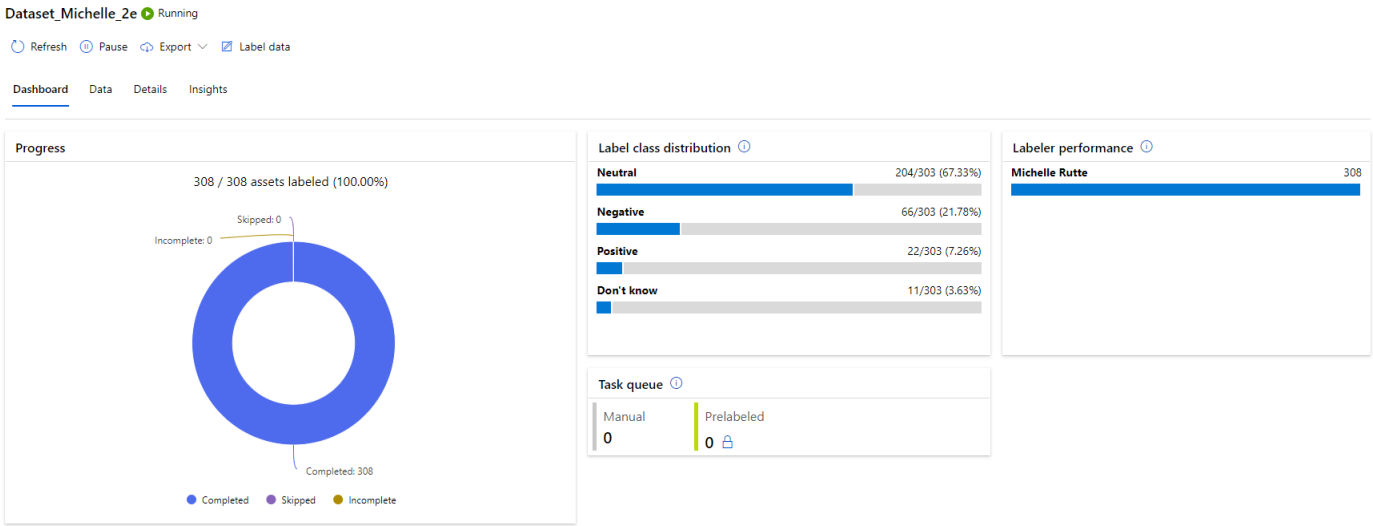


Figure 27: Labeling experiment 3: overview of Michelle's labels.

Jasper classified the data using Michelle's account because he was the external participant.

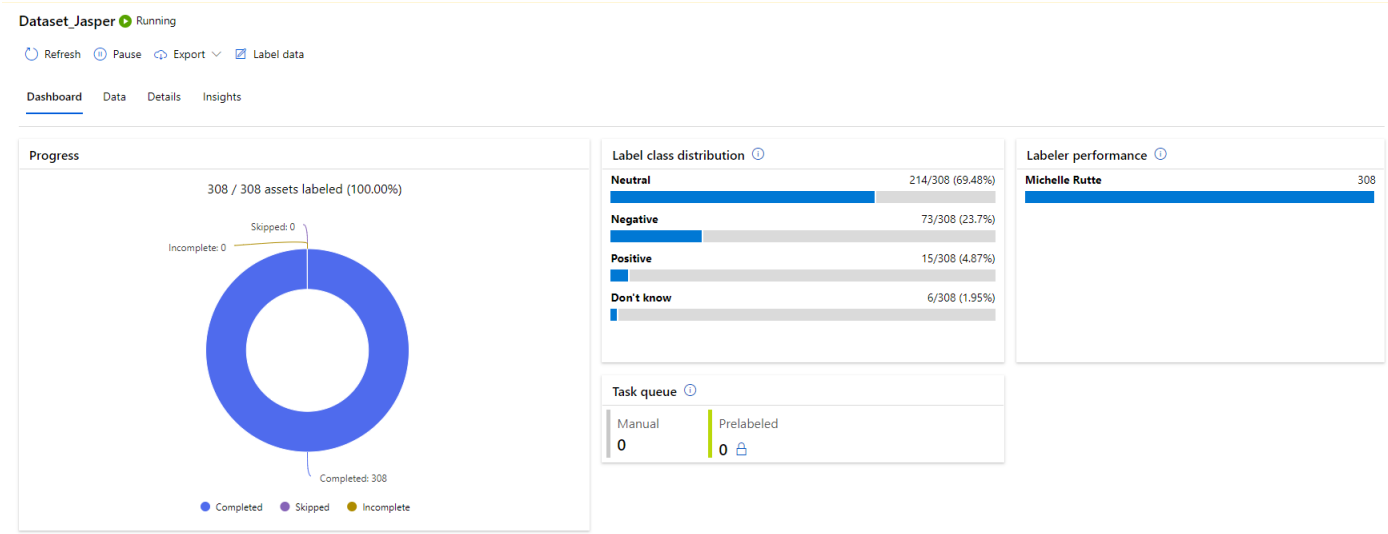


Figure 28: Labeling experiment 3: overview of Jasper's labels.

Appendix B

Appendix B contains the visualizations for the results on the validation set of experiment 1. Additionally, an overview of the results for experiment 1 for the evaluation metrics is provided.

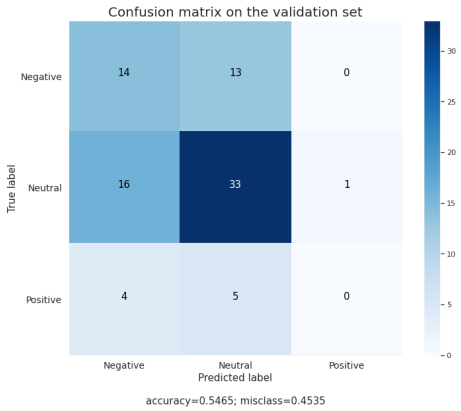
BERTje with batch size 4

Learning rate and weight decay

Figure 29: Learning rate versus the loss per weight decay for BERTje with batch size 4.

Momentum 0.95-0.85

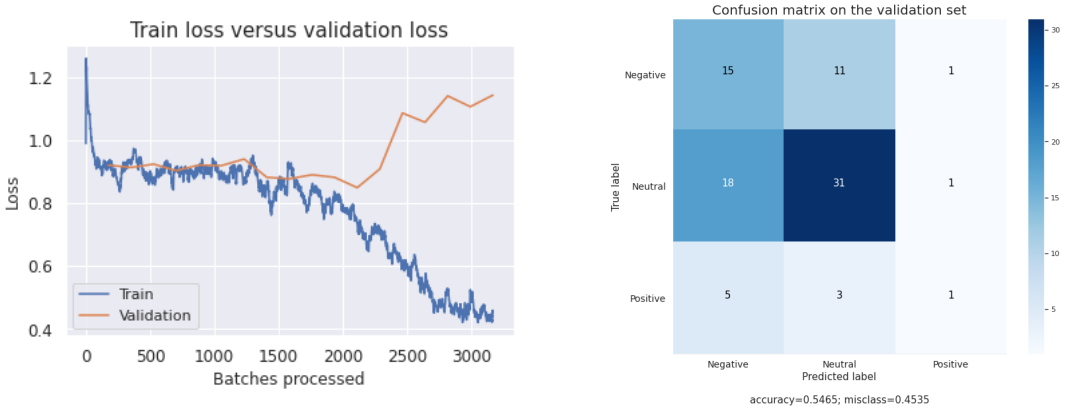
(a) Losses.



(b) Confusion matrix.

Figure 30: Validating results of BERTje with batch size 4 and momentum 0.95-0.85.

Momentum 0.99-0.90



(a) Losses.

(b) Confusion matrix.

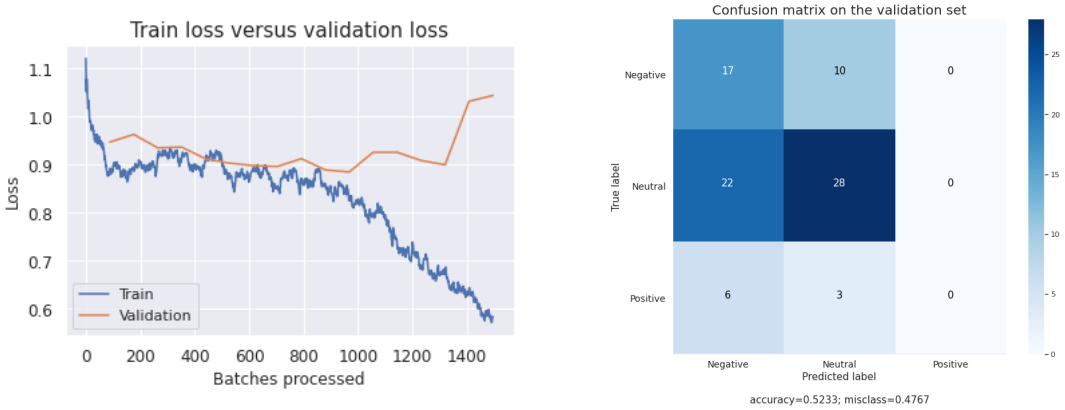
Figure 31: Validating results of BERTje with batch size 4 and momentum 0.99 0.90.

BERTje with batch size 8

Learning rate and weight decay

Figure 32: Learning rate versus the loss per weight decay for BERTje with batch size 8.

Momentum 0.95-0.85

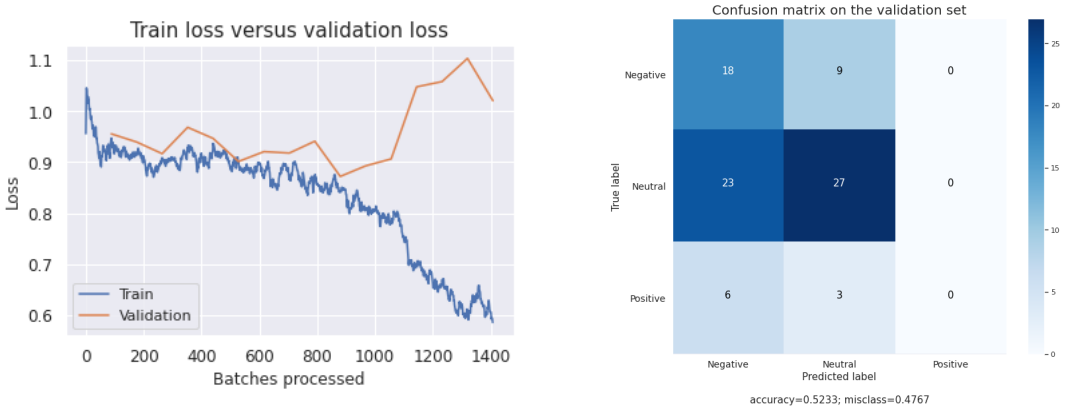


(a) Losses.

(b) Confusion matrix.

Figure 33: Validating results of BERTje with batch size 8 and momentum 0.95-0.85.

Momentum 0.99-0.90



(a) Losses.

(b) Confusion matrix.

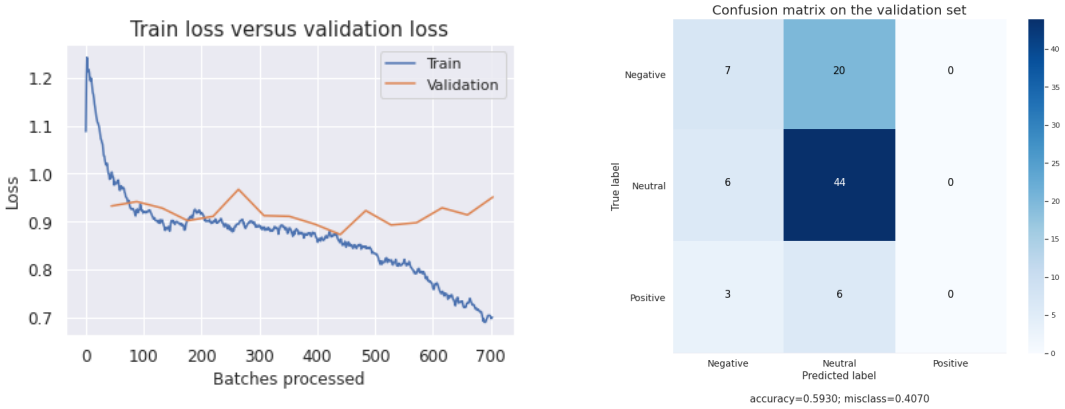
Figure 34: Validating results of BERTje with batch size 8 and momentum 0.99-0.90.

BERTje with batch size 16

Learning rate and weight decay

Figure 35: Learning rate versus the loss per weight decay for BERTje with batch size 16.

Momentum 0.95-0.85

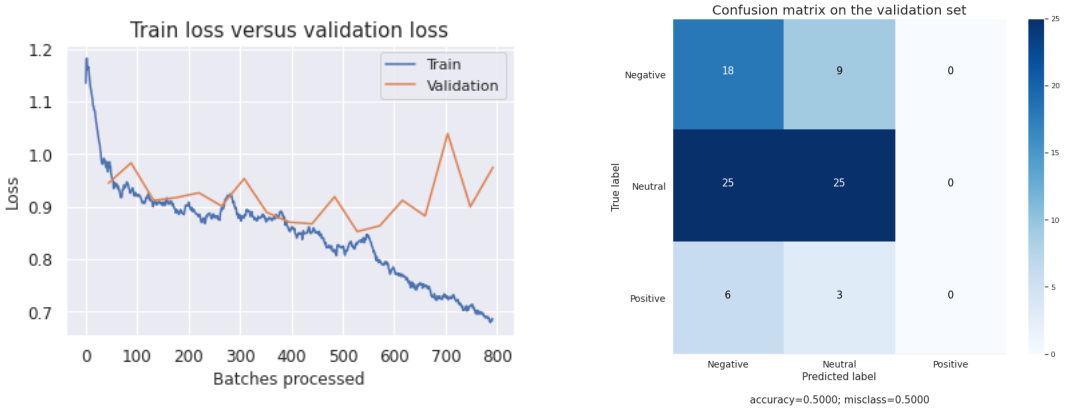


(a) Losses.

(b) Confusion matrix.

Figure 36: Validating results of BERTje with batch size 16 and momentum 0.95-0.85.

Momentum 0.99-0.90



(a) Losses.

(b) Confusion matrix.

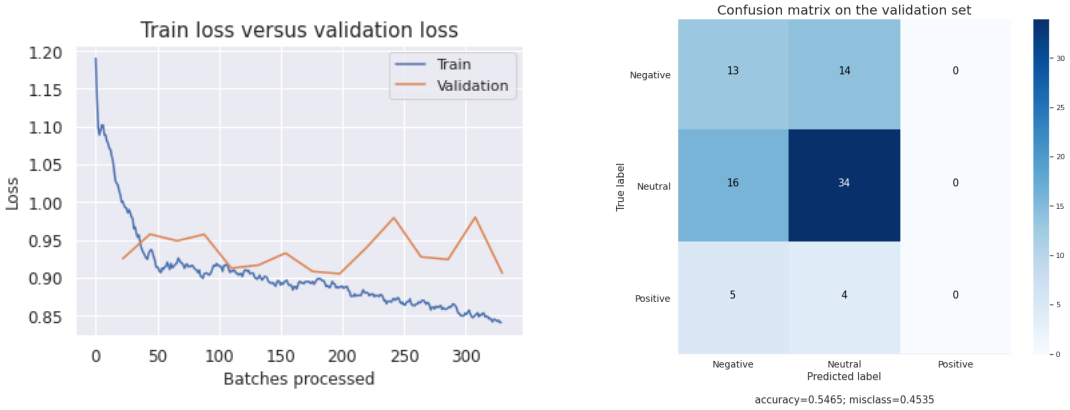
Figure 37: Validating results of BERTje with batch size 16 and momentum 0.99-0.90.

BERTje with batch size 32

Learning rate and weight decay

Figure 38: Learning rate versus the loss per weight decay for BERTje with batch size 32.

Momentum 0.95-0.85

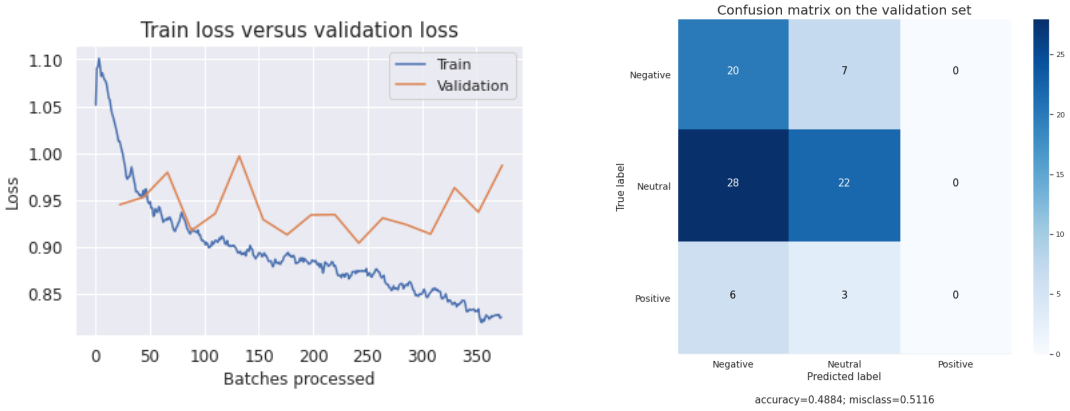


(a) Losses.

(b) Confusion matrix.

Figure 39: Validating results of BERTje with batch size 32 and momentum 0.95-0.85.

Momentum 0.99-0.90



(a) Losses.

(b) Confusion matrix.

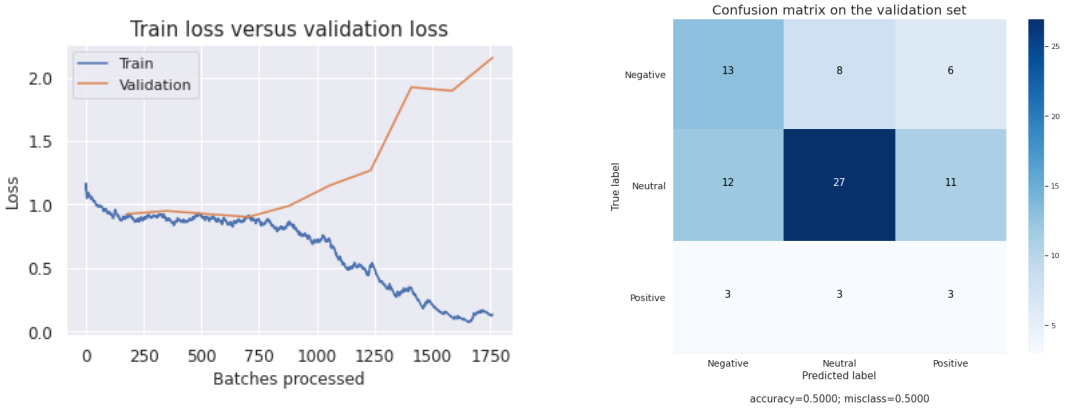
Figure 40: Validating results of BERTje with batch size 32 and momentum 0.99-0.90.

RobBERT with batch size 4

Learning rate and weight decay

Figure 41: Learning rate versus the loss per weight decay for RobBERT with batch size 4.

Momentum 0.95-0.85

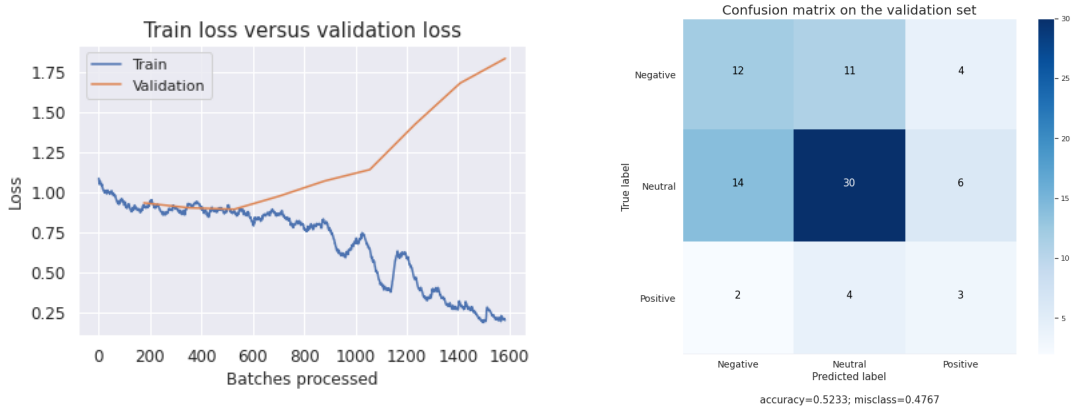


(a) Losses.

(b) Confusion matrix.

Figure 42: Validating results of RobBERT with batch size 4 and momentum 0.95-0.85.

Momentum 0.99-0.90



(a) Losses.

(b) Confusion matrix.

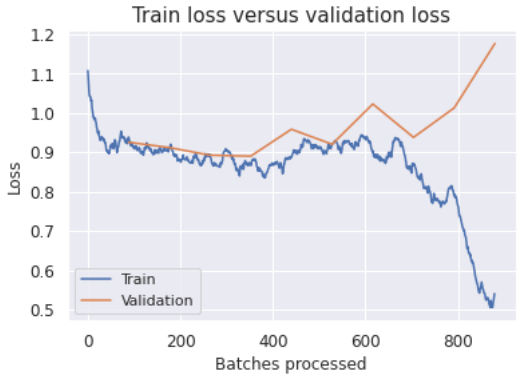
Figure 43: Validating results of RobBERT with batch size 4 and momentum 0.99-0.90.

RobBERT with batch size 8

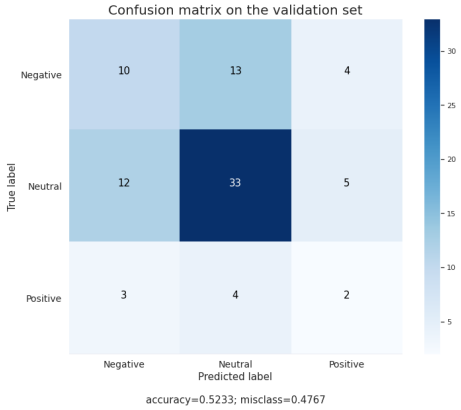
Learning rate and weight decay

Figure 44: Learning rate versus the loss per weight decay for RobBERT with batch size 8.

Momentum 0.95-0.85



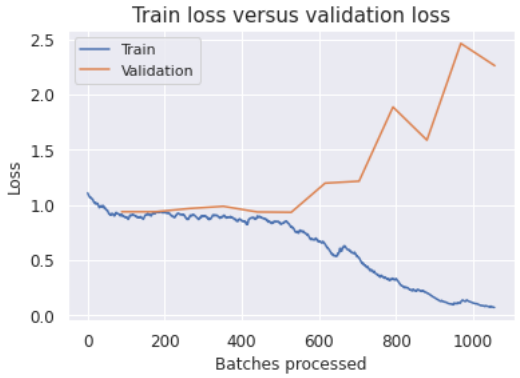
(a) Losses.



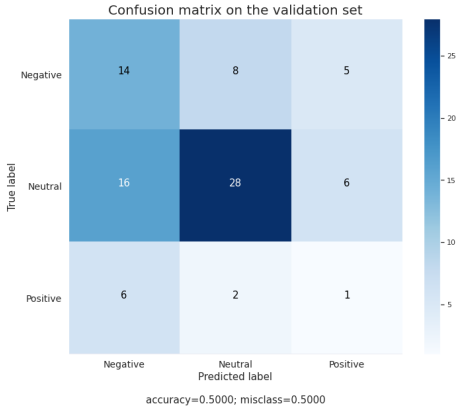
(b) Confusion matrix.

Figure 45: Validating results of RobBERT with batch size 8 and momentum 0.95-0.85.

Momentum 0.99-0.90



(a) Losses.



(b) Confusion matrix.

Figure 46: Validating results of RobBERT with batch size 8 and momentum 0.99-0.90.

RobBERT with batch size 16

Learning rate and weight decay

Figure 47: Learning rate versus the loss per weight decay for RobBERT with batch size 16.

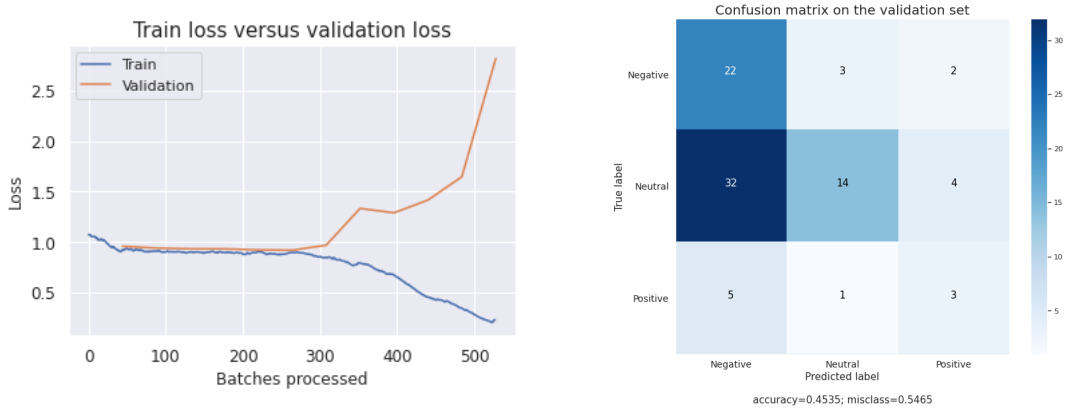
Momentum 0.95-0.85

(a) Losses.

(b) Confusion matrix.

Figure 48: Validating results of RobBERT with batch size 16 and momentum 0.95-0.85.

Momentum 0.99-0.90



(a) Losses.

(b) Confusion matrix.

Figure 49: Validating results of RobBERT with batch size 16 and momentum 0.99-0.90.

RobBERT with batch size 32

Learning rate and weight decay

Figure 50: Learning rate versus the loss per weight decay for RobBERT with batch size 32.

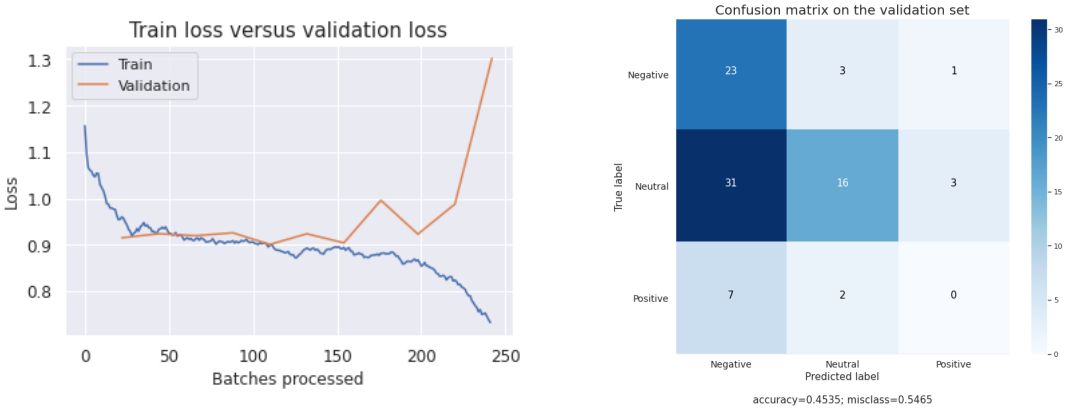
Momentum 0.95-0.85

(a) Losses.

(b) Confusion matrix.

Figure 51: Validating results of RobBERT with batch size 32 and momentum 0.95-0.85.

Momentum 0.99-0.90



(a) Losses.

(b) Confusion matrix.

Figure 52: Validating results of RobBERT with batch size 32 and momentum 0.99-0.90.

Overview results evaluation metrics experiment 1

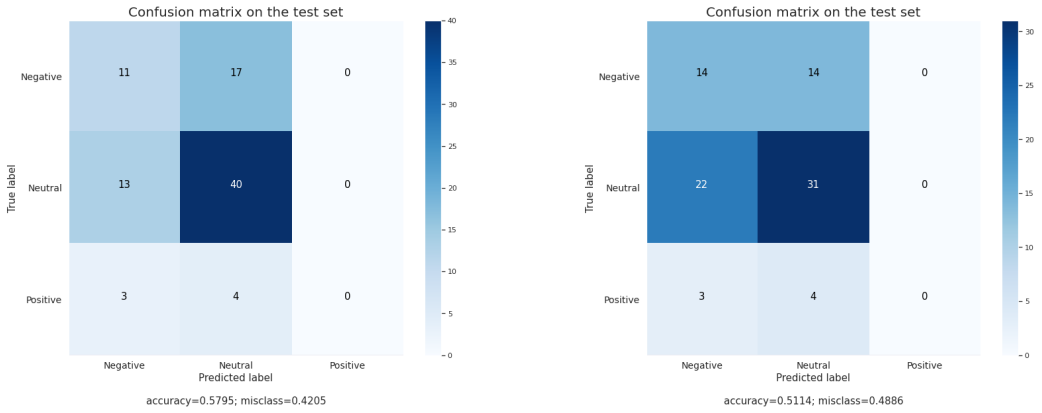
Table 21: Overview of the results per model on the validation set in experiment 1. BS = batch size, LR = learning rate, WD = weight decay, Moms = Momentum, Acc = accuracy, F1 = weighted average F1-score, P = weighted average precision, R = weighted average recall.

	Hyper parameters					Metrics			
Model	**BS**	**LR**	**WD**	**Moms**	**Epochs**	**Acc**	**F1**	**P**	**R**
BERTje	4	8×10^{-6}	10^{-4}	0.95-0.85	15	0.55	0.52	0.51	0.55
BERTje	4	8×10^{-6}	10^{-4}	0.99-0.90	11	0.55	0.54	0.56	0.55
BERTje	8	10^{-5}	10^{-2}	0.95-0.85	16	0.52	0.51	0.52	0.52
BERTje	8	10^{-5}	10^{-2}	0.99-0.90	15	0.52	0.51	0.52	0.52
BERTje	16	9×10^{-6}	10^{-6}	0.95-0.85	15	0.59	0.53	0.50	0.59
BERTje	16	9×10^{-6}	10^{-6}	0.99-0.90	17	0.50	0.48	0.51	0.50
BERTje	32	6×10^{-6}	10^{-2}	0.95-0.85	14	0.55	0.52	0.50	0.55
BERTje	32	6×10^{-6}	10^{-2}	0.99-0.90	16	0.49	0.47	0.52	0.49
RobBERT	4	4×10^{-5}	10^{-2}	0.95-0.85	9	0.50	0.53	0.57	0.55
RobBERT	4	4×10^{-5}	10^{-2}	0.99-0.90	7	0.52	0.53	0.55	0.52
RobBERT	8	5×10^{-5}	10^{-2}	0.95-0.85	9	0.52	0.53	0.53	0.52
RobBERT	8	5×10^{-5}	10^{-2}	0.99-0.90	7	0.50	0.52	0.56	0.50
RobBERT	16	8×10^{-5}	10^{-4}	0.95-0.85	7	0.50	0.48	0.48	0.50
RobBERT	16	8×10^{-5}	10^{-4}	0.99-0.90	11	0.45	0.43	0.60	0.45
RobBERT	32	10^{-4}	10^{-4}	0.95-0.85	11	0.48	0.49	0.51	0.48
RobBERT	32	10^{-4}	10^{-4}	0.99-0.90	10	0.45	0.43	0.56	0.45

Appendix C

Appendix C contains the confusion matrices on the test set for experiment 3.

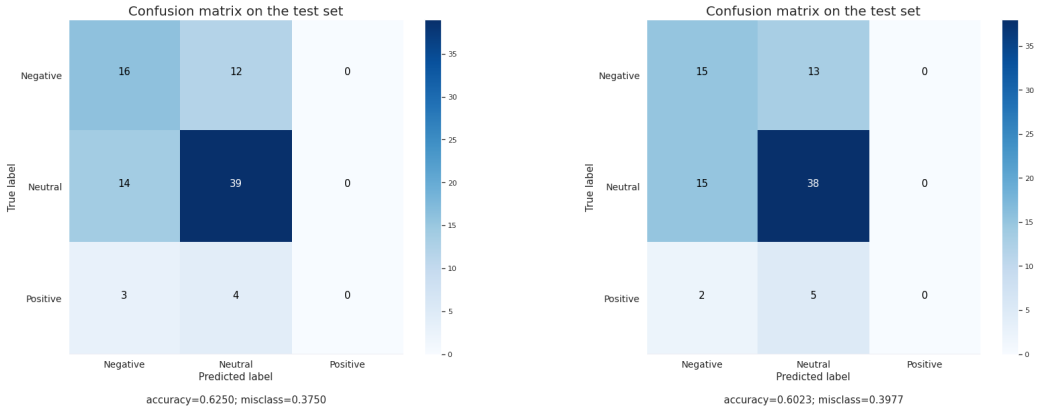
Training on 300 samples



(a) BERTje. (b) RobBERT.

Figure 53: Test set results of the best performing models of BERTje and RobBERT (trained on 300 samples).

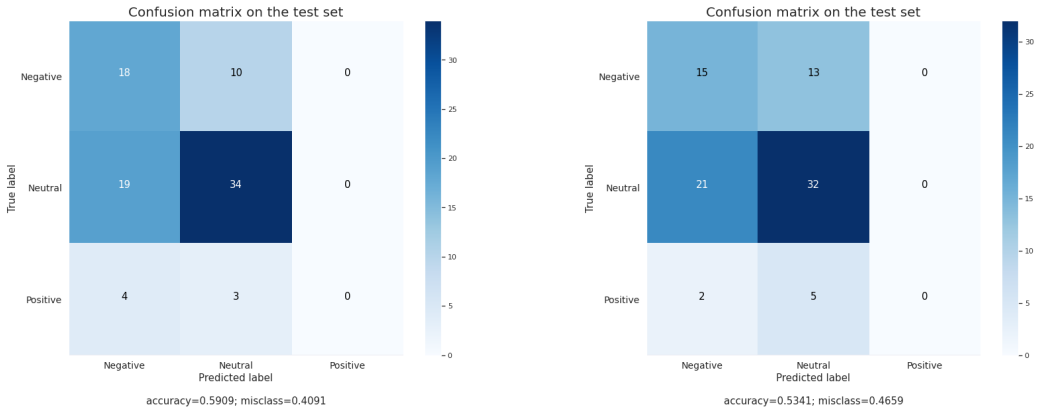
Training on 600 samples



(a) BERTje. (b) RobBERT.

Figure 54: Test set results of the best performing models of BERTje and RobBERT (trained on 600 samples).

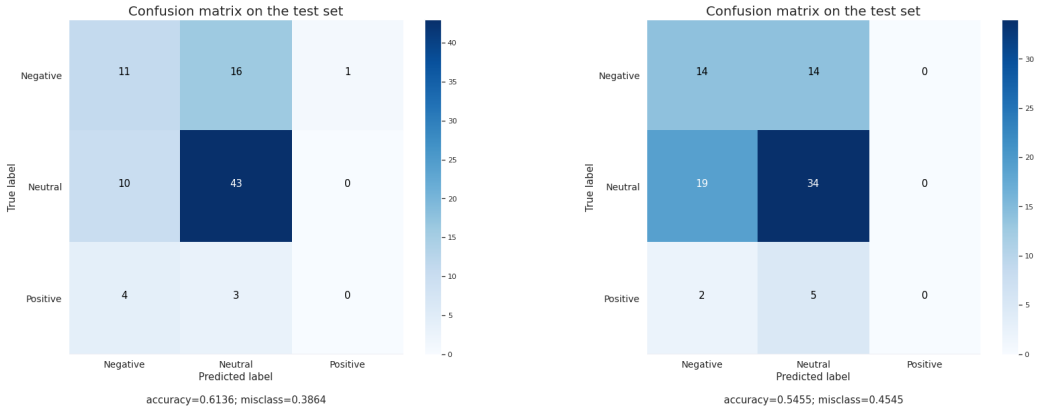
Downsampling neutrals



(a) BERTje. (b) RobBERT.

Figure 55: Test set results of the best performing models of BERTje and RobBERT (downsampling neutrals).

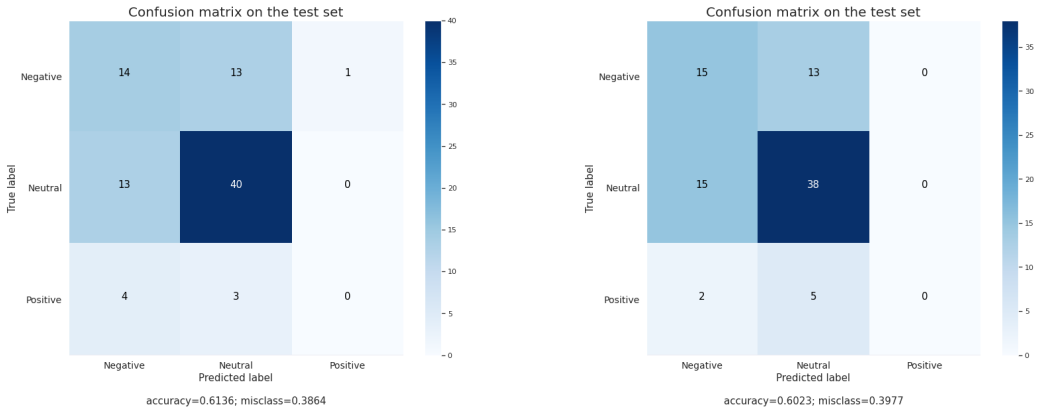
Upsampling positives



(a) BERTje. (b) RobBERT.

Figure 56: Test set results of the best performing models of BERTje and RobBERT (upsampling positives).

Combined: downsampling neutrals and upsampling positives



(a) BERTje.

(b) RobBERT.

Figure 57: Test set results of the best performing models of BERTje and RobBERT (combined sample techniques).