PhD-FHSE-2022-009
The Faculty of Humanities, Education and Social Sciences

# DISSERTATION

Defence held on 25/02/2022 in Esch-sur-Alzette

to obtain the degree of

## DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

## EN *SCIENCES DU LANGAGE*

by

Daniela GIERSCHEK
Born on 28 October 1990 in Wiesbaden, (Germany)

# DETECTION OF SENTIMENT IN LUXEMBOURGISH USER COMMENTS

## Dissertation defence committee

Dr Peter Gilles, dissertation supervisor
*Professor, Université du Luxembourg*

Dr Christoph Purschke, Chairman
*Professor, Université du Luxembourg*

Dr Barbara Plank
*Professor, IT University Copenhagen*

Dr Petra Kralj Novak
*Professor, Central European University/Jožef Stefan Institute, Ljubljana*

Dr Christoph Schommer
*Professor, Université du Luxembourg*

**Abstract**

Sentiment is all around us in everyday life. It can be found in blog posts, social media comments, text messages and many other places where people express themselves. Sentiment analysis is the task of automatically detecting those sentiments, attitudes or opinions in written text. In this research, the first sentiment analysis solution for the low-resource language, Luxembourgish, is conducted using a large corpus of user comments published on the RTL Luxembourg website `www.rtl.lu`. Various resources were created for this purpose to set the foundation for further sentiment research in Luxembourgish.

A Luxembourgish sentiment lexicon and an annotation tool were built as external resources that can be used for collecting and enlarging training data for sentiment analysis tasks. Additionally, a corpus of mainly sentences of user comments was annotated with *negative, neutral* and *positive* labels. This corpus was furthermore automatically translated to English and German.

Afterwards, diverse text representations such as *word2vec*, *tf-idf* and *one-hot encoding* were used on the three versions of the corpus of labeled sentences for training different machine learning models. Furthermore, one part of the experimental setup leveraged linguistic features for the classification process in order to study their impact on sentiment expressions.

By following such a broad strategy, this thesis not only sets the basis for sentiment analysis with Luxembourgish texts but also intends to give recommendations for conducting sentiment detection research for other low-resource languages. It is demonstrated that creating new resources for a low-resource language is an intensive task and should be carefully planned in order to outperform working with translations where the target language is a high-resource language such as English and German.

# Acknowledgements

Vor etwas mehr als zehn Jahren, 2010, kam ich das erste Mal für ein Jahr nach Luxemburg - eigentlich, um im Rahmen meines europäischen Freiwilligendienstes Französisch zu lernen. Groß war die Frustration als es dann „nur“ Luxemburgisch wurde! Nichtsahnend war ich damals, welchen großen Stellenwert diese Sprache einmal in meinem Leben einnehmen würde. Ein erster Dank geht deshalb an alle, die mich in diesem Jahr begleitet und mir mit viel Begeisterung ihre Sprache beigebracht haben. Ohne euch würde es diese Doktorarbeit heute nicht geben.

Ein großer Dank geht außerdem an meinen Doktorvater, Prof. Dr. Peter Gilles. Merci Peter für die Begleitung, Unterstützung und die intensiven Diskussionen über die letzten Jahre. Danke für die Möglichkeit der Doktorarbeit und dafür, dass du immer an mich geglaubt hast. Es war mir eine große Freude, in deinem Team promovieren zu dürfen!

Des Weiteren möchte ich Prof. Dr. Christoph Purschke und Prof. Dr. Christoph Schommer für die Begleitung meiner Promotion sowohl im STRIPS Projekt als auch in meinem CET danken. Ich durfte viel durch unsere regelmäßigen Diskussionen und Feedbackgespräche lernen und so als Wissenschaftlerin wachsen. Auch möchte ich in diesem Rahmen RTL Luxembourg nennen, deren Texte von `www.rtl.lu` ich für meine Doktorarbeit uneingeschränkt nutzen durfte.

Diese Doktorarbeit wäre außerdem nicht möglich gewesen ohne meinen Projektkollegen Dr. Joshgun Sirajzade. Danke Josh für die enge Zusammenarbeit, in der ich so viel über das Programmieren und die Arbeit mit Textkorpora lernen durfte und das Korrekturlesen. Du hast mich auch in den schwierigen Phasen der Promotion immer ermutigt.

Ein großer Dank geht außerdem an Dr. Nathalie Entringer, Dr. Siwen Guo, Dr. Sara Martin, Dr. Sam Mersch, Omar Ramírez Sánchez und Gabriel Alejandro Rivera Cosme für das Korrekturlesen, Feedbackgeben zu meiner Arbeit und die mentale Unterstützung während dieser Reise.

Zu guter Letzt möchte ich meiner Familie, meinen Freunden und meinem Partner danken, die mich in dieser intensiven Zeit begleitet haben und Ermutigung sowie Ablenkung in den schwierigen Momenten gegeben haben.

# Contents

# List of Figures

# List of Tables

# Nomenclature

**ANN** Artificial Neural Network

**ADJ** Adjective

**ADV** Adverb

**CL** Computational Linguistics

**DL** Deep Learning

**DT** Decision Tree

**GUI** Graphical User Interface

**IAA** Inter-Annotator Agreement

**IR** Information Retrieval

**KNN** K Nearest Neighbor

**LRM** Logistic Regression

**ML** Machine Learning

**MLM** Masked Language Model

**N** Noun

**NLP** Natural Language Processing

**NMT** Neural Machine Translation

**NSP** Next Sentence Prediction

**POS** Part-of-Speech

**RF** Random Forest

**SVM** Support Vector Machine

**TF-IDF** Term Frequency–Inverse Document Frequency

**V** Verb

Chapter 1

# Creating Sentiment Analysis Resources for Luxembourgish

People communicate with each other on a daily basis exchanging their ideas, plans, feelings, emotions and also sentiments. Sentiment thus is a crucial part of human expression and interaction. Its study has a long standing tradition in different scientific fields like psychology, linguistics and computer science using varying definitions of sentiment and related terms such as emotion, affect and feeling. More precisely, sentiment can be found in different digital platforms, for instance in social media, blog posts, reviews of products or movies as well as in user comments. Those wide-spread application fields are particularly interesting for Natural Language Processing (NLP) and Artificial Intelligence. While resources for languages with a large amount of speakers, such as English, already exist and research on those has been going on for roughly 20 years (some early NLP papers on sentiment analysis include Turney (2002) and Pang et al. (2002)), smaller languages have not been extensively studied in comparison. With the rise of the internet and social media, writing and exchanging ideas in real time has become convenient and fast leading to a large textual basis in more and more languages.

In this thesis, I focus on sentiment analysis for the low-resource language Luxembourgish. Luxembourgish is a language mainly spoken in the Grand Duchy of Luxembourg. It is crucial for the national identity and was declared the national language of the country in 1984 (Gilles, in press, p. 5). The importance of communicating in Luxembourgish and visibility of the language in general for the people can regularly be seen in different aspects of life such as the local media coverage. For instance, stopping to use Luxembourgish for airport announcements at the country's only airport made it to the national news (RTL Luxembourg (2021)). Despite this importance of Luxembourgish for the country and its speakers, NLP research is still scarce [†]. In the case of sentiment analysis, no prior research has been done making this thesis the first of its kind and thus laying the foundation in this field for all future research. In order to carry out sentiment analysis for Luxembourgish, I obtained a large corpus of news articles and user comments published

[†]Some tools that were specifically created for the Luxembourgish language can be found in Sirajzade and Schommer (2019), Sirajzade et al. (2020a) and Purschke (2020b).

between 1999 and 2018 on RTL Luxembourg's website `www.rtl.lu`. RTL Luxembourg is one of the biggest news providers in Luxembourg, providing a large quantity of their news in Luxembourgish. A recent study published by the market research institutes TNS Ilres and Kantar Belgium (TNS Ilres and Kantar Belgium (2021)) stresses this point by showing that RTL Luxembourg is the top news provider in Luxembourg with 39.9% of the population of the country aged 15 or more accessing the website `www.rtl.lu` on a daily basis. Besides simply delivering international and national news, they allow the general public to create a profile on their website to subsequently comment on the articles published there. This leads to a collection of diverse texts denoted as "RTL corpus" in this thesis. This corpus is particularly interesting due to the amount of different authors expressing their sentiments towards a lot of different topics and sometimes also entering into vivid discussions among each other.

This thesis intends to set the groundwork for future research in Luxembourgish sentiment analysis, hoping that other researchers will be able to benefit from those findings in order to move even further towards a well-functioning sentiment analysis system for this low-resource language. Additionally and most importantly, I aspire to answer the following research questions:

*- How can sentiment analysis be carried out in a low-resource setting? What are the challenges related to this task and how can they (partially) be overcome?*

As no sentiment resources for Luxembourgish existed yet, the first important step of this research was to carefully plan and create those. For a human reader, the RTL corpus obtained from RTL Luxembourg is full of expressions of sentiment towards a large variety of topics and aspects. However, it was provided without annotations that are essential for building (automatic) sentiment detection systems. Therefore, the corpus was enriched with sentiment labels which could either be *negative, neutral* or *positive* (see chapter 3.6). This was pursued with a multilayered approach combining annotation by a trained linguist with crowdsourcing as well as automatic and rule-based techniques. To be able to answer this research question, the next step consisted of automatically translating this annotated RTL corpus instances to English and German. That way, it could be studied whether using translated versions where the language is better resourced than Luxembourgish can be beneficial for sentiment detection. This was particularly interesting to examine whether sentiment analysis in a low-resource setting indeed requires their own custom-built resources, or if it is sufficient to leave all cultural and linguistic concerns aside and use automatic translations of already existing and tested resources for a different language. The reason for this particular interest is that building resources is a labor- and time-intensive process and should therefore ideally lead to satisfactory results.

*- What are the possibilities of evaluating the performance of different classification systems on the annotated corpus of this work? Are those evaluation measures sufficient to adequately determine the quality of the performance?*

The annotations with *negative, neutral* and *positive* labels on the RTL corpus were the essential target output for training a selection of machine learning (ML) algorithms using different vectorization techniques (see chapters 2.6.2, 2.6.3, 2.6.4 and 2.6.5). After training,

the quality of the performance of the different classifiers was evaluated on a small portion of previously unseen data from the RTL corpus. For this purpose, the $F_1$ score metric (see chapter 2.8) was chosen, an evaluation metric frequently applied to classification problems. Despite being a standard approach for evaluation, its applicability for the RTL corpus needed to be evaluated. As stated above, most of the annotations used as training output were automatically collected. Therefore, it is difficult to verify whether the algorithms were actually trained on well-annotated training data or if potential noise already had a significant influence on the learning process itself. Additionally, the question of whether or not to classify the instances of language into three boxes, i.e. *negative, neutral* and *positive*, is a good and sufficient representation of human expressions will be critically discussed in chapter 2.1 and studied later on the actual data in practice.

*- Does adding Part-of-Speech as additional linguistic features help to improve the automatic detection of sentiment?*

The RTL corpus used for this thesis was not only enriched with a multilayered process of sentiment annotations, but also with additional linguistic information. A Part-of-speech (POS) tagger (Sirajzade and Schommer (2019)) was used to provide POS information for every word in the corpus. As previous research such as Hatzivassiloglou and McKeown (1997), Farzindar and Inkpen (2015) and Taboada (2016) have shown, specific POS such as adjectives can carry important sentiment information of a sentence which makes them especially valuable for sentiment analysis. To study their impact on Luxembourgish sentiment detection, I added them as additional features for a part of the ML experiments carried out for this research.

Last but not least, this thesis also aims to give recommendations as to how sentiment analysis in low-resource settings can be best performed by carefully examining the work implemented and abstracting those findings from Luxembourgish to the general low-resource case. In order to approach towards this point, the following chapter will provide the theoretical background required for building resources as well as carrying out practical ML experiments on the RTL corpus.

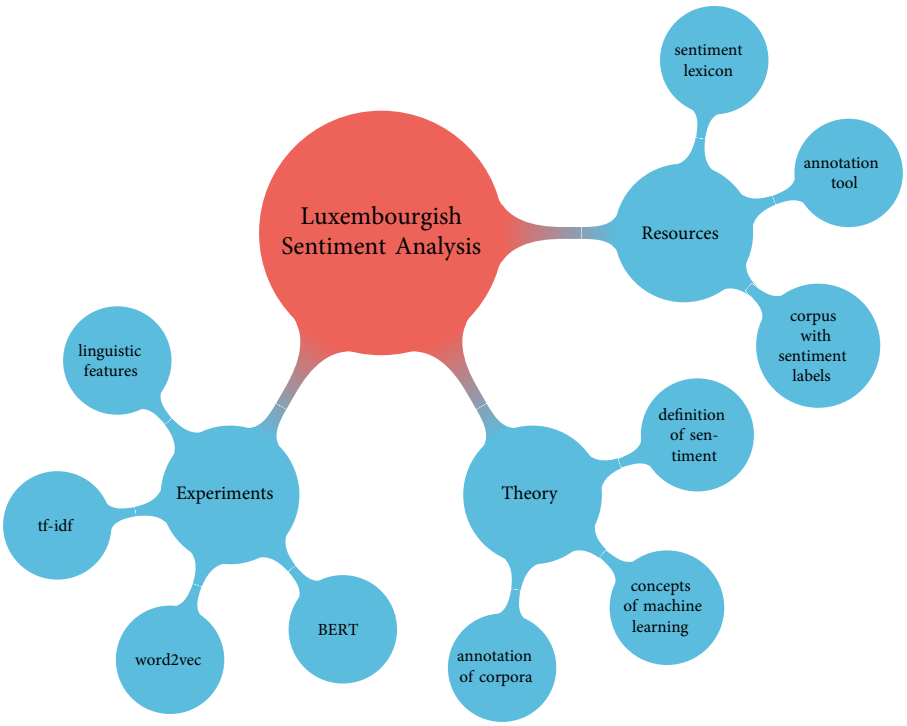



Figure 1.1 – Overview of the work of this thesis.

This work is based on several pillars (see figure 1.1): The theoretical embedding of this research, the creation of resources for Luxembourgish and the carrying out of experiments to automatically detect sentiment patterns. Those parts are presented and elaborated in five chapters: The beginning is an introduction to the thesis and its research domain as well as a statement of relevance (see chapter 1). Next, chapter 2 discusses the term sentiment with the challenges related to it and gives a working definition of the term for this thesis. Furthermore, the theoretical background of annotation, ML algorithms and their evaluation as well as an introduction to important aspects of the Luxembourgish language is given. Those factors are studied from a practical point of view in chapters 3 and 4 which are about the implementation carried out as part of this research. The different resources created and required for sentiment analysis with Luxembourgish are shown (chapter 3) and several ML experiments are conducted and carefully evaluated. Chapter 4 concludes with recommendations on how sentiment detection in low-resource settings should be best implemented. This follows the assumption that the challenges related to this work and the Luxembourgish background can also be transferred to other low-resource languages, as they are not always specific to Luxembourgish but rather to the low-resource background. This thesis concludes with chapter 5. In that chapter, I recap the work done and portray potential options for future work.

Chapter 2

# Background and State of the Art

The purpose of this chapter is to describe the methodology used and the literature reviewed which both serve as theoretical backbone. Due to the interdisciplinary manner of the thesis, a variety of topics both from Humanities as well as Computer Science are covered. This description is necessary to enable meaning detection in the RTL corpus which is the textual basis of this work. This corpus, which will be shown in greater detail in chapter 3.1, consists of 179,283 news articles and 585,358 user comments published on the website *www.rtl.lu* between 1999 and 2018. The focus of this thesis lies on the user comments part of the RTL corpus. They cover a wide range of topics and authors and are mostly written in Luxembourgish.

First, the main concept of this work sentiment is explained and embedded in its occurrence in different research areas. This also includes a discussion of challenges that can appear in sentiment analysis. Second, an introduction to the different computational approaches and algorithms used for carrying out sentiment analysis is given. The third part of this chapter describes different ways of how a corpus can be annotated and prepared for a computational analysis. The chapter ends with a portrayal of important aspects of the Luxembourgish language.

## 2.1 Definition of Sentiment

This work is set in the field of sentiment analysis, a prominent and promising research area of the NLP community. What is this sentiment that needs to be analyzed? And why is it such an important field? In this section, I will first of all intend to grasp a definition of sentiment and related concepts. Next, I will shed light on which disciplines use sentiment and affiliated ideas for such work. Additionally, the difficulties and limitations of the presented approaches will be pointed out.

Sentiment is a concept that is everywhere around us and is part of our daily life. For instance, sentiment can be found on social media, internet shopping platforms and websites for movie reviews. It has to be differentiated from the notion of *feeling*.

Liu (2015) explains:

> "Sentiment analysis, also called *opinion mining*, is the field of study that analyzes people's opinions, sentiments, appraisals, attitudes and emotions toward entities and their attributes expressed in written text. The entities can be products, services, organizations, individuals, events, issues, or topics (Liu, 2015, p. 1)."

Sentiment, according to Liu (2015), is thus the examination of what people think and believe towards a certain entity, i.e. dealing with personal feelings and opinions. It is also important to note that the task of sentiment analysis is used synonymously with opinion mining by many researchers. This definition by Liu (2015) is certainly a first good approach to understand sentiment analysis. However, it uses related concepts such as opinion, appraisal, attitude and emotion to explain what sentiment analysis is supposed to mean. Those concepts are, nonetheless, not further explained, assuming that they are intuitively understandable by everybody. This is questionable, as every person comes with their own personal prior experiences and might understand concepts such as emotions and attitudes differently. I therefore need to dig deeper and examine not only sentiment in NLP but also those related concepts in greater detail to understand the meaning of sentiment. This is crucial, as those related terms are often used interchangeably in NLP research which leads to confusion (Munezero et al., 2014, p. 101) and also unclear guidelines of how to proceed in data preparation and processing.

Taboada (Taboada, 2016, p. 326) approaches sentiment analysis from a slightly different angle, and consequently adds a different layer to the illustration of what sentiment is by stating that the goal of sentiment analysis is to examine if a text is subjective or objective. If it falls into the subjective category, the aim is to determine "the valence or polarity of a piece of text" (Mohammad, 2017, p. 6) to find out whether a *positive* or a *negative* opinion is expressed (Taboada, 2016, p. 326). Therefore, the following questions need to be raised: What exactly is meant by *positive, negative* and also *neutral*? How adequate is such a scale to comprehend the different aspects of sentiment? And how is sentiment different from related concepts like attitudes, feelings and emotions? In order to better grasp the concept of sentiment and *negative, neutral* and *positive*, I will first of all examine related ideas before coming back to a more detailed definition of how sentiment is defined for this work.

### 2.1.1 Attitudes and Emotions in Psychology

Concepts related to sentiment, such as feelings, emotions and attitudes, are studied in several research domains besides sentiment analysis. They are found in social psychology, positive psychology, sociolinguistics and discourse analysis, among others. It should not be surprising that psychology scholars study emotions and related concepts like attitude. What is surprising, however, is that neither emotion nor *positive* and *negative* are clearly defined assuming that all those concepts are somehow intuitively understandable. Finding a clear definition of those notions is certainly hard, as the boundaries between emotion,

negative and positive are to some extend fluent and can depend on various aspects like, for instance, cultural contact. Mulligan and Scherer (Mulligan and Scherer, 2012, p. 345) write that "there is no commonly agreed-upon definition of emotion in any of the disciplines that study this phenomenon". Nevertheless, scholars in psychology study *positive* and *negative* emotions. Baumeister et al. (Baumeister et al., 2001, p. 331) state that the differentiation between good and bad emotions is familiar to everyone in psychology and that it is possible to "compare positive and negative emotional states against neutral ones".

Furthermore, the research area of positive psychology, a strand of psychology, studies particularly *positive* emotions (e.g. Fredrickson (1998), Oishi and Kurtz (2011), Seligman and Csikszentmihalyi (2000)). Seligman and Csikszentmihalyi (Seligman and Csikszentmihalyi, 2000, p. 5) write that positive psychology, in contrast to traditional psychology, focuses not on curing people, but on the "positive qualities" of the human being. It works on three levels: the subjective, the individual and the group level. The subjective level is about subjective experiences such as happiness and well-being, the individual level about personal skills like forgiveness and courage and the group level about civic virtues like tolerance and work ethics that contribute to a better society (Seligman and Csikszentmihalyi, 2000, p. 5). Interestingly, the term *positive* is not necessarily defined and neither is the contrasting expression *negative* emotions, despite the fact that the work is centered around *positive* emotions. This is surprising, as it would have been expected for such central concepts to be defined in a very detailed way to prevent potential ambiguity. Fredrickson (Fredrickson, 1998, p. 5), for instance, provides anger, fear and disgust as examples for *negative* and joy, contentment and interest for *positive* emotions. However, no further explanation is given suggesting that the difference between *positive* and *negative* might indeed be intuitively understandable for psychology scholars. This is problematic, because every person has a slightly different understanding of emotions and also of what a *positive* and a *negative* emotion is depending on previous life experiences. Working with examples such as *negative* emotion = fear and *positive* emotion = joy (Fredrickson, 1998, p. 5) can give a first intuition, but is not sufficient to deeply understand what *emotion*, *negative* and *positive* is really all about.

Psychology scholars do not only study (*positive*) emotions, but also similar notions like *attitude*. *Attitude* is one of the most important concepts studied in the field of social psychology ((Graumann, 2005, p. 866) and (Bohner and Dickel, 2011, p. 410)). Bohner and Dickel (Bohner and Dickel, 2011, p. 392) state that the research community agrees that *attitude* is "an evaluation of an object of thought". Those thoughts can be about anything a person can possibly think of. They additionally influence the behavior of a person and their way of processing information (Bohner and Dickel, 2011, p. 407). However, there is no absolute consent in the research community as to whether *attitudes* should be considered as stable or as temporary concepts. A good overview of the different definitions of the term from a psychological point of view can be found in Bohner and Dickel (Bohner and Dickel, 2011, p. 393). Bohner and Dickel show how diverse the subtleties in the different definitions are and how difficult it is thus to find a universally applicable way of understanding the term *attitude*. To summarize, it was observed in this chapter that psychology as well as some of its subfields study *attitudes* and *positive/negative* emotions. Even so, those last

concepts are not always clearly defined. This is remarkable, as it leads to ambiguous cases that could have been prevented to some extend by providing clear and well-structured definitions.

### 2.1.2 Attitudes in Linguistics

*Attitudes* have not only been studied in the field of social psychology, but also in sociolinguistics (e.g. Purschke (2014), Purschke (2015), Huesmann (2017) and Vandermeeren (2005)). Most sociolinguists see *attitudes* from a mentalist perspective meaning that they are "mediating variables between stimuli and responses" and that they can be analyzed from self-observance or externally inferred from the behavior of the person (Vandermeeren, 2005, p. 1319). Purschke (Purschke, 2015, p. 49) defines *attitudes* as "relevance-driven targeting and evaluation routines on a high level of activation". They are part of an individual's knowledge and are "(re)constructed in interaction" (Purschke, 2015, p. 49). Applied to the language situation, this means that *language attitudes* are tendencies to answer with a specific type of (language) behavior to specific languages/speech styles or language situations (Vandermeeren, 2005, p. 1319). Huesmann (Huesmann, 2017, p. 14) states that *attitudes* are used to explain an individual's language behavior. *Attitude* in this framework is divided into three components: a cognitive, an affective and a conative part. Cognitive denotes the way an object is perceived, affective the judgement of said object and conative a tendency for a specific action (Huesmann, 2017, p. 15). Dividing *attitude* into three components can, however, be problematic as the actual situation of evaluating language is part of a complex judging process that cannot clearly be split into three parts (Purschke, 2014, p. 133).

Beside sociolinguistics, *attitudes* have also been studied in the research field of discourse analysis. Martin and Rose (Martin and Rose, 2013, p. 25) propose the *theory of appraisal*, which deals with evaluation in texts. More precisely, the purpose of this theory is to "model language's ability to express and negotiate opinions and attitudes within text" (Taboada and Grieve, 2004, p. 159). There are three aspects to appraisal: the *attitude*, its corresponding amplification and its source. Amplification is defined as the varying intensity of an *attitude* and source as the origin of the *attitude*, i.e. the speaker who expresses the *attitude* (Martin and Rose, 2013, p. 42 and 48). *Attitude* is then further divided into affect, judgment and appreciation (Martin and Rose, 2013, p. 29). Affect is the most interesting and important one for this thesis as it denotes the way people express their feelings in a discourse. This is where the concepts of *negative* and *positive* are furthermore brought into the equation. Martin and Rose write: "Firstly, we can have good feelings or bad feelings, so affect can be **positive** or **negative**." (Martin and Rose, 2013, p. 29) (bold in original statement). This statement is subsequently supported by providing examples to illustrate what a *negative* or a *positive* sentence would look like. "I was torn to pieces." is presented as a *negative* and "We were ecstatic." as a *positive* sentence (Martin and Rose, 2013, p. 32). Interestingly, no further explanation of the usage of *positive* or *negative* is given and both example sentences are shown with only a couple of sentences as context. While this certainly serves as a first intuition to the concepts, it is not enough to clearly understand and define what the authors really mean by *positive* and *negative*. Also, *feelings*

is used to further explain but is never defined in itself although it is illogical to assume that concepts such as *attitude* and *affect* need explanation whereas *positive*, *negative* and *feelings* do not. The other two sub-elements of *attitude* described in this context are *judgment* and *appreciation*. *Judgement* denotes the judgement of people's character and can also be either *positive* or *negative* while *appreciation* is about how a person feels about certain objects (Martin and Rose, 2013, p. 32 and 37). That way, the authors (Martin and Rose (2013)) subdivide *attitude* as comprising of *affect*, *judgment* and *appreciation*. It should, however, be noted that this division of *attitude* into affect, judgement and appreciation does not come without challenges. Taboada and Grieve (Taboada and Grieve, 2004, p. 160) analyze, for instance, adjectives and state that whether they are meant in a judgemental, affective or appreciative way depends on the context they are used in. Therefore, choosing one of the three options out of context is problematic and does not represent the way language actually works.

### 2.1.3 Questions Related to those Concepts

To sum up, concepts related to sentiment such as feelings, attitude, emotions as well as *positivity* and *negativity* are studied in several research domains like different strands of psychology (social and positive psychology) and linguistics (sociolinguistics and discourse analysis). However, studying those ideas does not necessarily mean that they are well-defined. Attitude is likely to be the most extensively defined concept of all of them, summarized as "an evaluation of an object of thought" (Bohner and Dickel, 2011, p. 392).

Feelings and emotions are also discussed in the literature reviewed above, but are not nearly as extensively defined as attitude. The same applies to *positive* and *negative*. For both of those concepts, examples are mentioned but no deep definition is applied leading to the assumption that those are considered to be intuitively understandable by researchers and laymen alike. For instance, Fredrickson (Fredrickson, 1998, p. 5) mentions anger and fear to illustrate *negative* and joy and interest to portray *positive* emotions. Martin and Rose (Martin and Rose, 2013, p. 32) give "I was torn to pieces." as example for a *negative* and "We were ecstatic." as example for a *positive* sentence. Relying on the intuition of people to correctly understand the meaning of a term is, however, rather problematic. This common certitude that every person has a general understanding of what emotion, feeling, *positive* and *negative* is, is founded on a certain intuitive notion that is deemed common, but that is not quantifiable. This understanding is truly personal and can differ (more or less greatly) from person to person. It is therefore essential to define the fundamental terms for a study in order to avoid possible misunderstandings or ambiguity in meaning. For this reason, it is important to examine sentiment as deeply and as extensively as possible before coming up with a clear working definition for this thesis.

### 2.1.4 Moving towards the Analysis of Sentiment

Early research in subjectivity and sentiment analysis has approached the problem by not defining *sentiment* right away but first of all *semantic orientation*. This term is coined as the "evaluative characterization of a word's deviation from the norm for its semantic group" (Hatzivassiloglou and Wiebe (2000), chapter *1 Introduction*). Hatzivassiloglou and Wiebe (2000) furthermore write: "Words that encode a desirable state (e.g., beautiful, unbiased) have a positive orientation, while words that represent undesirable states have a negative orientation" (chapter *2 Semantic Orientation*). Those reflections are based on previous linguistic research by Lehrer (1974). The author examines semantic fields, i.e. "[...] a 'group of words closely related in meaning' [...]" (Lehrer, 1974, p. 1) and notes that expressions are frequently on a scale with one end being *positive* and the other end being *negative* (Lehrer, 1974, p. 68). This intuition of language being on a scale with shifts from the norm being either on the *positive* or *negative* side was also noted by Lyons (Lyons, 1969, p. 467). Similar to researchers in psychology and discourse analysis, *positive* and *negative* are mentioned and the idea of a scale ranging from one value to the other are given. Furthermore, Lehrer (Lehrer, 1974, p. 68) provides examples such as "big" for the *positive* and "small" for the *negative* end of the scale. It is interesting that this observation has been made quite early in literature and yet has not been explored further by more recent research. This assumption that *positive* and *negative* are just obvious and intuitive concepts is also found in more recent sentiment analysis literature. For instance, Korayem et al. (2012) explain the classification for sentiment as determining whether a subjective text is *positive, negative, neutral* or *mixed* and additionally provide example sentences for each of those concepts (p.128). They suppose that one example per label is enough to understand what they mean by *positive, negative, neutral* or *mixed* suggesting thus that the concepts are easily shared between people with different prior knowledge and do not need further explanation. Liu (2015) also writes about *positive* and *negative* and mentions example sentences for both cases (p.11). However, neither Korayem et al. (2012) nor Liu (2015) go beyond the sentence level for explaining *positivity* and *negativity*.

Attempts have been made by the research community to differentiate between different subjective terms like affect, feeling, emotion, sentiment and opinion (Munezero et al. (2014)) and to go beyond the simplistic view of seeing sentiment as having three values (*positive, negative* or *neutral*) that does not question what sentiment or opinion actually is (Li and Hovy (2015)). However, those attempts are not that frequent. The focus of scientists has rather been put on training and improving ML algorithms for sentiment analysis/opinion mining and comparably little progress has been made towards comprehending the deeper meaning of sentiment such as how and why they are chosen (Li and Hovy, 2015, p. 9). I believe that it is essential to examine this further by collaborating with researchers from other domains, especially from psychology. As mentioned above, they study (*positive*) emotions and attitudes and could therefore provide crucial insights that can be transferred to working with sentiment. Additionally, collaborating with scholars from discourse analysis and sociolinguistics could be beneficial to not only study the cognitive (psychological) aspects of sentiment and related concepts, but to also gain insights from linguistics. Later in this thesis (see chapter 3.4), it will be shown that the understanding of *positive, negative*

and *neutral* is indeed not intuitively equal for everybody and that no clear definitions of those concepts leads to confusion. This is a practical example of why more research needs to be done towards clearly defining sentiment and the categories that belong to it.

### 2.1.5 Understanding of Sentiment for this Thesis

For this thesis, a rather simplified definition of sentiment is used, despite having pointed out the limitations of existing approaches above. The reason for this choice is that this is, to the best of my knowledge, the first work on sentiment analysis for the Luxembourgish language. No preliminary work by other researchers existed meaning that this research had to start from the very beginning. Consequently, the first step was to build a corpus for Luxembourgish sentiment detection. Next, as many annotations in this corpus as possible for training a system had to be provided. Enriching a corpus with such labels is essential groundwork required in order to implement algorithms capable of finding sentiment patterns in data, learning those structures and being able to detect sentiment in new and unseen cases. The experiments of ML, i.e. teaching the computer to extract patterns, will be discussed in chapter 4. Due to those limitations, this simple definition of sentiment was chosen to be able to focus the research attention not only on the development of a new definition of sentiment but also on the practical building of an annotated corpus and a sentiment detection system.

Sentiment is seen as emotional expressions of language situated on a scale ranging from *positive* to *negative* with *neutral* in the middle. In this sense, I follow Hatzivassiloglou and Wiebe (2000) in seeing each word as having a semantic orientation (chapter *1 Introduction*). 'Beautiful', for instance, would be *positive* and in contrast to 'ugly' (Hatzivassiloglou and Wiebe (2000), chapter *1 Introduction*). *Neutral* accordingly means not carrying any sentiment (Liu, 2015, p. 21). An example of this scale is figure 2.1 which shows three adjectives to describe for example the physical appearance of someone ranging from *negative* (*ellen* 'ugly') to *neutral* (*normal* 'normal') and further on to *positive* (*schéin* 'beautiful'). Additional examples are shown in table 2.1.

Table 2.1 – More examples for sentiment on a scale from *negative* to *positive*.

| Word | Translation | Sentiment |
|---|---|---|
| krank | sick | *negative* |
| mëttelméisseg | average | *neutral* |
| gesond | healthy | *positive* |
| trauereg | sad | *negative* |
| net schlecht | ok | *neutral* |
| extatesch | ecstatic | *positive* |
| pessimistesch | pessimistic | *negative* |
| net schlecht | alright | *neutral* |
| optimistesch | optimistic | *positive* |


Figure 2.1 – Sentiment on a scale ranging from *negative* to *positive*.

Not only words can carry either a *negative*, *neutral* or *positive* sentiment, but also bigger syntactic units such as sentences or user comments that will be the textual basis in this thesis. The reason for this is that those sentences, user comments or other syntactic units are comprised of words and other tokens that carry a semantic orientation. Determining the semantic orientation of those bigger units comes with challenges though, as expressions of language are ambiguous and diverse. Intuitively, one could, for instance, think that the semantic orientation of a single word token (*negative, neutral* or *positive*) can be used directly to find the overall orientation of a whole sentence. Unfortunately, this does not always work. Let us examine the arbitrary example sentence *De Manuel ass mega léif an ech freeë mech mega op eist Treffen muer.* 'Manuel is super nice and I am very much looking forward to our meeting tomorrow.' to illustrate this point. When reading this sentence made up by me, a human reader would likely directly understand it to be *positive*, if not expressed in a sarcastic tone. * Yet, simply counting the occurrence of words with a *negative, neutral* or *positive* semantic orientation in such a sentence and taking the majority class would not lead to *positive*. This example sentence contains ten *neutral* (De, Manuel, ass, an, ech, mech, op, eisen, Treffen, muer) and four *positive* (mega, léif, freeë, mega) words. Subsequently, there are far more *neutral* than *positive* words and the sentence should therefore be *neutral*. Why is it then that a human reader could see that it is not a *neutral* but a *positive* sentence? The most obvious explanation for this is that every human comes with some prior experience and background knowledge making it possible to adequately understand what is meant in different situations. This background knowledge is therefore helpful for going beyond the word level and making sense in more complex situations. The goal of this PhD thesis is, however, to automatically train a computer to understand expressions of Luxembourgish. A well-trained system should hence be capable of adequately detecting sentiment in known as well as in unknown cases.

I am well aware of the fact that this definition of perceiving sentiment as emotional expressions of language of either *negative, neutral* or *positive* value comes with even more of its own limitations that will be pointed out now. First, language is highly complex and stating that expressions are on a scale of semantic orientation with either *negative*, *neutral* or *positive* simplifies the problem of sentiment to a great extend. In the analysis of sentiment, the most diverse areas of linguistics actually play a role. Although it is mainly about pragmatics, the areas of phonology, morphology, semantics and syntax are also involved in sentiment recognition. For this reason, it is very simplistic to see sentiment purely as a scale. However, even implementing a more fine-grained approach such as a range from very *negative* to very *positive* with several intermediate steps will always just stay an ap-

*Sarcasm detection is a research field of its own that is related to sentiment analysis, the main approach of this thesis (Zhang and Teng, 2021, p. 18). Dealing with the automatic detection of sarcasm with the help of a computer is therefore beyond the scope of this thesis.

proximation of what is really expressed in language. No categorization can grasp the whole scope of grading that happens when people express some sort of sentimental or emotional expressions. Second, this definition of sentiment relies on the intuitive understanding of *negative, neutral* and *positive* by speakers of a language. Whereas (native) speakers of the same language certainly have some common intuitions, they are not 100% the same, as every person has a different background and experience in using a language. Additionally, personal preferences might be different. If we look at the example portrayed in figure 2.1 and think about the usage of those adjectives in context, this becomes clear. One person might enjoy the idea of being considered *schéin* 'pretty' while another might be very much against them being seen as pretty. Somebody might like being considered *normal* 'normal', whereas somebody else might not at all appreciate the idea of normality.

This potential of (dis)agreement can also be found in the corpus data that is the textual basis for this work. One important part of this thesis was the annotation for sentiment of the corpus provided by the partner RTL Luxembourg by means of different techniques. Such annotations are the foundation for training a computer to learn what *negative, neutral* or *positive* looks like in language. The goal of such a training is to ideally be successful enough to reliably perform sentiment detection in an independent manner. For example, a well-trained system should label the example sentence from earlier *De Manuel ass mega léif an ech freeë mech mega op eist Treffen muer.* 'Manuel is super nice and I am very much looking forward to our meeting tomorrow.' as *positive*, just like most humans would. One part of this annotation process of the corpus was the labeling of sentences, which was done by asking several people to look at the same sentences and to intuitively decide whether they understand this sentence to be *negative, neutral* or *positive*. While the different annotations and the crowdsourcing process itself are discussed in greater detail in chapter 3.3, comparing their answers to the very same sentences gives us interesting insights into the fact that this semantic orientation shown in figure 2.1 is not always that easy to detect.

Examples from the crowdsourced annotated part of the RTL corpus (see chapter 3.3.2 for a more detailed description of this corpus section) show that people can indeed agree on the allocation of sentiment in specific cases. *Negatively* annotated by all three annotators were the following sentences: *Et huet een d Gefill wie wann bei eisen Autoen all Schick Schnack drann as an e vu lauter Optiounen nik mei wees waat den Auto alles kann, awer dei Elementar Saachen nik mei zielen.* 'One gets the feeling that our cars are full of knick-knack and that, because of all those options, one doesn't know anymore what the car is capable of while the basic things are not important.' (RTL corpus, 01/2009, sentence ID 1117ab) and *DAAT ASS SCHON SEIT E PUER HONNERT JOER ESOU AN MIR LOOSSEN ONS NACH EMMER USCHMIEREN.* 'It's been that way for a couple hundred of years and we are still getting ripped off.' (RTL corpus, 01/2009, sentence ID 17571ab). Examples for *neutrally* annotated sentences are *Ass daat richteg?* 'Is that correct?' (RTL corpus, 2008, sentence ID 1876aa) and *Ze betounen ass zwar och nach, dass éen Meteokollég vun Holler bai Waiswampach mir éng rekordverdächteg Minimal-Temperatur eran gemailt hat, an zwar waren et do -19,7C!* 'It should also be pointed out that a friend of mine passionate about the weather from Holler close to Waiswampach sent me an email with a record-breaking minimum temperature, namely -19.7C!' (RTL corpus, 01/2009, sentence ID 1330ab). All

annotators agreed that the following instances were *positive*: *Denger Famill drecken ech main déifsten't Mattgefill aus.* 'I express my deepest sympathy to your family.' (RTL corpus, 01/2009, sentence ID 6527ab) and *Eddy thierry an merci fir deng ganz filmwieker deis du eis hannerlos hues!* 'Goodbye Thierry and thanks for all your cinematic works you left us!' (RTL corpus, 01/2009, sentence ID 11672ab). However, the annotated corpus also includes cases in which all three annotators provided a different sentiment for an instance proving that seeing sentiment as a scale from *negative* to *positive* can help to guide the natural intuition of people but does not have to.

To sum up, the definition of sentiment for this work is to perceive it as an emotive expression of language that can be either of *negative, neutral* or *positive* semantic orientation. This semantic orientation is on a scale, like shown in figure 2.1. This definition relies heavily on the intuitive understanding of *negative, neutral* and *positive* by the (native) speaker of a language which leaves leeway for (miss-)interpretation. Nevertheless, seeing sentiment as part of a spectrum, like in table 2.1, guides people towards structuring their personal intuitive understanding of sentiment without specifying too much in advance. That way, they can rely to a great extend on their personal prior experience and training to adequately grasp the intended sentiment meaning of a sentence, user comment or other segment of language. In the next section, the challenges of sentiment analysis will be discussed in more detail, especially when using such an open definition of the term sentiment.

## 2.2 Challenges in Sentiment Analysis

Categorizing an entity such as a sentence or a whole comment as *positive*, *negative* or *neutral* might at first glance seem easy and intuitive. As was shown in chapter 2.1, it is not always that obvious which label is the adequate one. Nevertheless, early work in sentiment analysis has used this simple classification technique for determining the *positive, negative* or *neutral* sentiment of a whole text or text span such as a movie review (Pang et al., 2002, p. 79) without further questioning the definition of sentiment, *negative, positive* and *neutral*. In this section, I will discuss the challenges of doing sentiment analysis, especially with a broad definition of the term sentiment and the concepts *negative, neutral* and *positive*. It is crucial to think about those challenges, as they impact the way sentiment is perceived and assigned in an annotation process. During an annotation process, sentiment is attributed to utterances by (native) speakers of a language. Those are then later on used to train and test an automatic sentiment detection system and are the basis for automatically giving a sentiment score to unseen instances. The foundation for labeling are carefully thought-out guidelines that are crucial to ensure as consistent and high-quality annotations as possible. Ideally, those can include information about how to deal with such challenging examples which will be discussed below in order to avoid confusion. This is important, as annotation tasks are often carried out by several annotators and different sentiment scores for the exact same text span can lead to problems in the training of the ML system. More information about the way annotation is undertaken can be found in chapter 2.9. The way annotation was performed in this thesis is also described in chapter 3.3.

### 2.2.1 Associations of Sentiment

First of all, it should be noted that sentiment can be associated with the speaker, listener or one or more of the entities that are referenced in the utterance itself (Mohammad, 2017, p. 3). If a sentence such as "James: The pop star suffered a fatal overdose of heroine." is considered, for instance, it is not clear what is happening without giving further context. Is this supposed to be a newspaper headline? Is James the speaker of the event? And if he is: Is he personally emotionally affected by the death of the celebrity or is he just reporting about the passing in a neutral tone (Mohammad (2017), p. 3)? It is crucial to correctly identify the intended meaning, as it influences the sentiment that needs to be attributed to this sentence. There are a couple of different ways to do that. First, if the utterance is a newspaper headline, the sentiment of this sentence is likely to be *neutral*, as newspapers usually report in an objective way without giving any judgement. Second, the sentence could also be part of a conversation between James and his friends. In this case, there are two options for interpreting: Either James reports the passing of the star in a *neutral* tone or he is personally saddened by the event and has trouble speaking about it. In the first case, the sentence should be assigned a *neutral* sentiment, as it deals with an objective telling of a fact. In the second case, the sentence could be attributed a *negative* sentiment to accommodate the personal feeling (sadness) of the speaker James. This example shows that assigning a sentiment to an instance without adequate background knowledge is a hard task.

### 2.2.2 Aspects in a Text Span

A second possible challenge for sentiment analysis is the amount of different sentiments that can be expressed in a single text. Contrary to the approach pursued by early researchers who just attributed one sentiment per text (e.g. Pang et al. (2002)), different sentiments can be expressed in the same text or even in the same text span (Mohammad (2017), p. 4). A movie review might, for example, voice a *positive* sentiment towards the main actor of the movie whereas the overall judgement of the plot might be *negative*. A person that tells their friend about their new laptop might speak highly of the color and the design, but very *negatively* about the battery life and so on. Giving only one single sentiment might thus not be sufficient in such cases.

There are different ways to resolve this kind of difficulty. One way is to use a technique called aspect-based sentiment analysis (e.g. Pontiki et al. (2014)) which does not come up with a single sentiment for the whole text in question, but is capable of attributing a different sentiment for each aspect. For the movie review example mentioned above, the actor would hence be tagged with a *positive* and the plot with a *negative* sentiment score. Both the color and the design of the laptop would receive a *positive* tag whereas the battery would take a *negative* sentiment tag. If applying aspect-based sentiment analysis is not an option in the specific research environment involved, traditional sentiment analysis methods might also be used. By traditional sentiment analysis methods, I mean the assigning of one single score (either *positive, negative* or *neutral*) to a whole text. In this case however,

it is absolutely essential to clearly state in the annotation guidelines how to deal with such situations. Otherwise, confusion is bound to happen and different annotators might attribute completely different sentiments to the same text. The conversation between friends about the new laptop could, for example, be labeled with a *positive* sentiment label if the annotator believes the color and the design to be the most important elements of the text. Another annotator might yet allocate a *negative* score due to the importance of the battery for the performance of a laptop or even a *neutral* score to balance the *positive* and *negative* aspects mentioned in the conversation. If three annotators were to annotate this conversation to use it as training data for an ML algorithm, the output should ideally be to have the same sentiment score for all three and not three different scores. Otherwise, this could lead to confusion, as ML tries to learn to detect patterns in data and generalizes it to unseen cases.

### 2.2.3 Expressing Sarcasm

A third case that is very challenging for sentiment analysis and annotation of such is sarcasm, i.e. the proclamation of something that is meant in the opposite way. In a more fine-grained view, sarcasm furthermore should be distinguished from irony (see e.g. Yaghoobian et al. (2021)). As sarcasm and irony are not the main focus of this thesis, the difference between those two terms will not be taken into account. A statement like "Oh wow! What a great view!" when entering a hotel room while on vacation could, for example, be interpreted in different ways. Without prior background knowledge, the reader could assume that the speaker is ecstatic and has never seen such a beautiful view, i.e. think that the value of the statement is of *positive* kind. It could also be read as a sarcastic statement if the speaker has just opened the window and looks straight at a noisy construction site that they were not warned about prior to their arrival. In this case, the meaning of the utterance would be *negative* despite the usage of *positive* words in the statement. Sarcastic instances are already hard to detect for humans, even if they are native speakers of the language the utterance is made in. It is even harder for ML algorithms to detect such cases, especially when no background knowledge of the situation is provided and examples are shown with very little or no context at all. Frequently, sarcasm detection is even treated as a separate field apart from or on top of sentiment analysis (e.g. Liebrecht et al. (2013), Parde and Nielsen (2018)).

Dealing with sarcasm in simple sentiment detection is challenging, because a single label has to be decided for each instance. One possibility is that the emotional state of the speaker is different from their attitude at the moment of expressing themselves. This could potentially be a *positive* emotional state due to mocking somebody but the general attitude might not be a *positive* one towards something or someone (Mohammad, 2016, p. 175). Even in such cases, traditional sentiment analysis approaches would require the choice of one single label, i.e. either *positive, negative* or *neutral*, which would not be easy in this case. In order to smooth this problem, some background knowledge could be provided to adequately chose a label for sentiment. Unfortunately, this is not always possible and done in practice due to a number of reasons such as lack of resources or time.

### 2.2.4 Further Challenges

Other aspects that require special attention when planning and implementing sentiment analysis are re-tweeting or quoting and rhetorical questions (Mohammad, 2016, p. 176). Without specification, it is not clear how those should be treated. Should the retweet of a completely sad and depressed (hence *negative*) tweet be labeled as *negative* although the person retweeting on their page adds a (*positive*) message to cheer the original author up? What about tweets that are just retweeted without any comment at all? And what about quotes or rhetorical questions? Should they be annotated *positive, negative* or *neutral* depending on the context they are used in and the general sentiment expressed in the surroundings? Best practice for such situations would be to define beforehand in the annotation guidelines if such cases should be treated as *neutral* or if they should keep the sentiment expressed by the initial author. Otherwise we might see annotators give very different labels to such cases which is not the ideal case.

So far, the attention of this chapter was mainly focused on challenging aspects that need to be taken into account when planning the annotation process of sentiments. However, difficulties do not only concern the annotation part but also the actual ML implementation with the data set created during the annotation process. Frequently, data sets constructed for sentiment analysis tasks only contain "gold-standard sentiment labels" (Kenyon-Dean et al., 2018, p. 1886). This means that every annotation not passing a certain threshold is discarded. More precisely, (native) speakers of a language label a certain amount of instances with a sentiment label in the annotation process. By doing so, they follow the guidelines provided by the researcher. From all of those annotations, the researcher then only chooses a certain amount to use for training and testing their ML algorithms creating the gold standard from all annotations that were provided to them. One way to do this would be to say that, for example, only those annotations that were given the same label by all annotators will be considered for training and testing, i.e. included in the gold standard. However, eliminating every single case of disagreement is questionable when it comes to applying the implemented automatic system to a real-world setting. The reason for this is that language is ambiguous and speakers have different views of how to see the world. It is therefore not unlikely that not everybody will agree and give the same sentiment for the same instance. Trained sentiment analysis systems do not know which instances are "noisy" or "controversial" (Kenyon-Dean et al., 2018, p. 1889). If they are only trained on perfect examples on which all annotators agreed, they might perform poorly on real life data that is not always as easy and clear as the data gathered in the gold standard. An example of such "noisy" data is this sentence from the RTL corpus which is the textual basis for this thesis: *Vive de Grand-Duc well duerch den Haff hu Mir Letzebuerger eis Politesch Freiheet behalen noom Simon vu Preisen.* 'Long live the grand duke, because we Luxembourgers have kept our political freedom through him after Simon of Prussia.' (RTL corpus, 2008, sentence ID 148aa). It was annotated by three different annotators to use as training example for detecting sentiment. Two annotators labeled it with *positive* and one with *negative*.

Figure 2.2 summarizes the different challenges related to sentiment analysis. Those challenges portrayed in this chapter show that labeling for sentiment is not as easy as just deciding randomly if an utterance is *negative, positive* or *neutral*. Instead, sentiment analysis requires a careful planning of both the annotation process and the ML implementation part. Without those planning steps, the performance of the system might not be as good as required leading to unsatisfactory results. Naturally, the amount of time and effort spent on planning, dealing with challenges and training the system always depends on possible time- and budget-constraints the researcher might encounter. Not all difficulties related to sentiment detection are therefore likely to be addressed in a real research setting. Nevertheless, researchers should be aware of possible challenges when planning both the manual annotation and the algorithmic training process to be aware of potential implications of choices.

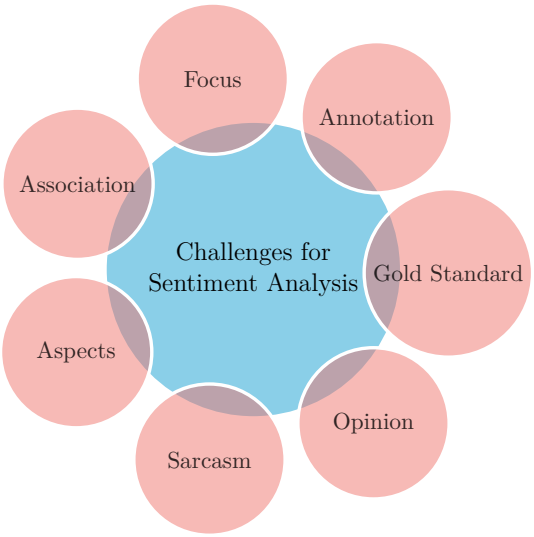



Figure 2.2 – Summary of challenges in sentiment analysis.

## 2.3 Detection of Sentiment on Different Levels

The detection of sentiment can be performed on different layers of a text depending on the requirements of the research project. With layers, I denote the various depths of a text that can potentially be studied. For example, the goal of a study might be to extract the sentiment of single word tokens in a corpus to examine the overall distribution of sentiment values. Another possible research project's aim could be to discover the sentiment allocation of all product reviews in a corpus. For the first example, the layer of study would be word-level whereas the second example would explore the sentiment distribution on whole texts or document level. Sentiment analysis on document-, sentence- and aspect-level are the most prominent possible depths for sentiment analysis. Those will be further discussed in the following sections.

### 2.3.1 Sentiment Analysis on Document-Level

Document-level sentiment analysis is the most general one of the three approaches discussed in this chapter. It looks at the whole text and determines if a *positive, negative* or *neutral* label should be attributed to it (Liu, 2015, pp. 47-69). Document in this case can refer to a text with a variety of lengths, ranging from long documents such as newspaper articles to short user comments on social media. The task of detecting sentiment on document-level was already tried out at an early stage of sentiment analysis research using supervised ML techniques and treating sentiment detection as a simple text classification problem (e.g. Pang et al. (2002)). Using classical text classification methods makes this approach easy to implement which is great for quickly solving problems. Drawbacks of detecting sentiment on document-level are not insignificant. Frequently, this approach is used for short texts such as movie or product reviews to portray the overall sentiment in those types of texts. From a consumer point of view, those kind of documents are important to make different life choices such as if that new movie that just came out is worth seeing or if that laptop they are currently interested in is really worth buying. A customer checking laptop reviews on a web-based platform, for instance, is likely not just interested in the overall sentiment score but wants to know the details as well. They might be especially looking for a laptop with high battery life and a nice keyboard while not being that interested in the design or the performance of the machine. Knowing that a laptop has an overall *positive* sentiment score will therefore be insufficient to satisfy their quest for information. A more fine-grained approach would likely be more helpful.

Apart from reviews, document-level sentiment analysis is also applied to other kinds of texts like blog posts, news articles and user comments. While news articles are often well-organized, the length, structure and language of blog posts and user comments can vary to great extents. Those kind of texts are often not clearly built towards the goal of judging some entity and are often even directed towards different ones which makes using a simple text classification task not easy for those kind of texts (Liu, 2015, pp. 68-69). Especially blog posts and user comments are likely to have spelling mistakes and use colloquial forms which also has an impact on the performance of classical text classification methods. A shortened example from the RTL corpus of user comments underlines those points:

> [...] Wat huet de Bistum da mat den Öffnungszeiten vun de Buttiker ze dinn??? Virwat net och nach un de Grand-Duc schreiwen? Eise Letzebuerger Modell ass zwar speziell awer och alt net esou retrograd! Onwahrscheinlech esou eppes ze liesen. Zum Thema selwer : ech hu kee Verständnis dofir. Ganz vill Leit schaffen op Schichten an de weekend : Doktoren Infirmièren Polizei Piloten Taxichauffeuren an der Restauratioun etc. "De" bekloen sech net iwwert hirt Familieliewen. Virwat kann eng vendeuse dann net samsdeg bis 8. Auer schaffen? (RTL corpus, 06/2010, sentence IDs 9678as-9689as)

The English translation of this comment is: "[...] What has the diocese got to do with the opening hours of the shops ??? Why not write to the grand duke as well? Our Luxembourg model is special but not too retrograde! Unlikely to read something like this.

On the subject itself: I don't agree with it. A lot of people work shifts on the weekends: doctors nurses police pilots taxi drivers in restaurants etc. "They" do not complain about their family life. Why can't a saleswoman work until 8pm on Saturday?"

This comment is not full of colloquial forms, but some spelling mistakes can still be observed. For instance, *vendeuse* 'saleswoman' and *weekend* 'weekend' are not capitalized. Furthermore, the comment addresses different points and thus has different targets of sentiment. The first part makes a reference to a previous comment about contacting the grand duke and the diocese as well as the Luxembourgish model. Next, the commentator states his personal opinion of a possible opening of shops on Saturdays. Labeling such a comment with different sections could be challenging. The first section about the Luxembourgish model is more *positive* supporting the usefulness of said model whereas the second part about the shop assistants is more *negative*, as the writer of the comment does not agree that they should be treated differently from other occupational groups working shifts. Attributing an overall sentiment to such documents is thus not an easy task, as different parts carry different sentiments. Nevertheless, document-level sentiment analysis only allows for one overall sentiment label (*negative, neutral* or *positive*) and one of those has to be chosen.

### 2.3.2 Sentiment Analysis on Sentence-Level

Sentence-level sentiment detection goes one level deeper than just looking at documents and examines the opinion expressed in a single sentence (Liu, 2015, pp. 70-89). This kind of approach is closer to real-life scenarios than document-level sentiment analysis, as people are normally interested in the sentiment directed towards different targets (Liu, 2015, pp. 70-71), like the sentiment towards the battery life of a laptop or the color of the product. It can, however, still just assign one sentiment score to a whole sentence. Therefore, it still could not adequately deal with instances such as "The battery life is great but I absolutely hate this ugly blue color." For sentences like this one, only one label can be used despite the two opposing sentiments inside the sentence (*positive* for battery life and *negative* for the blue color). The sentiment of other kinds of sentences like "I could not be happier right now!" seems, however, easier conceivable, as only one sentiment is expressed. There are still two different ways of interpretation: If such a sentence is expressed in a sarcastic way, it could be labeled *negative* and if the speaker intends the literal meaning, a *positive* label could be attributed to this case. In practice, sentence-level sentiment analysis can be implemented like a classification task treating one sentence as one short document. This is more challenging than detecting the opinion on a longer document (such as a user comment or a movie review). The reason for this is that the algorithms have a lot less context to learn from during the training process than when working with longer texts (Liu, 2015, pp. 70-71).

### 2.3.3 Sentiment Analysis on Aspect-Level

The third approach for opinion mining that is introduced in this chapter is called aspect-based sentiment analysis. In this context, aspect denotes part of an expression, such as the shape of a shampoo bottle in a product review about a new shampoo. It is thus different from the way linguistics employ the term aspect (see e.g. Roussel et al. (2019)). Aspect-based sentiment analysis goes deeper than document- or sentence-level sentiment detection and is thus the most fine-grained approach of the three. This method aims to detect the opinion expressed towards different entities in a text (Liu, 2015, p. 91). Aspects in this context mean different targets that a sentiment can be directed towards. For instance, a sentence like "I love my phone's color but I'm so disappointed with the battery." contains two different aspects with conflicting sentiments. The first aspect, the color of the phone, is perceived as *positive* and the second one, the phone's battery life, is seen as *negative*.

On the implementation level, aspect-based sentiment detection is a two-step process. First, the aspects in question need to be extracted from the text (Liu, 2015, p. 91). Second, the sentiment used for each aspect should be determined (Liu, 2015, p. 91). Going back to the phone example, this would mean that the entity in question (*phone*) has to be first of all detected. Subsequently, one would need to extract the aspects (*phone's color* and *battery*) and end with determining the sentiments (*positive* and *negative*). Whereas being most adequate for real-life applications, aspect-based sentiment detection is a complicated task requiring a lot of rule-based knowledge and cannot be considered a solved task yet (Liu, 2015, p. 135).

### 2.3.4 Levels of Sentiment Analysis for this Thesis

In this work, the main focus is on two of the three sentiment analysis tasks described above. The prominent part of this thesis concentrates on sentence-level sentiment detection whereas a short document-level sentiment analysis part was also included. More precisely, the attention was mostly on training the computer to learn *negative, neutral* and *positive* sentiment expressions in sentences and only a small part of the experiments was run with sentiment on the whole document level, i.e. in this case user comment level. Note that I am convinced that aspect-based sentiment analysis is the kind of task that is closest to real-life settings. This approach is, as Liu (Liu, 2015, p. 135) stated, the most complex one and not yet adequately resolved. Due to the fact that this thesis is, to the best of my knowledge, the first research working on sentiment analysis for the Luxembourgish language, it was subsequently decided to focus the attention on sentiment detection on document- and sentence-level. Future work could then build upon the findings of this thesis and implement an aspect-based sentiment detection system for Luxembourgish. Such a system would be beneficial for many language instances and could also better grasp sentiment in the rich and diverse RTL corpus (see chapter 3.1). Figure 2.3 summarizes sentiment detection from document-level (largest) to sentence-level (more fine-grained than document-level) to aspect-based-level (the most specific).


Figure 2.3 – Different levels of sentiment analysis from broadest (document) to finest (aspect-based).

## 2.4 Machine Learning

Machine learning is the main field of computational approaches that was used for the experiments presented in chapter 4. Due to the interdisciplinarity and scope of this thesis, this chapter can only be understood as a brief introduction to the area. The main basic idea of ML is to provide examples for a certain task to a computer and educate it to learn typical patterns from those. It is closely related to the fields of statistics and data mining (Murphy (2012), preface). This concept of learning mentioned here is "the process of converting experience into expertise or knowledge." (Shalev-Shwartz and Ben-David, 2014, p. 1). The knowledge obtained from learning to discover patterns in a specific data set should then be taken to generalize to unseen cases and find such studied patterns in new data ((Shalev-Shwartz and Ben-David, 2014, p. 2) and (Murphy, 2012, preface and p. 1)) instead of simply imitating exactly what has already been seen before. For instance, the computer should not be able to only memorize that *D'Jenny ass mega genervt vum Anton.* 'Jenny is really annoyed by Anton.' carries a *negative* sentiment. Rather, the system should learn the general pattern of such an expression and apply this expertise to unseen (similar) cases. New examples such as *Ech si genervt vum Samara.* 'I am annoyed by Samara.' or *Den Anton nervt d'Jenny mega.* 'Anton really annoys Jenny.' should therefore not pose a problem and be correctly identified as *negative*. Over time, learning should lead to improvement and a more and more accurate performance of the desired action (Marsland, 2015, pp. 4-5).

Generally speaking, a computer does not have any prior knowledge facilitating the learning process before coming in contact with a certain type of data for the first time. However, it should be noted that "the stronger the prior knowledge [...] that one starts the learning process with, the easier it is to learn from further examples." (Shalev-Shwartz and Ben-David, 2014, p. 3). For smaller data sets, using ML might cost more time than just doing the job yourself manually. Nevertheless, computers come with one very important advantage over humans which should not be neglected. They do not get tired and thus become less concentrated after a while which could lead to possible mistakes when a task is carried out by humans. Additionally, computers come with large memory capacities and processing speed (Shalev-Shwartz and Ben-David, 2014, p. 4) exceeding human capacities and can therefore be used as a support (Shalev-Shwartz and Ben-David, 2014, p. 6) when the job involves very large collections of data.

### 2.4.1 Types of Machine Learning

There are several main areas of ML which carry out learning in different ways. The first one, which is the approach for this thesis, is supervised learning.

Supervised learning uses training data that comprises a set of examples with the correct target, i.e. label, to learn patterns in data (Shalev-Shwartz and Ben-David, 2014, p. 6). Another way of calling this is "learning from exemplars" (Marsland, 2015, p. 6). For this thesis, the examples in the training data would be sentences or whole user comments and the target either *negative, neutral, positive* or *undecided/not labeled*. In more formal terms, an input x is mapped to an output y. This is formalized as the set of a labeled input-output pairs

$$D = \{(x_i, y_i)\}_{i=1}^{N} \tag{2.1}$$

(Murphy, 2012, p. 2). In this notation, D is the training set, N is the number of training examples, $x_i$ the training input and $y_i$ the response variable of some kind (Murphy, 2012, p. 2). Figure 2.4 shows a supervised learning process in a different visualization manner (adapted from Desarkar and Das (2017)). In supervised learning, a training input and its label are converted to feature vectors and handed over to an ML algorithm for learning. This predictive model is subsequently used to obtain expected labels for new documents. Some examples of supervised learning are all sorts of text classification such as spam detection or sentiment analysis and image classification (working on resolving tasks such as *Is this a picture of a dog or a cat?*). Other kinds of supervised learning applications include age prediction (*What is the age of the viewer of a given video on YouTube?*) (Murphy, 2012, p. 9) or prediction of stock prices (Zhang and Teng, 2021, p. 21). Those last two tasks are examples of regression tasks which have a range of "real-valued numbers" as output (Zhang and Teng, 2021, p. 21). The category of supervised learning in this thesis is called classification. This concept will be further explained in chapter 2.7, together with the different algorithms used for classification.

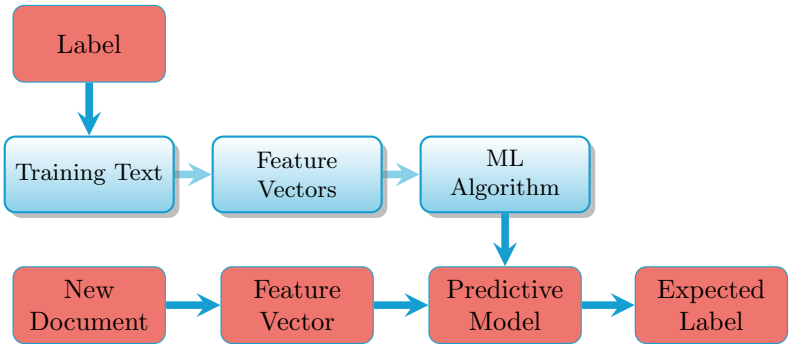



Figure 2.4 – Supervised learning flow.

Unsupervised learning lets algorithms find patterns in the input data and discover some kind of structure automatically (Marsland, 2015, p. 6). In formal terms, this could be noted as

$$D = \{x_i\}_{i=1}^{N} \tag{2.2}$$

with yet again D being the training set, $x_i$ the training input and N the number of training examples (Murphy, 2012, p. 2). Figure 2.5 displays the learning flow of an unsupervised system (adapted from Desarkar and Das (2017)). In contrast to the supervised way of learning shown in figure 2.4, training texts are provided without a label, but are still taken as input to an ML algorithm. The model secured from this training process can then be used to find structure in new documents, i.e. to perform clustering (Manning and Schütze, 1999, p. 232). To put it differently, unsupervised learning wants to "find clusters of similar inputs in the data without being explicitly told that these data points belong to one class and those to a different class." (Marsland, 2015, p. 281). Clustering could, for example, be performed on the RTL corpus (see chapter 3.1) and ideally would get some meaningful groupings such as clusters according to the topics of the news articles.

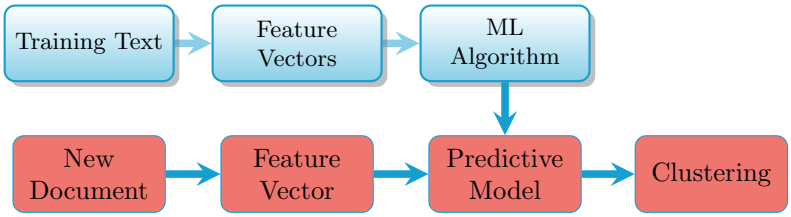



Figure 2.5 – Unsupervised learning flow.

The third field of ML is called reinforcement learning. In this setting, the training examples include more information as the test examples (Shalev-Shwartz and Ben-David, 2014, p. 5). In other words, "information is provided about whether or not the answer is correct, but not how to improve it." (Marsland, 2015, p. 231). Reinforcement learning has to apply different strategies and find out for itself which of those are the best (Marsland, 2015, p. 231). Figure 2.6 shows a simplified reinforcement learning flow (adapted from (Marsland, 2015, p. 233)). In this cycle, the agent carries out an action $a_t$ in state $s_t$. For this action, it obtains a reward ($r_{t+1}$) from the environment and finally ends up in state $s_{t+1}$ (Marsland, 2015, p. 233). Reinforcement learning basically works the way a baby, for example, learns to walk by receiving positive signals when it does well and negative ones when it is not correct (Murphy, 2012, p. 2).

Deep learning (DL) is another subfield of ML that is gaining more and more popularity and shows strong results on numerous ML tasks (Shalev-Shwartz and Ben-David, 2014, p. 228). It has become an important, if not the most important, approach of our times (Zhang and Teng, 2021, p. 4). Due to the complexity of the field, discussing DL in great detail would be beyond the scope of this thesis. Notwithstanding, it is important to give a brief introduction at this point. This should help the reader to better understand the experiments that were carried out with *BERT* (see chapters 4.1.2, 4.2.2, 4.3.2), a state-of-the-art DL system which will be further introduced in chapter 2.7.6.

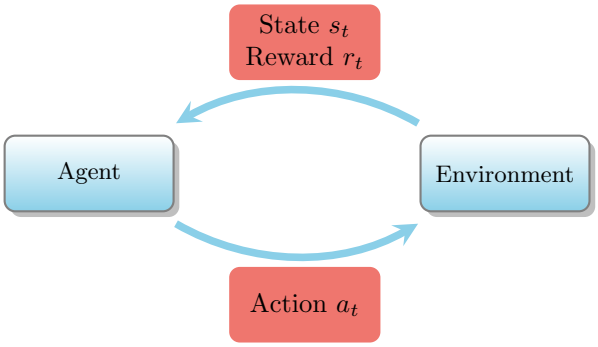



Figure 2.6 – Reinforcement learning flow.

The basis of DL is the usage of artificial neural networks (ANN), a kind of computational model that was inspired by the way a brain is set up (Shalev-Shwartz and Ben-David, 2014, p. 228). Those neural networks in the brain are comprised of many neurons that transmit information from one to another by means of electrochemical signals (Zaki and Meira Jr., 2020, p. 637). Through those connections between neurons, a brain is capable of carrying out "highly complex computations" (Shalev-Shwartz and Ben-David, 2014, p. 228). This structure was taken and adapted to create ANNs that can be used for diverse automatic learning processes with a computer.

What does this ANN look like? And how does it learn? Shalev-Shwartz and Ben-David (2014) write:

> "A neural network can be described as a directed graph whose nodes correspond to neurons and edges correspond to links between them. Each neuron receives as input a weighted sum of the outputs of the neurons connected to its incoming edges." (p. 228)

In more mathematical terms, this claim of what an ANN is can be formalized as a "weighted directed graph G = (V, E), with each node $v_i \in$ V representing a neuron, and each directed edge $(v_i, v_j) \in$ E representing a synaptic to dendritic connection from $v_i$ to $v_j$. The weight of the edge $w_{ij}$ denotes the synaptic strength" (Zaki and Meira Jr., 2020, p. 637). Figure 2.7 portrays the setup of a single neuron and its connections (Reading (2019)). In figure 2.8, a complete simple ANN with one hidden layer (Reading (2019)) is displayed. An artificial neuron consists of several parts (see figure 2.7). First, there are multiple inputs ($x_1$, $x_2$ and $x_3$). Each of those inputs $x_i$ are linked to a weight ($w_1$, $w_2$ and $w_3$) which act as connections between the inputs and the neuron. The weights imitate the synaptic strengths of non-artificial neural networks. Those weights and inputs are summed together by an *adder* ($\sum$). The sum of the inputs and weights are completed with a bias $b$ which effects the impact of the activation function. This function can have different forms and is used to influence the output $y_{out}$ of a neuron (Kantardzic, 2020, pp. 233).

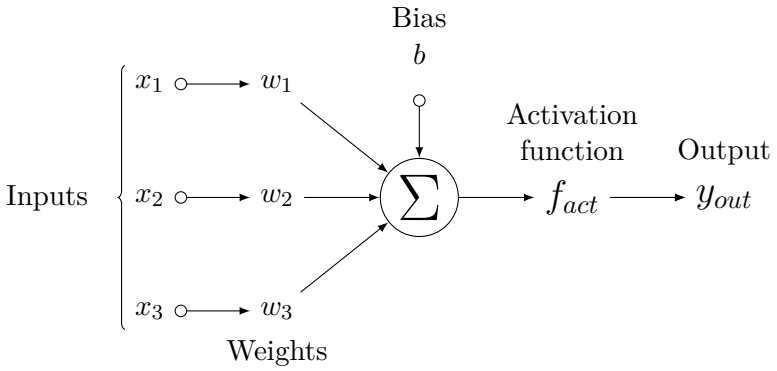



Figure 2.7 – Model of a single artificial neuron.

Figure 2.8 shows the simple setup of an ANN with one hidden layer to illustrate the general structure. It should be noted at this point that ANNs are often deeper and contain a lot more nodes and hidden layers as portrayed in this figure. The structure is simplified to a great extend to facilitate comprehension. This neural network has an input layer with five nodes that are connected to a single hidden layer with three nodes. This hidden layer is then again linked to the output layer which produces one output value. The input layer of the network includes one neuron for each feature and the subsequent hidden layer(s) a number of neurons that get passed the output of the nodes in the previous layers. The output layer receives the output of the hidden layer(s) and uses this to calculate a final output value (Fonseca and Cabral, 2019, p. 3). Important for this brief overview of ANNs is not only the layout of the network in general, but especially also the connections between the different nodes. Those carry weights that are crucial for the intelligence of the network and that are often initialized in a random manner (Fonseca and Cabral, 2019, p. 4). During training, the labeled training examples are fed to the network several times. Each time, the predicted outcome is compared to the expected label and the weights are adapted accordingly, which is called backpropagation (Fonseca and Cabral, 2019, p. 4).

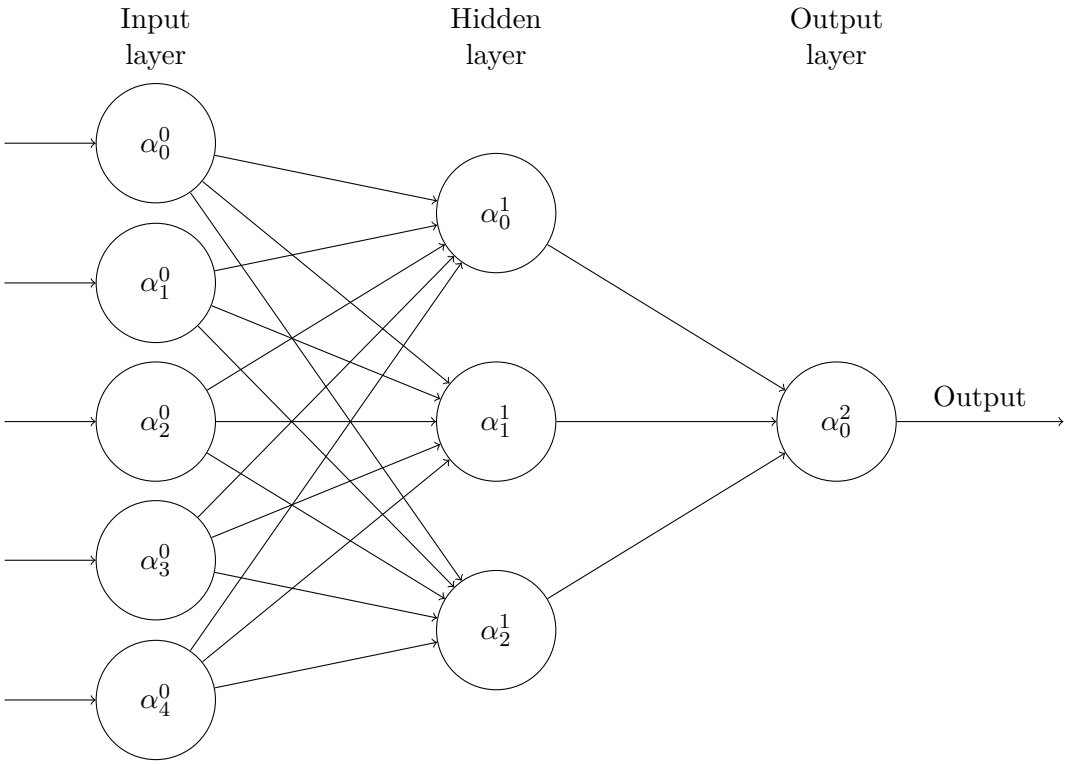



Figure 2.8 – A simple artificial neural network structure.

## 2.5 Natural Language Processing

A large part of the thesis deals with using ML techniques for NLP tasks (see chapter 4). NLP is a field that is closely related to computational linguistics (CL). As discussing fine-grained differences between computational linguistics and NLP is beyond the scope of this thesis, they will be treated as synonyms in the following chapter.

Computational linguistics is about the processing of natural language in either written or spoken form with the help of a computer ((Carstensen et al., 2010, p. 2) and (Lobin, 2010, p. 10)). Additionally, (Zhang and Teng, 2021, p. 3) mention that NLP can also deal with "synthesising human languages". It is a very inter-disciplinary field between linguistics and computer science, including artificial intelligence and data science aspects as well as components of psychology, cognitive and neural science that has been around since the 1950s (Zhang and Teng, 2021, p. 3). Ideally, an NLP system should be capable of determining "the structure of text, so it can answer questions about meaning or semantics of the written language." (Martinez, 2010, p. 353).

When NLP research first started in the 1950s, methods were mostly based only on rules ((Zhang and Teng, 2021, p. 4) and (Lobin, 2010, pp. 14-15)). They were in large parts replaced by statistical methods and ML during the 1980s, as those techniques can learn

patterns from data to make decisions (Lobin, 2010, pp. 15-16). This is a big advantage over rule-based systems that are not flexible and struggle a lot with challenges such as ambiguity in language (Zhang and Teng, 2021, p. 4). For instance, an ML system should be capable of learning that *mega* 'fantastic' and *meeeeeeeegggaaaa* 'fantastic' refer to the same concept whereas a rule-based system would struggle, as those words are not written the same way. The pragmatic difference between the standard form *mega* 'fantastic' and the enthusiastic one *meeeeeeeegggaaaa* 'fantastic' would be taught to an ML system by feeding those instances as examples to the training process. During the late 2000s, the technique of DL has become the dominant approach and outperforms traditional systems in many cases (Zhang and Teng, 2021, p. 4). For linguistics researchers, it might be interesting to note at this point that DL relies less on linguistic knowledge as statistical, ML and rule-based methods (Zhang and Teng, 2021, p. 4). Different NLP techniques include POS tagging, parsing and lemmatization that help structure texts and find meaning in them ((Martinez, 2010, p. 356) and (Zhang and Teng, 2021, pp. 6)). Those can be taken to help with a large variety of tasks such as text data mining, information retrieval, machine translation, document summarization ((Martinez, 2010, p. 356) and (Lobin, 2010, p. 19)) as well as sentiment analysis.

In this thesis, a variety of those methods were used to grasp meaning in the corpus. In order to better prepare the texts for analysis, preprocessing was performed with, among other techniques, lemmatization and POS tagging (see i.a. chapters 3.1, 4.1 and 4.2). A rule-based method to automatically annotate parts of the corpus with sentiment was also applied (see chapter 3.5). Besides, a range of ML and DL experiments on different corpus sections were performed to examine the best possible ways to detect sentiment in Luxembourgish user comments (see chapter 4). For some of those experiments, linguistic features were included combining the knowledge of Luxembourgish linguistics with different ML algorithms.

## 2.6 Converting Text to Vectors

To human beings that have learned to read, seeing a sequence of words in a language they understand and making sense of that sequence has become an automatic way of processing. However, a computer cannot work with those words directly which is why some transformation steps are required to allow the computer to access and understand those language points. If supervised classification of texts is the task, for instance, the model aims to predict the label $c$ of input $x$. $X$ is a vector containing feature values taken from a document $d$ (Farzindar and Inkpen, 2015, p. 16). The training instances for such a supervised classification task are document vectors and their corresponding class labels (Farzindar and Inkpen, 2015, p. 16). In the following chapter, some methods for transforming a sequence of words into vectors that can be used by the computer to learn to represent language will be discussed.

### 2.6.1 Vector Space Model

As already mentioned above, a computer cannot simply read a text like we humans do and understand directly what is expressed here. We need to convert the texts and the words inside of them to numerical vectors first. This is achieved by transforming documents into a vector space model of texts, "which maps documents into points in a high-dimensional **feature vector** space, with each coordinate representing the importance of a specific word to the document [...]" (Zhang and Teng, 2021, p. 50). Words are thus represented by coordinates which makes it possible to compare different words to each other by observing their positions (Hellrich, 2019, p. 19). Such a vector space can help us to observe similarities between documents by calculating the distance between vectors (Zhang and Teng, 2021, p. 50). Additionally, vector spaces can be used to perform classification. For classification, we would let the computer find hyperplanes in the vector space that separate the points of different classes from each other (Zhang and Teng, 2021, p. 50).

Let us make this point clearer by introducing the mathematical notation proposed by (Zhang and Teng, 2021, pp. 50-51):

The vocabulary of a document in a vector space, in which each coordinate stands for a word, can be defined as V = $\{w_1, w_2, ..., w_{|V|}\}$. Each word in this vocabulary carries its unique index, but the position in the vocabulary is not important. The goal would be to map all of the text documents into points in the |V|-dimensional vector space with each dimension referring to a word $w_i$ in the vocabulary. The vector representation of a document $d$ would henceforth be:

$$\vec{v}(d) = \langle f_1, f_2, ..., f_{|V|} \rangle \tag{2.3}$$

$F_i$ is used to denote the impact of the vocabulary word $w_i$ to the document $d$. This impact can for example be the amount of times the word $w_i$ appears in the document $d$. As the whole vector space |V| includes several documents, most vector representations can be high-dimensional sparse vectors with a lot of zeros denoting those cases in which the word is in the general vocabulary of all documents, but not in the specific document at question. For instance, *Grand-Duc* 'grand duke' could appear in some documents but not in others resulting in a "0" in those documents in which the word does not occur.

Such a vector space is essential for performing various ML tasks like classification. Interestingly, vector spaces also have a property that allows us to find out how related certain documents are. This is possible due to similar documents being placed close to each other in the vector space (Jurafsky and Martin, 2009, p. 803). This closeness can be measured by calculating the cosine between two document vectors. If both vectors are identical, the cosine will be 1 and if there are no common words, the cosine will be 0 ((Jurafsky and Martin, 2009, p. 803) and (Hellrich, 2019, p. 22)). It should also be noted that sometimes, stopwords, i.e. words with a high frequency like *en* 'a (masculine)' or *eng* 'a (feminine)' are removed from the vocabulary when the documents are converted to vectors, as they do not carry specific sense for a specific document and rather occur in most or all of them (Zhang and Teng, 2021, p. 51).

### 2.6.2 One-Hot Encoding

One of the techniques for feature engineering, i.e. representing the input data in a way that allows an ML algorithm to adequately work with the data (Müller and Guido (2017), chapter 4), is called *one-hot encoding*. *One-hot encoding* is a concept for transforming features to numerical values without attributing an artificial ordering inside (Mertz (2021), Feature Engineering/One-Hot Encoding). Not accrediting any kind of order can be useful in case of categorical variables. In chapter 4.3.4, for instance, the experiment working with categorical features such as Luxembourgish adjectives, adverbs, nouns and verbs is presented. While those are essential for the classification task itself, they do not have any natural order. Adjectives such as *domm* 'stupid', *langweileg* 'boring', *lächerlech* 'ridiculous' and *ustrengend* 'exhausting' are certainly useful for detecting the sentiment in a piece of text, but converting them to numerical values such as 1 - *domm* 'stupid', 2 - *langweileg* 'boring', 3 - *lächerlech* 'ridiculous' and 4 - *ustrengend* 'exhausting' would put them on an (artificial) scale suggesting that they can be ranked. Instead of ordering the different values of a feature, *one-hot encoding* "transform it into multiple features, one for each class value." "The "one-hot" in the name of this encoding indicates that exactly one of these new features will have a one, and the others will be zeros." (both citations from Mertz (2021), Feature Engineering/One-Hot Encoding). This way, non-numerical features can be transformed for working with ML algorithms without given a real ordering to the features at hand. *One-hot encoding* was applied for working with linguistic features, an experiment discussed in chapter 4.3.4.

### 2.6.3 Tf-idf Weighting

Term frequency–inverse document frequency (*tf-idf*) is an approach which gives more weight to words that are important for specific documents in a corpus and that are not present throughout the whole collection of texts (Jurafsky and Martin, 2009, p. 805). It computes a score for the relationship of the frequency of the term $t$ and its inverted document frequency *idf*.

*Tf-idf* is denoted by the following formula

$$tf\text{-}idf_{t,d} = tf_{t,d} \times idf_t \tag{2.4}$$

(Manning et al., 2008, p. 109). The term frequency *tf* is the number of times term t occurs in document *d*. A weight is attributed to $t$ depending on the amount of times it appears in *d*. The subscripts henceforth stand for the term $t$ and the document $d$ (Manning et al., 2008, p. 107). The inverse document frequency *idf* is added as a second factor to the equation to assign a "higher weight to words that occur only in a few documents" (Jurafsky and Martin, 2009, p. 805). This *idf* part is useful, as simple term frequency (*tf*) assigns the same weight to any terms independent from the amount of documents they occur in ((Jurafsky and Martin, 2009, p. 805) and (Manning et al., 2008, p. 108)). For instance, the personal pronoun *ech* 'I', as in *Ech fannen de Bettel mega cool.* 'I really like [Mr.] Bettel.',

is likely to appear in a lot of user comments and sentences and therefore does not help to discriminate one user comment/sentence from another. In other words, it does not help an algorithm to learn a specific pattern better.

*Idf* is given by

$$idf_t = log\frac{N}{df_t} \tag{2.5}$$

with $N$ being the whole amount of "documents in a collection" (Manning et al., 2008, p. 108) and $df_t$ the amount of documents that contain term $t$ (Jurafsky and Martin, 2009, p. 805). Through this calculation, the weight for term $t$ in document $d$ is high when $t$ appears frequently only in specific documents and low when this term is used in many documents of the collection (Manning et al., 2008, p. 109). That way, the significance of words which exist frequently in most documents is lessened (Zhang and Teng, 2021, p. 52).

Applying *tf-idf* to a corpus is useful, as the first step of every NLP problem is to create a vector space model of the texts in the corpus. The documents, in the case of this thesis sentences and user comments, are mapped to a feature space vector "with each coordinate representing the importance of a specific word to the document" (Zhang and Teng, 2021, p. 50). The *tf-idf* scores for each word in a document are thus concatenated to a vector (Manning et al., 2008, p. 109) resulting in similar vectors for user comments or sentences in a corpus that include resembling words. While creating vectors in a vector space for letting various algorithms learn patterns in a data set is not an idea specific to *tf-idf*, it brings the advantage that special weight is given to the most relevant and rare words. Sentences such as *Ech fannen dat Buch immens blöd.* 'I think that this book is really stupid.' and *Ech fannen dat Buch déck flott!* 'I think that this book is amazing!' certainly carry opposing sentiment (*negative* vs. *positive*), but are both about books. Therefore, special attention will be given to the term *Buch* 'book' and the resulting vectors of both sentences will be close in the vector space whereas other words such as *ech* 'I', *fannen* 'to find' and *dat* 'that' are likely the topic of many sentences in such a corpus and will contribute less to the overall meaning of those two example sentences. *Tf-idf* was taken for several experiments on the English, German and Luxembourgish version of the corpus. They are discussed in chapters 4.1.3, 4.1.3.1, 4.2.3, 4.2.3.1, 4.3.3 and 4.3.3.1.

### 2.6.4 Word Embeddings

So far, methods for creating vector representations that use the index in a dictionary to create vectors were discussed. Those vectors certainly make texts "understandable" to a computer. Nevertheless, those encodings come with one important drawback: The semantics of a word get lost (e.g. (Chen et al., 2018, p. 1), (Mikolov et al., 2013a, p. 1) and (Zhang and Teng, 2021, p. 114)). For instance, those kind of vectors do not incorporate the information that *Dësch* 'table' and *Stull* 'chair' are closer in meaning to each other than *Telefon* 'telephone' and *Äerdbier* 'strawberry' but only that they might exist in the vocabulary of a vector space. Word embeddings, the technique which will be presented in this chapter, is, on the contrary, capable of incorporating this information into the vectors.

This follows the distributional semantics intuition coined by (Firth, 1962, p. 11) stating: "You shall know a word by the company it keeps!". To put it differently, words can be considered as similar if the structures underlying the related contexts are somewhat shared (Hellrich, 2019, p. 14).

Incorporating such an idea of the surrounding of a word into a vector was already proposed in the 1980s (Hinton (1986)). Different from the simple vector space approach described above though, word embeddings are opaque (Hellrich, 2019, p. 21), meaning that "their dimensions do not directly correspond to contexts in the underlying data" (Hellrich, 2019, p. 21). Each part of such a word embedding encodes a feature or an attribute of a specific word (Zhang and Teng, 2021, p. 301). The words are thus encoded as points in a high-dimensional vector space, considering to some extend the semantic information of the word in question (Chen et al., 2018, p. 179). Subsequently, vectors of similar words, such as *Fra* 'woman' and *Mann* 'man', will be close together in the vector space (Zhang and Teng, 2021, p. 301). Measures like the cosine similarity and euclidean distance can then be used to compute the similarity between such words (Chen et al., 2018, p. 178).

One of the most popular group of algorithms for word embeddings is the word2vec one proposed by Mikolov et al. (2013a) (see also Mikolov et al. (2013b)). Its input is a large collection of texts that are required to learn semantic connections of words. This knowledge is acquired with the help of neural networks (Mikolov et al., 2013a, p. 2). More precisely, Mikolov et al. (2013a) introduce two separate architectures for learning those distributed representations called CBOW and Skip-gram. Both share a common intuition that learning of the semantics should be carried out by looking at words in a certain window around the target word (Chen et al., 2018, p. 179). CBOW forecasts a word based on the context words around it and Skip-gram, on the contrary, makes use of the context to predict the current word (Mikolov et al., 2013a, p. 5). Both CBOW and Skip-gram perform similarly well with Skip-gram sometimes working a little better (Zhang and Teng, 2021, p. 404). Other popular algorithms for achieving word embeddings are fastText (Bojanowski et al. (2017) and Joulin et al. (2016)) and GloVe (Pennington et al. (2014)).

There are various ways to make use of such a word embedding space for NLP problems. In this thesis, word embeddings will be applied as input to ANNs and to different classifiers. The idea behind this is that one can pre-train word embeddings on large text collections and take this prior knowledge as input to a model (Zhang and Teng, 2021, p. 302). This way a model can have more information about a language than could simply be inferred from the corpus that is used as training input. It should also be noted at this point that another way of working with word embeddings is to randomly initialize the vectors and tune them as part of the training process (Zhang and Teng, 2021, p. 396). As will be discussed further in chapter 4, the ML experiments of this work were performed with the same corpus data in three different versions. The RTL corpus was once taken annotated for sentiment in its Luxembourgish original form and both a German and an English translation of those annotated instances were created. Henceforth three separate word embedding spaces were needed for the experiments.

For the experiment with English word embeddings, Google's Word2Vec model (Řehůřek and Sojka (2010)) that was pre-trained on corpora of the news including about 100 billion

words was used (following Galeshchuk et al. (2019)). The advantage of taking such a large pre-trained model is that many words will have appeared in several contexts making the embeddings stronger than if they were simply trained on a smaller corpus such as the RTL one. The German word embeddings were obtained using a 300-dimensional embedding space trained on German Wikipedia data and proposed by Yamada et al. (2020). For the Luxembourgish word embeddings, the supervisor of this doctoral thesis trained a 64-dimensional Luxembourgish word2vec word embedding model. The experiments with word embeddings are described in chapters 4.1.1, 4.1.1.1, 4.2.1, 4.2.1.1, 4.3.1 and 4.3.1.1.

### 2.6.5 BERT's Word Embeddings

BERT, or *Bidirectional Encoder Representations from Transformers* (Devlin et al. (2019)), is one of the most influential language models at the time that this thesis was written. The general structure of BERT will be described in chapter 2.7.6. Here, the focus lies on the way word embeddings are created for BERT, as this approach is different from the word2vec one presented above. Word2vec, in contrast to BERT, creates word embeddings that are unchangeable (Zhang and Teng, 2021, p. 409), i.e. the embeddings for homonyms such as *Schlass* 'castle' and *Schlass* 'lock' will not be different despite their very different meaning and usage. They are therefore called context-free embeddings (Ravichandiran (2021), Understanding the BERT Model/Basic idea of BERT).

BERT uses contextualized word embeddings. The embeddings generated by this model are determined by the environment. Subsequently, each word obtains a different embedding based on the context ((Zhang and Teng, 2021, p. 409) and Ravichandiran (2021), Understanding the BERT Model/Basic idea of BERT). For the example of *Schlass* 'castle' and *Schlass* 'lock', this would mean that there are two different embeddings due to the different context those words appeared in. In order to achieve this, BERT uses a bidirectional transformer based on Vaswani et al. (2017). It is bidirectional, as it is capable of skimming a sentence from both directions (Ravichandiran (2021), Understanding the BERT Model/Working of BERT). BERT furthermore uses a multi-head attention mechanism to understand the context for each word in each sentence (Ravichandiran (2021), Understanding the BERT Model/Working of BERT). The experiments with *BERT* are described in chapters 4.1.2, 4.1.2.1, 4.2.2, 4.2.2.1, 4.3.2 and 4.3.2.1.

## 2.7 Algorithms Used for Sentiment Detection

For the purpose of the analyses carried out as part of this thesis, several different algorithms for classification were employed, a category of supervised learning introduced in chapter 2.4.1. Classification aims at determining the "label or class for a given unlabeled point" (Zaki and Meira Jr., 2020, p. 467). In more formal terms, classification uses a function M to predict the class label $\widehat{y}$ for an input x. This means that $\widehat{y} = \mathrm{M(x)}$ with $\widehat{y} \in \{\mathrm{c}_1, c_2, ..., c_k\}$ and each $c_i$ being a class label (such as *negative, neutral* or *positive* for the purpose of detecting sentiment in this thesis) (Zaki and Meira Jr., 2020, p. 467).

An ML classification system consists of four parts: First of all, a feature representation for the input is created. Each input observation, such as a sentence, is a vector of features. Second, a classification function computes the estimated class for an input ($\hat{y}$). Third, an ML system uses an objective function for learning, which frequently means reducing the error on training examples as much as possible. Last but not least, an algorithm for optimising the objective function for learning is used (Jurafsky and Martin, p. 2).

In this chapter, the algorithms used for classification will be described (see chapter 4 for their use in practice). Despite their different approaches to assigning a *negative, neutral* or *positive* label to the corpus, they have one thing in common. All work with a vector space model (see chapter 2.6.1) and try to find decision boundaries between the different classes during training (Zhang and Teng, 2021, p. 56). Those boundaries are useful to know which label to attribute to new data instances in the testing stage. Note also that all models discussed here are precisely that: models. Therefore, each and every one of them has drawbacks as well as advantages (Murphy, 2012, p. 24). This is also sometimes referred to as the no free lunch theorem (Wolpert (1996)).

### 2.7.1 K Nearest Neighbor

Like the name suggests, the k nearest neighbor (KNN) algorithm chooses a label for a new instance by checking the most similar data points in the training set and subsequently making its choice based on the majority class of the training examples (Manning and Schütze, 1999, pp. 604-605). Henceforth, KNN always chooses the decision boundary locally (Kantardzic, 2020, p. 134) which is different from the other algorithms discussed below.

In mathematical terms, this nearest-neighbor method can be defined as

$$\widehat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i \tag{2.6}$$

with $N_k(x)$ being "the neighborhood of x defined by the k closest points $x_i$ in the training sample" (Hastie et al., 2017, p. 14). $\widehat{Y}$ is the predicted value of y for new examples not part of the training set. The closeness is measured using a metric such as the Euclidean distance (Hastie et al., 2017, p. 14). To put it in other words, "we find the k observations with $x_i$ closest to x in input space, and average their responses" (Hastie et al., 2017, p. 14). There is no rule concerning how to chose the parameter k. However, picking an odd number such as 3 or 5 is usually a good thing, as it avoids potential ties in majority voting (Kantardzic, 2020, p. 136).

To visualize this, an example of a 15-nearest-neighbor algorithm is plotted (see figure 2.9). The training data on the inputs $X_1$ and $X_2$ is portrayed in a scatter plot. The output class variable G of this simulated data carries either a blue (0) or an orange (1) value with 100 points in each class (Hastie et al., 2017, p. 12). If the value for $\widehat{y}$ is higher than the threshold 0.5, the orange class (1) is assigned and if the value is smaller or equal to 0.5, the blue class (0) is selected. The plot 2.9 shows that the decision boundary separating blue

and orange is not straight which is due to KNN deciding about the decision to be taken locally (Hastie et al., 2017, p. 14). In the case of plot 2.9, this means that the 15 nearest neighbors of a data point are taken to decide about the class of a new data point following the majority vote principle.

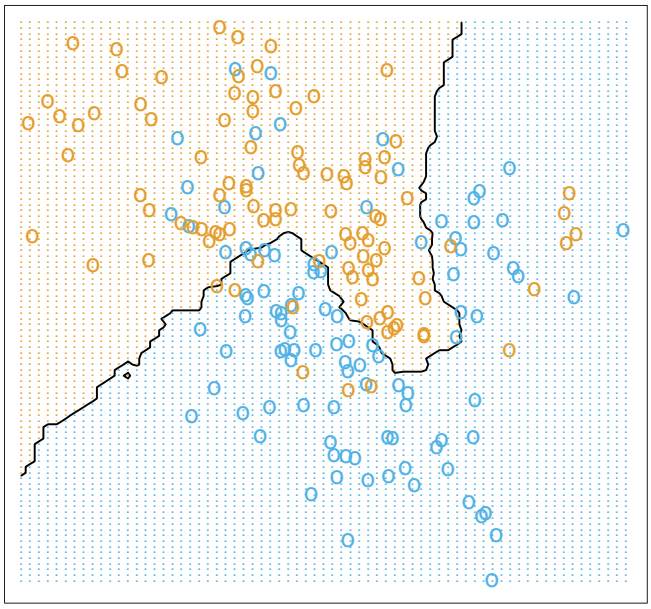


Figure 2.9 – Example of a 15-nearest neighbor classifier (Hastie et al., 2017, p. 15).

### 2.7.2 Decision Tree

A decision tree (DT) is a predictor that "predicts the label associated with an instance x by traveling from a root node of a tree to a leaf" (Shalev-Shwartz and Ben-David, 2014, p. 212). The basic idea of this algorithm is that the classification task is compartmentalized and it is moved through a tree from root down to the leaves to obtain a decision (Marsland, 2015, p. 249). An advantage of such a system is that they are intuitively understandable for humans (Farzindar and Inkpen, 2015, p. 16). At each node on the journey from the root on the top to the leaves on the bottom, a decision on how to move forward is taken depending on either some pre-defined rules or features of x. This leads to a final leaf with the output label (Shalev-Shwartz and Ben-David, 2014, p. 212).

Figure 2.10 shows a simple example of how a DT works following (Shalev-Shwartz and Ben-David, 2014, p. 212). The features taken into account for the classification choice are displayed in red whereas the potential output labels are shown in blue. The goal of this tree is to find out if a papaya is ready to be eaten (label *tasty*) or not (label *not-tasty*). First of all, the color of the fruit is examined as a feature. If it does not lie between a pale green and pale yellow tone, the algorithm will automatically judge the papaya as being *not-tasty*. In classification terms, that means that a *not-tasty* label would be given to the data instance. If this is not the case, the tree moves on to examine the softness feature

of the papaya. In the case of it being soft enough to give slightly to palm pressure, the papaya will be judged as *tasty* and otherwise as *not-tasty*.

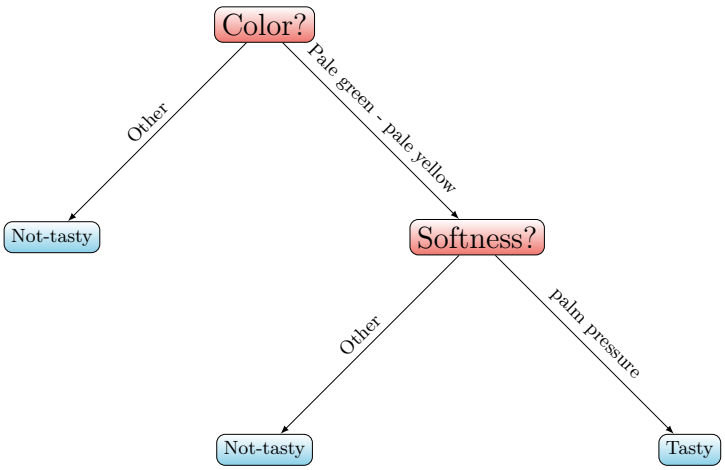



Figure 2.10 – Example of a decision tree (adapted from (Shalev-Shwartz and Ben-David, 2014, p. 212).

### 2.7.3 Random Forest

Random forest (RF) is a method that was first presented by Breiman (2001). Such a forest consists of several DTs (see chapter 2.7.2). Each tree in the forest is made up by taking an algorithm A on the training data S with a random vector $\theta$ that is sampled from a distribution (Shalev-Shwartz and Ben-David, 2014, p. 217). The final prediction, i.e. class label assigned to a data point, is subsequently drawn by a majority vote of all DTs in the RF (Shalev-Shwartz and Ben-David, 2014, p. 217). One advantage that using RF instead of DT has is that it is less prone to overfitting (Shalev-Shwartz and Ben-David, 2014, p. 217), i.e. learning the distribution of the training data very well but not being able to generalize this to new unseen data (Shalev-Shwartz and Ben-David, 2014, p. 16). This is assured by taking a diversity of trees constructed "by sampling a random subset of the attributes at each internal node in the decision tree" (Zaki and Meira Jr., 2020, pp. 575-576) instead of simply putting very similar DTs next to each other.

Figure 2.11 shows a simplified version of the RF algorithm. First, the training data is used and distributed for learning to the different DTs in the forest. Second, the outcome of all DTs is taken into account. Last but not least, the mean of those predictions is taken to come up with a final label. If we consider a sentiment analysis situation, for instance, that has three different potential labels (*negative, neutral* or *positive*), those labeled data points would be given to the different DTs in the RF for training. Subsequently, each tree comes up with one of the three possible labels. Through means of majority voting, one label is chosen and provided as the official output of the RF.

*The random forest visualization was adapted from `https://tex.stackexchange.com/questions/503883/illustrating-the-random-forest-algorithm-in-tikz` [Accessed: 07/13/2021].

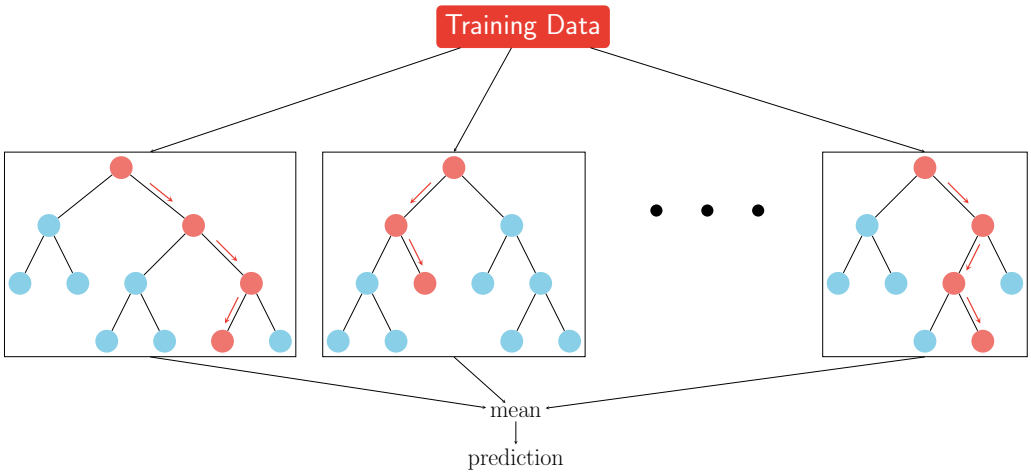



Figure 2.11 – Visualization of a random forest*.

### 2.7.4 Support Vector Machine

Support Vector Machine (SVM) is an algorithm frequently used because of its good performance on a variety of tasks (Farzindar and Inkpen, 2015, p. 16). The most simple case is to take SVM for a binary classification problem, for instance for learning to distinguish *negative* from *positive* classification examples.

For that purpose, "SVM finds a hyperplane that best separates the two classes of training examples, by maximising the distance between the hyperplane and the training examples that are the closest to it" (Zhang and Teng, 2021, p. 57). *Support vectors* denote the data points closest to the hyperplane (Zhang and Teng, 2021, p. 57). SVMs are considered to be maximum margin models, as they aim to maximize the distance between the *support vectors* and the hyperplane. Figure 2.12 visualizes such a simple SVM case with two perfectly separable classes, once denoted by round and once denoted by cross shapes. The hyperplane is portrayed by the solid line in the middle of the graph and the three points situated on the dashed lines are the support vectors.

Unfortunately, data is frequently not as nicely separable into two classes as shown in figure 2.12. For those cases as well as for multi-class problems, the kernel trick is taken that maps the original data points to a high-dimensional feature space in order to create a hyperplane that can clearly separate the data (Zaki and Meira Jr., 2020, p. 533). SVM can be extended to multi-class cases as well ((Zhang and Teng, 2021, pp. 62-63) and (Murphy, 2012, p. 497)) which is useful for cases such as the RTL corpus in which there are three possible labels (*negative, neutral* and *positive*) and not only two.

*This visualization of a support vector machine was taken from Ng, Andrew. CS229 Lecture notes, p. 11. `https://see.stanford.edu/materials/aimlcs229/cs229-notes3.pdf` [Accessed: 07/16/2021].

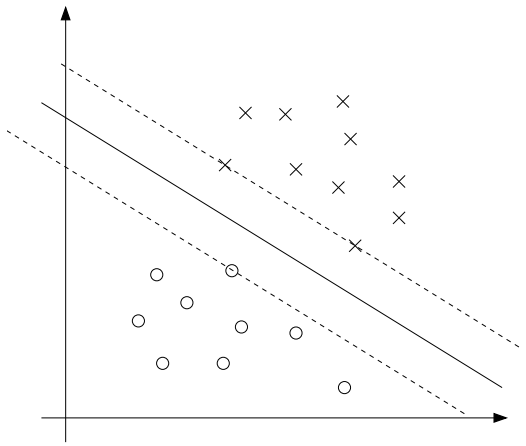


Figure 2.12 – Visualization of a support vector machine *.

### 2.7.5 Logistic Regression

The simplest form of logistic regression (LRM) is a binary classification problem such as training a classifier whether a sentence is *positive* or *negative* (Jurafsky and Martin, p. 2). LRM is frequently a baseline supervised ML algorithm for classification tasks and has a close connection to ANNs (Jurafsky and Martin, p. 1). LRM solves the classification task of determining the probability for either class *negative* or *positive* by learning "a vector of weights and a bias term" from a training corpus (Jurafsky and Martin, p. 3).

LRM fits a sigmoid function (also known as logistic function) to the data (Murphy, 2012, p. 21). The sigmoid function is s-shaped and is defined as

$$\theta(z) = \frac{1}{1 + exp(-z)} = \frac{exp(z)}{1 + exp(z)} \tag{2.7}$$

(Zaki and Meira Jr., 2020, p. 624). It creates an output value between 0 and 1 for any input z making $\theta(z)$ be interpretable as a probability (Zaki and Meira Jr., 2020, p. 624). Plot 2.13 shows the way logistic regression establishes a decision boundary. The idea of LRM can also be applied to classify more than two classes. In that case, the algorithm is called multinomial logistic regression (Jurafsky and Martin, 2009, p. 235).

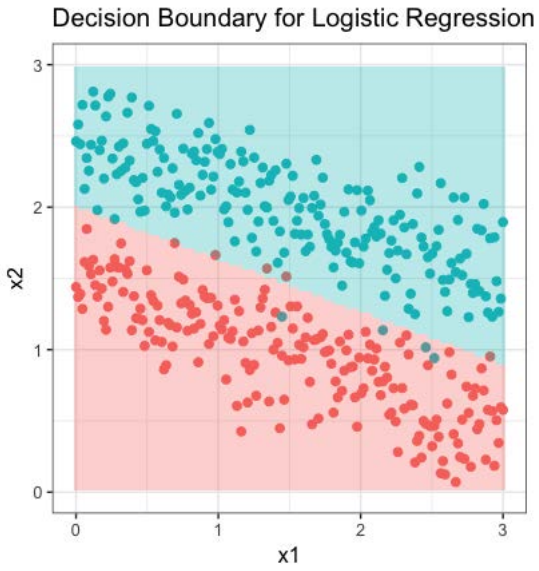



Figure 2.13 – Visualization of logistic regression *.

### 2.7.6 BERT

At the time of writing this thesis, BERT was one of the most influential, if not the most influential, models in the NLP community. The basic idea behind BERT is that one can train a model during pre-training to get a general understanding of the natural language itself, i.e. English, German or Luxembourgish in the experiments of this work. This general knowledge model is then fine-tuned to a specific task such as the sentiment analysis one in this thesis.

BERT is the abbreviation for Bidirectional Encoder Representations from Transformers (Devlin et al., 2019, p. 4171). Its setup is a "multi-layer bidirectional Transformer" (Devlin et al., 2019, p. 4173) based on Vaswani et al. (2017) that works with bidirectional self-attention (Devlin et al., 2019, p. 4173). BERT consists of two parts: one pre-training and one fine-tuning section that are very similar in their setup. While pre-training uses unlabeled data, fine-tuning is then initialized with the parameters from pre-training (Devlin et al., 2019, p. 4173). The idea is that the information about language obtained during pre-training can be carried on to fine-tuning and lead to better results. This is a kind of transfer learning (Ravichandiran (2021), Understanding the BERT Model/Pre-training the BERT model), as knowledge from pre-training is a useful base to build upon during fine-tuning helping to learn as much about language as possible in general before adapting to a specific task in the fine-tuning stage.

During pre-training, BERT carries out two different tasks, a masked language modeling (MLM) and a next sentence prediction one (Ravichandiran (2021), Understanding the

*This visualization of logistic regression was taken from Jun M. Logistic Regression from Scratch in R. `https://towardsdatascience.com/logistic-regression-from-scratch-in-r-b5b122fd8e83` [Accessed: 11/10/2021].

BERT Model/Pre-training the BERT model/Pre-training strategies). The MLM in pre-training (Devlin et al., 2019, p. 4171) is based on the intuition that the context information on both the right and the left side of a word are important for forecasting a word (Zhang and Teng, 2021, p. 412). An example for this would be: *Ech fueren elo an d'[MASK], fir an der Galeries Lafayette akafen ze goen.* 'I am going to the [MASK] now to buy something in Galeries Lafayette.' The masked target word in this case is *Stad* 'Luxembourg city'. If only the left context is considered, it already becomes apparent that the masked word is a location of some sort. However, it remains unknown what the purpose and the precise idea is. Looking at the right context answers this question. "Galeries Lafayette" is a store which only exists in Luxembourg city (*Stad*) in the whole country of Luxembourg. By following through with such an MLM approach, BERT obtains a deeper understanding of the language structure itself than if training was only done considering the right or the left context. The second task in pre-training is the next sentence prediction (NSP) one. It is used to pre-train "text-pair representations" (Devlin et al., 2019, p. 4172 and p. 4174). Having a NSP task as part of the pre-training procedure is helpful due to the fact that it helps the model to learn the relationship between different sentences. This is required to better perform tasks such as question answering or text generation (Ravichandiran (2021), Understanding the BERT Model/Pre-training the BERT model/Pre-training strategies/NEXT SENTENCE PREDICTION).

The BERT model can subsequently be fine-tuned for a wide range of tasks. For this thesis, for example, a pre-trained BERT model was used and fine-tuned for the sentiment analysis task. Fine-tuning is carried out updating the weights of the pre-trained BERT model (Ravichandiran (2021), Getting Hands-On with BERT/Fine-tuning BERT for downstream tasks). Additionally, a classification layer can be added to the pre-trained network before tuning all the parameters (Devlin et al., 2019, p. 4179).

## 2.8 Evaluation Measures for ML and NLP

During development of an ML or NLP system, for instance for a classification task, certain evaluation measures are frequently used in order to objectively determine the quality of the performance. Those measures have crossed over from the field of information retrieval (IR) and can give a first indication whether or not the system that is developed is of sufficient quality (Manning and Schütze, 1999, pp. 267-268). For the development of a sentiment analysis system, for instance, this would mean that the purpose of those scores is to evaluate how well the system learned to detect sentiment classes such as *negative* or *positive* in the data.

In order to understand the evaluation measures better, a contingency table taken from an IR setting is shown in table 2.2 (Manning et al., 2008, p. 143). In this table, true positives and true negatives denote items that were correctly identified by the model and false positives and false negatives those that received an incorrect label during the classification process.

Table 2.2 – A contingency table for evaluation measures.

| | relevant | irrelevant |
|---|---|---|
| retrieved | true positives (tp) | false positives (fp) |
| not retrieved | false negatives (fn) | true negatives (tn) |

Potentially, the most intuitive way of evaluation an ML system is to use a metric called accuracy. The accuracy score denotes the probability of a correct prediction of a classifier (Zaki and Meira Jr., 2020, p. 547). In respect to contingency table 2.2, accuracy is defined as (Manning et al., 2008, p. 143):

$$Accuracy = \frac{\text{tp} + \text{tn}}{\text{tp} + \text{fp} + \text{fn} + \text{tn}} \tag{2.8}$$

Accuracy consequently divides the number of correctly classified observations by the amount of all observations. If the classification system thus is evaluated on 100 instances of which 25 are correctly identified as *positive* and 10 as *negative*, the accuracy of the system would be 0.35, i.e. 35%. While accuracy seems like an intuitive choice for evaluation, it comes with one important drawback, namely that it has a problem working with unbalanced data sets. If a data set consists of 99 *positive* and 10 *negative* observations and all 99 *positive* instances are classified correctly, the accuracy value is 0.91, i.e. 91% despite the fact that all *negative* instances are put into a wrong class.

Another evaluation metric examining data from another angle is precision. Precision is defined as the proportion of all the items that were correctly classified by an algorithm (Manning and Schütze, 1999, p. 268). In other words, it is the part of the retrieved documents that are relevant (Manning et al., 2008, pp. 142-143). In mathematical terms, this means:

$$Precision = \frac{tp}{tp + fp} = \frac{\#\text{ (relevant items retrieved)}}{\#\text{(retrieved items)}} \tag{2.9}$$

If a model classified a total of ten instances into the *positive* class of which six were adequately labeled *positive*, the precision would be 0.6, i.e. 60%.

The third evaluation metric which is applied in ML projects is recall. Recall denotes the "proportion of the target items that the system selected" (Manning and Schütze, 1999, p. 269), i.e. the part of relevant documents that the model picked (Manning et al., 2008, p. 143).

Recall is thus defined as:

$$Recall = \frac{\text{tp}}{\text{tp} + \text{fn}} = \frac{\#\text{ (relevant items retrieved)}}{\#\text{(relevant items)}} \tag{2.10}$$

Going back to the classification example from the precision part, this would mean that the recall value is 0.86, i.e. 86%, if there are seven *positive* instances in total of which one was misclassified as *negative* and six adequately classified as *positive*.

While precision and recall might already give a better intuition of the performance of a model than simply examining accuracy, they still measure different aspects. In order to get a harmonic balanced mean of precision and recall, $F_1$ score can be used which is defined as ((Manning et al., 2008, p. 144) and (Zhang and Teng, 2021, p. 195)):

$$F_1 = \frac{2\mathrm{PR}}{\mathrm{P} + \mathrm{R}} \tag{2.11}$$

The maximum value of 1 for $F_1$ is obtained when both the precision and the recall score are 1 and the minimal value of 0 when either one of them is 0 (Shalev-Shwartz and Ben-David, 2014, p. 207). For the ML experiments described in chapter 4, this $F_1$ score was used in order to show a harmonic evaluation value.

## 2.9 Annotation and Agreement of Corpus Data

As described in chapter 2.7, annotated data is required for training well-performing ML systems. Algorithms use those labeled instances to learn general patterns related to different classes and can ideally then apply this obtained knowledge to unseen cases. The corpus data used for this thesis comes in an unannotated state, i. e. does not include any information required for automatically detecting sentiment (see chapter 3.1). Therefore, all annotations needed for building a sentiment analysis system still need to be done. It is crucial for those annotations to be planned and evaluated thoroughly, as the choices made at this step have an impact on the quality of the annotations collected and therefore later on also on the performance of the algorithms that were trained. The first part of this chapter will be an overview of the typical structure of an annotation process in order to subsequently describe those steps in greater detail. Most of those steps are typical for annotation, no matter for which phenomena the annotation is done. For this thesis, annotation is used to create a corpus of Luxembourgish user comments enriched with *negative, neutral* and *positive* sentiment tags on sentence-level (and also to some extend on document-level). This is shown in more detail in chapter 3.3.

Frequently, annotation projects are structured as follows: First, a corpus is created and guidelines for the required annotation are determined. Second, the annotators are recruited, given the guidelines and complete the task they are asked to do. They can be either experts in the specific field or laymen. Third, the researcher(s) use all the data collected by the annotators to calculate the inter-annotator agreement (IAA). In case of a low agreement score, the guidelines and annotation process is often revised. If the agreement score is of satisfying value, the corpus and its markup is taken to create the gold standard for training and testing an ML algorithm (Pustejovsky and Stubbs, 2013, pp. 105-104).

### 2.9.1 Establishing the Annotation Guidelines

During the first step of this process of creating the corpus and adequate guidelines for annotation, the task of how to frame the general annotation question is crucial. This is important, as having a precise idea of the task helps defining exact definitions and annotation guidelines (Pustejovsky and Stubbs, 2013, p. 51). Mohammad (2016) distinguishes between two main directions for creating sentiment analysis annotation guidelines, the simple sentiment questionnaire and the semantic-role based sentiment one.

The simple sentiment questionnaire approach works with open guidelines asking the annotator whether a *positive, negative* or *neutral* sentiment is expressed (Rosenthal et al., 2015, p. 452). An advantage of this method is that the natural and intrinsic intuition of native speakers is not influenced in any way by providing definitions and examples of what the author of the guidelines understands as *positive, negative* and *neutral*. However, this approach misses one important point, namely how the annotator is supposed to deal with special cases such as sarcasm or mixed sentiment (Mohammad, 2016, p. 174). Also, it is questionable whether *positive, negative* and *neutral* are really precise enough to allow for a high-quality annotated corpus or if not defining those three concepts further will lead to a high dispersion between the annotations made. Even if all annotators have a similar background, it will never be exactly the same as every single person comes with different background knowledge and prior experiences that influence the way they see the world. The subjectivity of an annotator can fluctuate and has to be taken into account when working with annotations. Having a bad day can, for example, lead to a potentially worse mood than having a great day which could have an influence on how *positive, negative* or *neutral* the annotator perceives the instances that are presented to them.

The semantic-role based sentiment questionnaires approach (Mohammad, 2016, p. 174) is more precise than the simple sentiment questionnaires one. For this method, the annotator is asked to find the "target of opinion" and the "sentiment towards this target of opinion" (Mohammad, 2016, p. 174). While this is more explicit than simply asking the annotator to annotate according to their personal idea of what is *positive, negative* or *neutral*, it still comes with limitations. For instance, it is not clear how the emotional state of the speaker should be labeled (Mohammad, 2016, p. 174). If we look at sentences such as "I absolutely loved that game but my sister hated it.", the target of opinion *game* is quickly identified. Interestingly, the sentence includes two sentiments towards said target. Without further specification in the annotation guidelines, it is not clear whether the sentiment expressed by the speaker *I* or the second person *my sister* should be taken into account for the annotation.

No matter how explicit the annotation guidelines, limitations always remain. Even if the researcher or research group establishing the guidelines came up with pages after pages of description and examples, it is highly unlikely that all contexts will be considered or that annotators might interpret certain situations differently. Language is beautifully diverse and cannot easily be captured in all-encompassing rules. Annotation guidelines therefore always remain an approximation to the problem and their depth and cover range depends on several factors such as how much time and effort can be put into the drawing up of the

guidelines and also how the setup of the actual annotation will be. Researcher(s) therefore just have to be aware of those limitations.

### 2.9.2 Choice of Annotators

After having throughoutly planned the kind of annotation approach that should be used (simple sentiment questionnaire vs. semantic-role based questionnaire in this case of sentiment related annotation (Mohammad (2016)), it also has to be considered who will undertake the annotation. One option is to recruit expert annotators with specific domain and/or linguistic knowledge. Another option is to work with crowdsourcing techniques through platforms such as *Amazon Mechanical Turk* (Turk). Recruiting experts comes with the advantage that their prior knowledge can be built upon and that it is a lot easier to train those annotators to follow very specific (linguistic) annotation guidelines than to train laymen. Also, those annotators are hand picked by the researcher making it easy to chose the adequate candidates that posses the desired prior knowledge and intrinsic motivation for the task. If the annotation task can be broken down into easy and short guidelines that do not require prior linguistic and domain knowledge, working with laymen through crowdsourcing platforms is, however, an option that allows to collect a large amount of annotations in a quick and inexpensive way (Pustejovsky and Stubbs, 2013, p. 107). For this PhD thesis, annotations using several methods including crowdsourcing and labeling by trained linguists have been collected. Those approaches are shown in more detail in chapter 3.3.

### 2.9.3 Calculation of Agreement

The next step is then to take all annotations and calculate the IAA to see how much the annotators agree on the annotations performed. In case of a low agreement score, the guidelines are sometimes revised and the annotation is repeated should the project's budget and time frame allow it. There are several possibilities for calculating the IAA scores. As mentioned before, crowdsourced labeling for sentiment on a part of the RTL corpus was carried out (see chapter 3.3). The precise description of how this process was undertaken is described in chapters 3.3.2 and 3.4. For this crowdsourcing, multiple annotators were recruited and each of them performed a different amount of annotations. As Krippendorff's $\alpha$ and Fleiss' kappa are the most adequate agreement score measures for this situation, they will be further presented in the following section building on the discussion in Sirajzade et al. (2020a).

Fleiss' kappa is based on the assumption that it is essential to find out how much agreement above chance was perceived. For this purpose, the ratio between $A_o$ - $A_e$ and 1 - $A_e$ is calculated where $A_o$ denotes the observed agreement, i.e. the amount of agreement perceived between the annotators of an instance, and $A_e$ the expected agreement:

$$k \equiv \frac{A_o\text{-}A_e}{1\text{-}A_e} \tag{2.12}$$

This calculation can result in a value of agreement between -$A_e$/1-$A_e$ (no observed agreement) to 1 (observed agreement - 1). A value of 0 means chance agreement, as the observed and the expected agreement are equal in this case. Furthermore, Fleiss' kappa assumes an independence between the coders' distributions (Artstein and Poesio, 2008, pp. 558-559).

Krippendorff's $\alpha$ takes a slightly different approach. Its coefficient is expressed in terms of disagreement and not in terms of agreement like Fleiss' kappa and other measures (Artstein and Poesio (2008), p.559). It can be used to determine the agreement for any amount of categories, annotators and can also deal with missing values (Krippendorff, 2011, p. 1). The calculation of the coefficient $\alpha$ can be expressed as follows:

$$\alpha \equiv 1\text{-}\frac{D_o}{D_e} \tag{2.13}$$

(Artstein and Poesio, 2008, p. 566). An $\alpha$ value of 0 stands for complete absence of reliability and a value of 1 is a perfect score (Krippendorff, 2011, p. 1). The IAA was calculated with the help of both measures to be able to compare the scores.

To conclude, this section showed the different steps of a typical labeling process of a corpus in NLP. An overview of those steps is shown in figure 2.14. It should be noted that annotation is the backbone of not just sentiment analysis, but also many other NLP tasks. Therefore, it is essential to clearly plan the annotation process and be aware of implications different decisions will have.

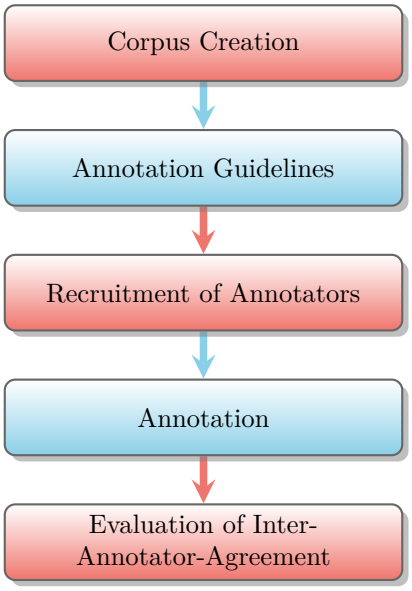



Figure 2.14 – Sequence of steps from corpus creation to evaluation of the annotation.

## 2.10 Important Aspects of the Luxembourgish Language

The corpus used for analysis in this thesis comprises user comments and news articles written in Luxembourgish. Working with this low-resource language comes with special challenges and particularities. In order to better understand those, a brief introduction to the Luxembourgish language and important POS used for the detection of sentiment will be given. This was also already discussed in Sirajzade et al. (2020a) and Sirajzade et al. (2020b) but will be done more extensively in this chapter. It should still be considered that this chapter is only a brief introduction to the field of Luxembourgish and its particularities to allow the reader to better understand the special challenges that occur when working with this low-resource language.

The Grand Duchy of Luxembourg, a multilingual country with more than 590,000 inhabitants (Gilles, in press, p. 3), is home to the biggest population of Luxembourgish speakers. Luxembourgish was declared as the national language of the country in 1984 and is an important part of the national identity (Gilles, in press, p. 5). The other two official languages of the country are French and German. While Luxembourgish is related to German and has its origin in a Central Franconian dialect, it is now accepted and perceived as an independent language (Gilles, 2015, p. 1). It can be used in any spoken and written conversation and code-switching to one of the other official languages of the country is unthinkable if all members of the conversation know Luxembourgish (Gilles, in press, p. 5).

Special to Luxembourgish is that text messages, user comments and other texts in digital format are very often produced in Luxembourgish despite the remarkable fact that the educational system of the country concentrates more on German and French than on teaching the orthographic rules of Luxembourgish (Gilles, 2015, p. 1). This particularity leads to a great amount of variation in texts such as the ones in the corpus data. While this variation might be fascinating from a linguistic point of view, it also creates great challenges to working with Luxembourgish, especially on an NLP level. Several projects have been developed recently to capture or deal with this variation in the language. They show the importance of supporting and preserving Luxembourgish. One important example for archiving linguistic features of spoken instances is the *Schnëssen* app developed at the Institute of Luxembourgish Linguistics and Literatures (University of Luxembourg, Entringer et al. (2020)) to record and preserve the different variation that can be found in the Luxembourgish language. The researchers of the *Schnëssen* app use crowdsourcing to collect as much spoken Luxemburgish as possible to get insights on variation on different linguistic levels of the language (Entringer et al. (2020)). For NLP, several tools have been implemented so far to deal with the peculiarities of working with Luxembourgish. One of those is the *LuNa Open Toolbox* (Sirajzade and Schommer (2019)) which consists of a rule-based POS tagger and a tokenizer. Luxembourgish has also been added to the *spaCy* Python library to allow tokenization, lemmatization and POS tagging (Honnibal et al. (2020)). Furthermore, *spellux* was implemented to automatically correct Luxembourgish texts (Purschke (2020b)). Those tools were important for preparing the RTL corpus for

further analysis using different ML algorithms (see chapter 3.1), as reducing the amount of spelling variation and word forms means decreasing the number of different forms an algorithm has to learn (see chapter 4.3).

Part of the mission of this thesis is to investigate whether or not different kinds of POS and their components have an impact on the way sentiment is expressed and detected. Therefore, the most important POS in question, i.e. verbs, adjectives, adverbs, nouns and negation, will be briefly described in the following paragraph. The idea of this description is not to replace the reading of a grammar book. However, it should give a first insight into the composition of verbs, negation, adjectives, adverbs and nouns to understand their usage as linguistic features for sentiment analysis.

Luxembourgish distinguishes between compound and simple verbs. An example for a compound verb would be *unhunn* 'to wear', such as in *De Pol huet e Mantel un.* 'Pol wears a coat.' and an example for a simple verb would be *schaffen* 'to work' ((Braun et al., 2005, pp. 41-42) and Zenter fir d'Lëtzebuerger Sprooch (2020b)). Most infinitive forms of verbs are composed of a stem and either the ending *-en* (e.g. *schaffen* 'to work') or *-nn* (e.g. *sinn* 'to be') (Braun et al., 2005, p. 41). Verbs also have different kinds of prefixes. Inseparable prefixes are *be-, emp-, ent-, er-, ge-, mëss-, ver-, zer-* as in *bewonneren* 'to admire' or *verstoen* 'to understand' (Braun et al., 2005, p. 43). Separable prefixes are *op-, un-, vir-, hin-, hier-, lass-, of-, zou-, an-, aus-, bäi-, mat-, no-, eran-, eraus-, hannescht-, erëm-, erof-, erop-, fort-, weider-, zesummen-, zréck-, eidel-, fäerdeg-, fest-, fräi-, héich-, voll-* (Braun et al., 2005, p. 45). There are also some prefixes that are sometimes separable and sometimes inseparable: *duerch-, ëm-, ënner-, iwwer-, voll-, widder-* (Braun et al., 2005, pp. 46-47). Interesting about prefixes is that they can completely change the meaning of a simple verb. Examples for this behavior are *falen* 'to fall' - *gefalen* 'to appeal to' and *féieren* 'to lead' - *erféieren* 'to frighten' (Braun et al., 2005, p. 43-44). Additionally, verbs in Luxembourgish can be formed from adjectives (*déck* 'thick' - *décksen* 'thicken') or nouns (*den Hummer* 'hammer' - *hummeren* 'to hammer') (Braun et al., 2005, p. 41).

Luxembourgish has several different ways to express negation. More precisely, *net* 'not', *keen* 'nobody, no', *näischt* 'nothing', *ni, nimools* 'never', *net méi* 'not anymore', *néirens* 'nowhere', *weder ... nach* 'neither ... nor' are the main ways to negate an entity (Braun et al., 2005, pp. 137-139). Interestingly, those can be written in very creative and diverse ways in the user comments that form the main corpus of this thesis. Examples of different ways to write are *net, nët, nett, nit, nitt, nik, nek, nék, nöt, nött, ned, në̈d, keng, keen, kee, kéng, këng, kéin* with some of those being dialectal forms. Negation is especially important for the scope of this thesis, as it can reverse the sentiment of an entity. *D'Wieder ass haut net schlecht.* 'The weather is not bad today.' would, for instance, carry a positive polarity whereas omitting the negation *net* results in a negative polarity of the sentence: *D'Wieder ass haut schlecht.* 'The weather is bad today.'. *Deen Dësch fannen ech guer net schéin.* 'I don't like that table at all.' expresses a negative sentiment while *Dësen Dësch fannen ech mega schéin.* 'I really like this table.' is positive. Those examples show how important the detection and analysis of negation is for determining the sentiment of an instance, as not considering negation can result in accidentally detecting the opposite sentiment.

For adjectives and also nouns, declension exists in Luxembourgish. The form of the adjective therefore changes depending on the case, number and gender of the noun (Braun et al., 2005, p. 98). The most important affixes for Luxembourgish adjectives are *-lech, -eg, -esch, -bar, -sam, -iv, -bel, -haft, -är* and *on-* ((Sirajzade, 2018, p. 200) and (Sirajzade et al., 2020b, p. 161)). Luxembourgish also has superlative forms such as *dee schéinsten* 'the most beautiful' (Braun et al., 2005, p. 102). Looking at the prefix *on-*, for instance, indicates that not only negation but also prefixes can reverse the sentiment of an utterance. This is illustrated by adjective pairs such as *onglécklech* 'unhappy' - *glécklech* 'happy' and *onfrëndlech* 'unfriendly' - *frëndlech* 'friendly'. Also, the study of superlatives in comparison to positive and comparative forms could potentially yield interesting insights into whether or not the usage of superlative forms result in a higher degree of sentiment. The adjective *schéin* 'beautiful'; positive - *méi schéin* 'more beautiful'; comparative - *am schéinsten* 'most beautiful'; superlative, for instance, shows that all three forms carry a *positive* sentiment. Intuitively, a woman who is *am schéinsten* 'most beautiful' is connoted even more positively than a woman who is only *schéin* 'beautiful'.

Luxembourgish nouns, like adjectives, use declension. Suffixes for building nouns are *-ioun, -ung/-ong/-eng, -echt, -(eg)keet, -heet, -age, -er/-ler* (very frequent), *-el, -ement, -enz, -ist, -nes/-nis, -ment, -teur, -esch, -in, -tur/-dur* (frequent), *-mus, -schaft, -(el)chen, -(er)ei, -eur, -tum/-tem , -ert, -trice* (less frequent), *-(e)s, -wierk, -wiesen, -elt, -inne, -ling* (rare) (Sirajzade, 2018, p. 202). As nouns' function in sentences is either the subject or object position, they carry an important part of information on what the sentence is about. It is therefore relevant to examine how much sentiment is carried by those nouns.

A variety of different kinds of adverbs can be found in Luxembourgish, such as adverbs of time (*elo* 'now', *haut* 'today') or adverbs of place (*hei* 'here', *matzen* 'in the middle (of)'). Especially interesting for sentiment analysis are also adverbs such as *gutt - besser - am beschten* 'easily [without difficulty] - better - the best' that express some kind of degree. An overview of the different kinds of adverbs in the Luxembourgish language can be found in (Braun et al., 2020, p. 118-125). Another interesting aspect in this language is the *n-rule* that needed to be taken into account during preprocessing, e.g. with *spellux* (Purschke (2020b)). Generally, this rule states that if a word ends with an *n* or *nn* that is not pronounced, it is omitted from writing as well (see (Zenter fir d'Lëtzebuerger Sprooch, 2019, p. 46-50) for a more detailed description of this rule).

In this chapter, the theoretical foundation necessary for working with the corpus data was elaborated. First of all, the definition of sentiment and the limitations that this term carries in the NLP and other communities was presented. Next, the precise challenges that any sentiment analysis system has to deal with and the different ways to detect the sentiment in a piece of text were discussed. The algorithms for implementing the supervised and DL experiments as well as rule-based methods were introduced. Last but not least, the annotation processes and an introduction of important aspects of the Luxembourgish language and its particularities were given. The next chapter will be dedicated to the elaboration of those theoretical parts in practice.

Chapter 3

# Implementation

This chapter is dedicated to describe the different practical implementations that were developed as part of this thesis and during the *STRIPS* project. They were the essential foundation for the ML experiments which are presented in chapter 4. First of all, the corpus that was the textual foundation of all work and the preprocessing steps necessary for converting it to an adequate format are presented in this chapter. Next follows a description of the different kinds of annotations that were collected, the way the annotation tool to collect crowdsourced annotations was built and an analysis of those annotations using two different inter-annotator measures (Fleiss' kappa and Krippendorff's $\alpha$). Last but not least, the different ways the corpus was split into data sections is shown. Those were later on used for performing the ML experiments (see chapter 4).

## 3.1 Preprocessing of the RTL Corpus

The textual data that is used for the analysis is a corpus of 179,283 news articles and 585,358 comments to those in a time span from 1999 to 2018. The corpus was provided by *STRIPS'* partner RTL Luxembourg and is mostly written in Luxembourgish. It consists of data extracted from their website *www.rtl.lu*. As the data was received in a simple json file, some preprocessing had to be done in order to enrich the corpus with information useful for further analysis. The corpus was transformed from json to XML format and several linguistic preprocessing steps, i.e. tokenization, spelling correcting and POS tagging were carried out. This was most intensively done for the user comments part of the corpus, as this was mostly used for the sentiment analysis experiments. The reason for converting from json to XML was that it was opted for a *Temporal Warehouse* as a backbone for the project (Gierschek et al. (2019)). As part of this infrastructure, an eXist-db XML database (Siegel and Retter (2014)) was used. First of all, after having transformed the data from json to XML, all user comments were split into files containing one month of each year each to ensure an easy human readability. Next, regular expressions were used to prepare the data for further preprocessing steps in LuNa, a corpus tool specifically designed for working with Luxembourgish texts in XML format (Sirajzade and Schommer (2019)). Those regular expressions were applied to declare exceptions for the following

tokenization process. Per default, LuNa allows tokenization at any kind of punctuation mark. However, this would not be useful for many cases. For instance, I did not want to split *:-D* or *d.h.* into *:* - *D* and *d* . *h* . but rather wanted those cases to be considered as one single token. Otherwise, a punctuation mark with a white space would in the end lead to LuNa considering occurrences such as *d* . *h* . as two sentences which is clearly incorrect, as *d.h.* (abbreviation for *dat heescht* 'that means') is usually part of a longer sentence and does not carry its own sentiment. Additionally, POS tagging and setting a sentence id were performed with LuNa. To conclude the preprocessing of the corpus, I iterated through the data, provided a sentence tag with a unique sentence id for each sentence of each user comment (ranging from *aa* for the 2008 sentences to *ep* for the sentences of the user comments written in the 11th month of 2018, see table 3.1), and last but not least used Purschke (2020b)'s spellux tool to correct the diverse variation of spelling found in the corpus. Figure 3.1 shows an example sentence from the corpus in its final preprocessed form and figure 3.2 the preprocessing workflow which was repeated for each sentence in the data. This preprocessed form facilitates an easy access to several parts of each annotated sentence. They come in handy for the various ML experiments which were implemented and whose presentation will follow in chapter 4 as it allows to experiment with various feature combinations.

Table 3.1 – Sentence id per month and year of the user comment file in XML.

| Publication of User Comment | Sentence ID |
|---|---|
| 2008 | aa |
| 01/2009-12/2009 | ab, ac, ad, ae, af, ag, ah, ai, aj, ak, al, am |
| 01/2010-12/2010 | an, ao, ap, aq, ar, as, at, au, av, aw, ax, ay |
| 01/2011-12/2011 | az, ba, bb, bc, bd, be, bf, bg, bh, bi, bj, bk |
| 01/2012-12/2012 | bl, bm, bn, bo, bp, bq, br, bs, bt, bu, bv, bw |
| 01/2013-12/2013 | bx, by, bz, ca, cb, cc, cd, ce, cf, cg, ch, ci |
| 01/2014-12/2014 | cj, ck, cl, cm, cn, co, cp, cq, cr, cs, ct, cu |
| 01/2015-12/2015 | cv, cw, cx, cy, cz, da, db, dc, dd, de, df, dg |
| 01/2016-12/2016 | dh, di, dj, dk, dl, dm, dn, do, dp, dq, dr, ds |
| 01/2017-12/2017 | dt, du, dv, dw, dx, dy, dz, ea, eb, ec, ed, ee |
| 01/2018-11/2018 | ef, eg, eh, ei, ej, ek, el, em, en, eo, ep |

```
<xml>
<sentence id="4dg">
<c id="63" pos="$" sen="4" tagger="0,29">"</c>
<w corr="Äddi" id="64" pos="N" sen="4" tagger="0,07">äddi</w>
<w corr="a" id="65" pos="KO" sen="4" tagger="0,11">a</w>
<w corr="Merci" id="66" pos="N" sen="4" tagger="0,21">Merci</w>
<c id="67" pos="$" sen="4" tagger="0,33">"</c>
<w corr="Kapp" id="68" pos="N" sen="4" tagger="0,34">Kapp</w>
<w corr="Rësel" id="69" pos="N" sen="4" tagger="0,35">Rësel</w>
<c id="70" pos="$" sen="4" tagger="0,31">.</c>
</sentence>
</xml>
```

Figure 3.1 – Example of the preprocessed corpus data.

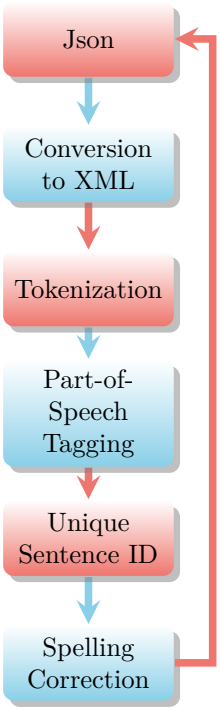



Figure 3.2 – Workflow showing the preprocessing steps for each sentence.

## 3.2 Creating the Annotation Tool

In order to be able to detect sentiment in the corpus, I first of all had to collect sentiment annotations that could later be taken to train the different algorithms. As no annotations for Luxembourgish existed yet and neither did an annotation tool that worked exactly the way that was expected as suitable for the needs of the project, it was decided to create an own one. This tool was first introduced in Sirajzade et al. (2020a) and is described in greater detail here. It was essential for collecting crowdsourced annotations, i.e. sentiment labels collected by different Luxembourgish-speaking people (see chapter 3.3). More precisely, two tools of identical structure were created. One was the selection tool that was leveraged to preselect sentences and one was the actual annotation tool. The preselection was necessary, as I wanted to ensure that all sentences presented to the annotators were actual and syntactically correct. In chapter 2.9, the general structure of an annotation process was introduced as well as the different choices for finding people to provide labels, i.e. annotators (see chapter 2.9.2). Despite a great effort put into the preprocessing stage described in chapter 3.1, the big variety of language found in the corpus made it impossible to correctly identify every single sentence boundary. Having the crowd annotate syntactically correct sentences was, however, an essential prerequisite to ensure the creation of the best gold standard possible that could later on be used for training the algorithms.

The selection tool showed sentences that were randomly retrieved from the whole corpus asking the respective person to judge whether it was a full sentence or not. In case of it being a correct sentence it was marked, saved and later on presented to the annotators using the annotation tool. The annotation tool itself was used by the crowdsourced annotators to label the sentences preselected in the selection tool for sentiment. As both tools are of identical structure and only differ in their purpose of either selecting or annotating data, I will not describe both tools in great detail but will rather focus on the annotation tool in the upcoming paragraphs.

### 3.2.1 Concept of the Annotation Tool

On the back-end side of the annotation tool, an eXist-db XML database (Siegel and Retter (2014)) was applied as a data backbone. It was used to store the annotations of the different annotators and to retrieve sentences for annotation. The annotation tool's back-end was written in *PHP* and *XQuery* was the language to communicate between the database and the annotation tool. The advantage of creating such an annotation tool from scratch was that it was possible to design and implement it exactly the way it was needed for the project avoiding possible pre- or postprocessing steps of the data collected by the annotators. Instead of labor- and time-intensive pre- and postprocessing, the corpus could simply be loaded in XML format into the database. With the help of the selection tool, the sentences that should be annotated could be selected and this selection could then be saved and marked directly into the same database with a simple click. Additionally, sentences could be retrieved from the database for annotation and the sentiment value attributed to the sentence by an annotator was saved to the database with one simple click again.

Figure 3.3 shows the sentence *Sorry, mee dann soll en sech eng aner Platz sichen.* 'Sorry, but then he should search for another place.' (RTL corpus, 07/2009, sentence ID 6151ah) in its final stage after having been selected by one and annotated by two users. The names of the three labeling persons have been made anonymous but are known to the author of this thesis. For the purpose of this work, sociodemographic factors about the annotator such as gender, name and age are irrelevant. However, future work could take this information to study, for instance, whether gender or age have an influence on the way humans perceive sentiment. It would be interesting to examine questions such as *Do younger women annotate the same way as young men do?* or *Does gender have an impact on our perception of sentiment?* in order to understand the concept of sentiment on an even deeper level. While the time and effort that had to go into designing and preparing an own annotation tool should certainly be acknowledged, I nevertheless believe that it was time spent wisely that provided the best result for the purpose of this work.

```
<sentence id="6151ah" selected="yes" selectingUser="jane doe"
value1="negative" username1="john doe"
skipped="yes" skippingUser="max mustermann">
    <w id="74676" pos="" sen="6151">Sorry</w>
    <c id="74677" pos="" sen="6151">,</c>
    <w id="74678" pos="" sen="6151">mee</w>
    <w id="74679" pos="" sen="6151">dann</w>
    <w id="74680" pos="" sen="6151">soll</w>
    <w id="74681" pos="" sen="6151">en</w>
    <w id="74682" pos="" sen="6151">sech</w>
    <w id="74683" pos="" sen="6151">eng</w>
    <w id="74684" pos="" sen="6151">aner</w>
    <w id="74685" pos="" sen="6151">Platz</w>
    <w id="74686" pos="" sen="6151">sichen</w>
    <c id="74687" pos="" sen="6151">.</c>
</sentence>
```

Figure 3.3 – Example of a selected and annotated sentence of the RTL corpus.

### 3.2.2 Procedure of Labeling a Sentence

But what does the annotation tool actually look like? And how could the crowdsourced annotators access it? To access the tool through its front-end, every user received their own personal invitation link leading to a personalized version of the annotation interface. By clicking on this link, the annotator was lead to the welcome page shown in figure 3.4. This page included a personal greeting and a thank you note for participation in the task. All instructions and greetings are given in Luxembourgish without translation to English in the annotation tool, as all annotators were expected to be of native or near-native language fluency. By clicking on the start button, the user was directed to the next page, i.e. the page containing the instructions for annotation. The setup of the page is displayed in figure 3.5. This part includes information about the project itself, who to contact in case of problems or questions on how to annotate. As will be explained in greater detail

in chapter 3.3, an open annotation approach was followed relying mostly on the personal intuition of the annotator and asking to label for *negative*, *neutral* or *positive* sentiment from the perspective of the author of the sentence (Abdul-Mageed and Diab, 2011, p. 23). The instruction page shown in figure 3.5 could be accessed again at any point of the annotation process and was intended to be a first place for help in case of any doubt. Last but not least, the annotator arrived at the page actually used for collecting annotations. The setup for this part is shown in figure 3.6 and includes several important features. On top of the page, the user is presented with the sentence to be annotated in bold. Two additional sentences are given as context, one before and one after the sentence in question. Below the sentence, the annotator can choose between several buttons: They can click on either *negativ* 'negative', *neutral* 'neutral' or *positiv* 'positive' to provide a sentiment label for the sentence presented in bold. Those labels are directly saved in the eXist-db XML database (Siegel and Retter (2014)) together with the username of the annotator (see user *john doe* in figure 3.3). Furthermore, unclear sentences can be skipped using the button *Saz iwwersprangen* 'skip sentence'. If unsure of how to proceed with the annotation, the annotator can also go back to the instructions page by clicking on *Uleedung* 'instructions'. As an incentive to continue annotating, a counter was included on the bottom of the page on top of the general information about the project. In the example in figure 3.6, the annotator *danielagierschek* has annotated 174 sentences [*Schonns 174 Sätz bewäert.*]. It was decided to include this counter as positive feedback hoping that it would motivate annotators to continue as much as possible with the labeling of sentences although no monetary compensation for participation was provided. Additionally, it should be noted that users are free to stop and resume the annotation process at any time by clicking on their personal annotation link. All process is saved and enables continuing whenever the annotator wants.


Figure 3.4 – Welcome message of the annotation tool for the user *Daniela Gierschek*.


Figure 3.5 – Instructions for annotation including the annotation guidelines.

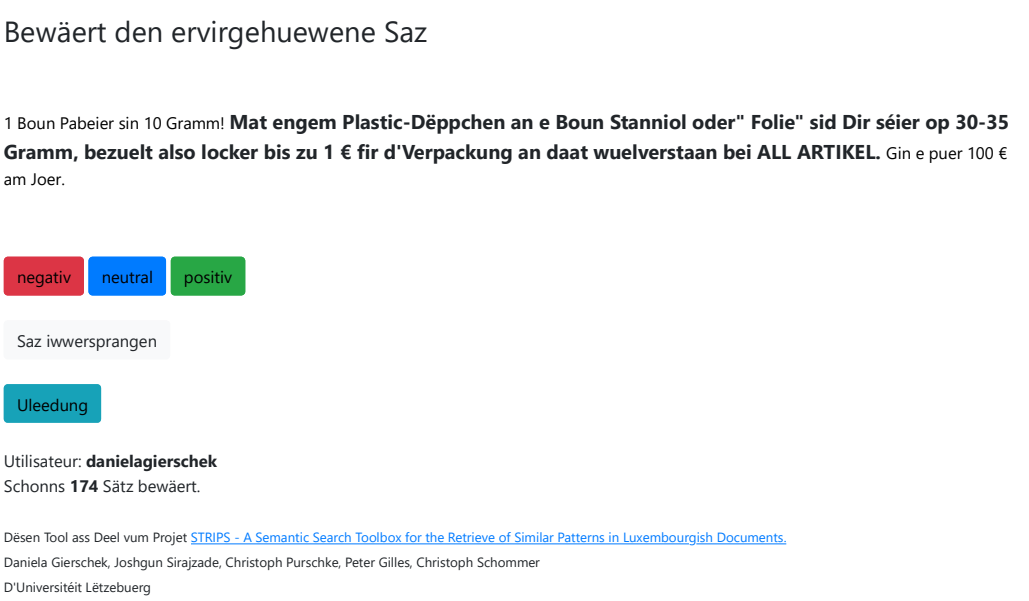



Figure 3.6 – Annotation interface of the annotation tool showing one example sentence in its context.

The tool presented in this chapter was part of a multilayered annotation process which was used to gather as much annotations as possible for training various algorithms to detect sentiment in Luxembourgish user comments (see chapter 4 for the implementation of those and the discussion of the results). Its user-friendly and intuitively usable appearance was used by at least 26 different annotators. * In the upcoming chapter 3.3, the two-fold manual

*The author writes about "at least" here, because part of the recruited annotators were High School students that decided to stay anonymous right from the beginning and therefore all used the same username *MaxMustermann*, a German version of the placeholder name *John Doe*. From now on, those will be considered as one person and it will be written about 26 annotators.

labeling process will first be introduced which was carried out by one trained linguist [the author of this thesis] and the 26+ crowdsourced annotators. Next, a rule-based (lexicon-based) method will be shown which was developed to significantly boost the number of annotations in the corpus.

## 3.3 Annotating the Corpus

The data received from RTL Luxembourg (RTL Luxembourg (2020)) is of interest for studying sentiment as it includes reactions to news articles from a wide range of topics and from Luxembourgish-speaking people with very different opinions and personal backgrounds. In chapter 3.1, it was shown how the corpus was preprocessed in order to facilitate the work. Next, in chapter 3.2, the annotation tool was introduced which was required for collecting a solid base of training instances labeled with *negative, neutral* or *positive*.

Here, the two-fold manual annotation process will be presented followed by an analysis of the quality of the crowdsourced labels with the help of two Inter-Annotator Agreements (IAAs) and the chapter will finish with a discussion of a lexicon-based labeling approach. This lexicon-based annotation method enabled to substantially increase the number of annotated sentences in the corpus. The reason for this multilayered labeling setup was that a large amount of sentences (and user comments) labeled as either *negative, neutral* or *positive* were required which could serve as input for training different ML algorithms. In theory, one approach would have been enough and training could have also been implemented on a small training corpus. Nevertheless, the idea of the project is to not tailor the detection of sentiment to a small subset of the language, but to ideally work towards a system that is able to successfully determine the *negativity, neutrality* or *positivity* of any kind of Luxembourgish statement. For this purpose, working with a large data set comprising many different aspects of the language was essential.

As discussed in chapter 2.9, guidelines for annotation are usually developed before starting the annotation to facilitate the process and to prevent insecurities when choosing between the different labels. The author of this thesis selected to work with open guidelines inspired by Mohammad (2016)'s *simple sentiment questionnaire* and asking the annotator to label from the perspective of the author (Abdul-Mageed and Diab, 2011, p. 23). This approach relies mostly on the intuitive understanding of *negative, neutral* and *positive* by (native) speakers. It was opted for this method as I did not want to infer too much with the intrinsic idea of sentiment that every annotator had within them. Taking this choice came with the potential drawback that too little guidance could lead to confusion as to what is exactly expected of the annotators.

### 3.3.1 Annotation by One Annotator

The manual labelling of sentences (as well as user comments and adjectives for the first part) was two-fold: The first part consisted of annotations carried out by the author of this thesis. One annotator was asked to label user comments, sentences and adjectives for sentiment using *negative, neutral* and *positive* as labels. In total, 448 comments, 2,142 sentences and 1,389 adjectives were annotated (table 3.2). This annotation was done by a trained computational linguist with a special interest in Luxembourgish linguistics [the author of this thesis]. It can therefore be assumed that the intrinsic motivation and understanding of the topic of sentiment is high, possibly leading to a good consistency in the annotations. The annotation was carried out in a three-fold way labeling user comments, sentences and adjectives for their *negativity, neutrality* or *positivity*. This annotation is more fine-grained than all other labeling processes used for the analyses which will be presented later in chapter 4, as it was planned to serve as a starting point for the linguistic feature analysis which was first proposed in Sirajzade et al. (2020b) and developed further in chapter 4.3.4. However, one big disadvantage remains, namely the fact that the labeling was only done by one and not by several annotators. Usually, as explained in chapter 2.9, annotations of a corpus are carried out by at least two annotators in order to be able to create training examples that are, to a certain extend, objective. What does that mean exactly? The purpose of annotation of a corpus is to create examples that ML algorithms can use to learn patterns from. The better the quality and consistency of the annotations, the better the model that can be learned from said examples. Besides, taking labeled instances from several annotators leads to learning more general patterns and not the personal preferences of a single person. Such personal preferences could be, for instance, a bias towards negativity due to the current state of mind resulting in a higher than usual amount of *negative* sentiment labels in the corpus data. Additionally to the lack of objectivity in the annotations, one annotator cannot provide as many (sentiment)-labeled instances as several people in the same amount of time. For those two reasons, it was decided to carry out a second kind of manual annotation, an annotation done by means of crowdsourcing.

Table 3.2 – Annotation values of sentiment by the author of this thesis.

| Level | Number of Annotations |
|---|---|
| User comment | 448 |
| Sentence | 2,142 |
| Adjective | 1,389 |
| | TOTAL: 3,979 |

### 3.3.2 Crowdsourced Annotations

For this second part of the manual annotation setup, it was decided to recruit people from the public, i.e. any kind of human being capable of understanding Luxembourgish and interested in partaking in the project. There was no restriction concerning age, sex or other factors. The only certain requirement was that they needed to be fluent in Luxembourgish and motivated to help. A call for participation was launched with students, both on university and High School level, and also published as an article on RTL Luxembourg's website asking the community for help. This way, a total of 17 participants were recruited bringing the total number of annotators (including those working at the university) to at least 26. Those 26 people received the same open guidelines as the annotator (author of this thesis) in the first phase asking to label from the perspective of the author (Abdul-Mageed and Diab, 2011, p. 23). The labels produced by those 26 annotators were then taken to further analysis that is described in detail in the following chapter 3.4. Up to three annotators labeled the same sentence with a *negative, neutral* or *positive* value. Different from the labeling process by one annotator was that they only had to annotate for sentiment on sentence-level and not on comment- and adjective-level. The reason for this choice was that the annotation tool was designed to be as intuitive and fun as possible and introducing too many tasks at once could have potentially lead to confusion. In future work, it would be interesting to include more fine-grained tasks in the crowdsourcing tool to be able to gather deeper annotations from different people and not only from the author of this thesis' point of view.

Using two different approaches for manually annotating the corpus might, at first glance, seem surprising. I believe that leveraging a two-fold annotation process will be practical for the purpose of this work. The first part of the data comes in handy for doing a more fine-grained analysis, as it does not only include labeled sentences, but also adjectives and whole comments with a sentiment value. The second part is potentially more suitable for large-scale ML algorithms due to its bigger span of annotators making the annotations more objective. In the upcoming chapter 3.4, a closer look at the quality of the crowdsourced annotations collected will be taken.

## 3.4 Inter-Annotator Agreement

After having collected the annotations by the annotators, the data to prepare for training the different ML algorithms was analyzed. As described in chapter 2.9, this step is frequently done after annotations have been gathered to evaluate the quality of the choices made during the labeling process. A first discussion of a part of those results was already published in Sirajzade et al. (2020a) and will be discussed here in greater detail. *

### 3.4.1 Studying the Collected Annotations

The first step of the process was to extract all annotations from the eXist-db database (Siegel and Retter (2014)) in order to get a first overview of the (crowdsourced) annotations that were retrieved using XQuery. The author of this thesis managed to have a total of 26 annotators that labeled 4,195 sentences with a sentiment label. 11 sentences were attributed a "skip", meaning that annotators did not know how to label those examples. In total, 4,206 annotations were collected in more than 2,000 distinct sentences. More precisely, 2,285 sentences were annotated by one person (value one), 1,302 instances by two (value two) and 637 by three annotators (value three). Table 3.3 gives an overview of the amount of annotations collected by value. Of all of those 4,206 labeled instances, 1,854 were *negative*, 1,417 *neutral*, 942 *positive* and 11 *skipped*. Table 3.4 shows the annotations gathered by their sentiment value. Furthermore, a closer look at the agreement and disagreement perceived during annotation was taken. For this purpose, the different scenarios were studied. First, the amount of sentences that carry the same sentiment three times (scenario one) were counted. This was the case in 154 instances. Second, it was examined that in 818 cases, two annotations were equal (scenario two). Equal in this case means that two annotators attributed the same label, for example a *positive* score, to the same sentence. The third and fourth scenario were about the amount of disagreement that was found in the annotated data. For scenario three, the cases in which all annotators disagree (136 sentences) were examined. The last scenario counted the instances in which two annotators disagreed on the sentiment score they gave. For this data, it was the case 712 times. Table 3.5 shows those four scenarios.

*The analysis presented in this chapter was done in January 2020. As the annotation tool remains open beyond this date, a new analysis could lead to different results due to a different amount of annotations.

Table 3.3 – Annotation overview by value.

| Value | Sentiment | Number of Annotations |
|---|---|---|
| 1 | negative | 977 |
| 1 | neutral | 786 |
| 1 | positive | 511 |
| 1 | skip | 11 |
| | | TOTAL: 2,285 |
| 2 | negative | 591 |
| 2 | neutral | 431 |
| 2 | positive | 280 |
| | | TOTAL: 1,302 |
| 3 | negative | 286 |
| 3 | neutral | 200 |
| 3 | positive | 151 |
| | | TOTAL: 637 |

Table 3.4 – Annotation total.

| Sentiment | Number of Annotations |
|---|---|
| negative | 1,854 |
| neutral | 1,417 |
| positive | 942 |
| skip | 11 |
| | TOTAL: 4,206 |

Table 3.5 – Agreement and disagreement.

| Scenario | Number of Annotations |
|---|---|
| 1: all are equal | 154 |
| 2: two annotations are equal | 818 |
| 3: all disagree | 136 |
| 4: two annotations disagree | 712 |

What does agreement or disagreement and those numbers actually indicate? Having a closer look at table 3.3 shows, first of all, that roughly half of the sentences are only annotated one time (2,285 vs. 1,939 if value two and three are combined). This is not necessarily surprising as all sentences were shuffled and proposed randomly for annotation. What is interesting is that no matter if a sentence was annotated one, two or three times, there

was always a tendency towards attributing *negative* labels. Table 3.4, which displays the number of annotations grouped by label, supports this first observation demonstrating that *negative* and *neutral* were a lot more frequent than *positive*. During the implementation phase (presented in chapter 4), it will be checked whether or not this tendency was learned by the algorithms, as unbalanced data sets can potentially influence the learning outcome of training. Table 3.5 furthermore indicates that the corpus data indeed includes cases of agreement. This is important to underline here, because the author ideally would like to have a high amount of agreement cases in the annotations. The reason for this is that high agreement suggests that the annotation guidelines were well-defined. Also, it implies that the annotated sentences passed to the algorithms as training data are consistent which could have a positive influence on the training results, i.e. shown by a high F1 score (see the experiments in chapter 4). By consistent I mean that I would like similar sentences to be annotated the same way. For instance, a sentence such as *Du bass blöd.* 'You are stupid.' should not be labeled once with a *negative* and once with a *positive* label, as this could confuse the algorithms trained on those kind of examples. Whereas full agreement was only achieved in 154 sentences, partial agreement of two annotators was observed for 818 instances. If the instances of (partial) agreement (972) are compared to the sentences for which (partial) disagreement was observed (848), a tendency towards more agreement than disagreement can be noted.

### 3.4.2 Calculating Fleiss' Kappa and Krippendorff's $\alpha$

After this first initial stage of simply counting the occurrences and looking at the output, the IAA was calculated. Fleiss' kappa and Krippendorff's $\alpha$ were used for the measurements as both can calculate the reliability for more than two annotators. Table 3.6 portrays the results of the IAA measurements. A couple of different setups for both Fleiss' kappa and Krippendorff's $\alpha$ were planned to examine whether they have an impact on the degree of agreement or not. First of all, it needs to be noted that three different groups of users that annotated the data were examined. *All* includes everybody that participated in the annotation task, hence 26 people. *Set 1* denotes those people working in the same research group or institute at the University of Luxembourg (9) and *set 2* the 17 annotators that were recruited from the crowd (general public). The reason for examining three different group setups was that I wanted to study whether the degree of linguistic knowledge had an impact on the quality of the sentiment annotation. All annotators part of *set 1* were trained linguists with extensive knowledge of Luxembourgish. Therefore, it can be assumed that theoretically, they should have sufficient knowledge to adequately complete the task. Furthermore, those three groups were used and two kinds of different calculations both for Fleiss's kappa and Krippendorff's $\alpha$ were done. Once the agreement of sentences that were annotated three times (username1, username2, username3) was looked at and once those that were labeled by two people (username1, username2). The intuition behind this two-fold calculation was that higher agreement when only two annotations were compared was expected. Krippendorff's $\alpha$ was computed on a nominal metric as the labels were non-numerical and the usage of other metrics such as ordinal or ratio would thus not have been suitable.

Table 3.6 – Results of Fleiss' kappa and Krippendorff's $\alpha$ calculations by number of annotators.

| Users | Value | Number of Annotators | Result | Agreement |
|---|---|---|---|---|
| all | username1,2,3 | 26 | -0.018 | Fleiss |
| set 1 | username1,2,3 | 9 | -0.025 | Fleiss |
| set 2 | username1,2,3 | 17 | -0.020 | Fleiss |
| all | username1,2 | 26 | -0.022 | Fleiss |
| set 1 | username1,2 | 9 | -0.021 | Fleiss |
| set 2 | username1,2 | 17 | -0.021 | Fleiss |
| all | username1,2,3 | 26 | nominal metric: 0.190 | Krippendorff |
| set 1 | username1,2,3 | 9 | nominal metric: 0.233 | Krippendorff |
| set 2 | username1,2,3 | 17 | nominal metric: 0.185 | Krippendorff |
| all | username1,2 | 26 | nominal metric: 0.200 | Krippendorff |
| set1 | username1,2 | 9 | nominal metric: 0.275 | Krippendorff |
| set 2 | username1,2 | 17 | nominal metric: 0.200 | Krippendorff |

As explained in greater detail in chapter 2.9, a Fleiss' kappa value of 0 denotes agreement purely by chance and a value of 1 perfect agreement between all annotators. A Krippendorff's $\alpha$ value of 0 means complete lack of reliability and a value of 1 is a perfect score. Studying the results displayed in table 3.6 demonstrates that all scores are close to zero. All of Fleiss' kappa's results are slightly below and all of Krippendorff's $\alpha$'s results marginally above zero. This suggests low agreement in the labeled instances. Also, all Fleiss' kappa values as well as all Krippendorff's $\alpha$'s scores are close to each other making an interpretation of which group of users annotated best very hard. Consequently, the hypothesis that being a trained (Luxembourgish) linguist (*set 1*) has a positive impact on the consistency and agreement of sentiment annotation could not be proven a valid claim. Neither could it be proven that the agreement between only two annotations is higher than between three. Those two findings are interesting, as it would have intuitively been expected to be a task which is easier for trained linguists and agreeing between two people should be easier than between three. There are two main possible reasons that could explain this low agreement. Primarily, it should be noted that the annotation guidelines are very open (see chapter 3.3 for more details). For this thesis, I mainly rely on the intuition of (native) speakers and give very little input on how a *negative*, a *neutral* or a *positive* sentence is defined. This could lead to confusion for annotators when having to deal with doubtful cases. For instance, the sentence *Mee schlecht Wieder an an der Nuecht, den Samü Centre ass ennerwee, dann muss den Samü Süden gescheckt gin wann e frai ass an dat ass net em den Eck fir op Maertert.* 'But bad weather and at night, the medical emergency aid services of the center are busy, then the medical emergency aid services of the south have to come if they are free and that's not around the corner from Maertert.' (RTL corpus, 01/2009, sentence ID 2297ab) in the corpus data was annotated three times and each of those with a different sentiment label. Another example for complete disagreement is the sentence *Pëngschden oder Christi Himmelfahrt als Congésdeeg géifen oofgeschaaft ginn.* 'Pentecost and Ascension as holidays would be abolished.' (RTL corpus, 01/2009, sentence ID 21342ab). Both examples show that labeling a sentence with *negative*, *neutral* or *positive* is really not that easy, especially when not a lot of context is given to clearly understand the author of the

statement's true intention when speaking. In this case of annotation, two sentences were provided as context in the annotation tool, one before and one after the sentence to be labeled. This might simply be too little context to be able to really understand if the author's mood in that specific moment is negative, neutral or positive. In future work, the annotation guidelines could therefore be refined to examine whether this has an impact on the quality and consistency of annotations. So far, two techniques which were applied for manually collecting instances labeled for sentiment were shown. Like it was mentioned earlier, I additionally implemented a rule-based (lexicon-based) approach to substantially increase the number of annotations for training the algorithms.

## 3.5 Lexicon-based Annotation

The manual annotation approach, as discussed in chapter 3.3, was two-fold. The first part consisted of a manual labeling process by one trained linguist who provided sentiment annotations for 448 comments, 2,142 sentences and 1,389 adjectives. Crowdsourcing was applied for the second part leading to additional 4,206 annotations on the sentence level. Despite those combined efforts, the data set still remains small. Therefore, it was decided to add an additional step to the annotation process, i.e. a lexicon-based annotation, to substantially increase the amount of training instances that can be used for training the algorithms. This is a crucial step toward setting the foundation of a large annotated training corpus showing the way *negativity*, *neutrality* and *positivity* are expressed in Luxembourgish.

### 3.5.1 Explanation of a Lexicon-based Approach

A lexicon-based approach uses dictionaries containing the semantic orientation of single words or phrases to calculate the overall orientation of a whole document (Turney (2002) and (Taboada et al., 2011, p. 268)). According to (Turney, 2002, p. 417), a semantic orientation can be either positive or negative. A positive semantic orientation means having some sort of good association and a negative one denotes bad associations. *Ofleenung* 'refusal' would, for instance, carry a negative semantic orientation whereas *Kamouditéit* 'comfort' would be of positive semantic orientation. Such an approach goes through all words of a document to note the semantic orientation of each word or phrase. Then, those single orientation scores are used to calculate the overall semantic orientation of the document following predefined rules. The advantage of using such a method is that, after having established a list containing all words and having defined rules for annotation, little additional human intervention is necessary. Theoretically, an annotated corpus of any size can be created that way. It is therefore crucial to plan both the setup of the sentiment dictionary and the labeling rules wisely to arrive at high quality labeled instances of sentiment and avoid unexpected surprises. To illustrate this point further, let us look at one simple, yet important, example. Intuitively, if not expressed in a sarcastic context, *Du bass blöd.* 'You are stupid.' would have been expected to receive a *negative* and *Du bass net blöd.* 'You are not stupid.' a *positive* sentiment label. In order to be capable of

detecting this with a lexicon-based system, having *blöd* 'stupid' marked as *negative* in a lookup list is not sufficient. Additionally, a rule stating that the particle *net* 'not' reverses the sentiment of an adjective if placed directly in front of it should be encoded. Otherwise, a lexicon-based system will be incapable of correctly distinguishing the sentiment expressed in those two sentences and receiving a *negative* annotation in both cases would clearly be undesirable, as it would distort the training instances that are the input for the algorithms.

### 3.5.2 Creating the Luxembourgish PolArt Lexicon

Due to its status as a low-resource language, and like for many other NLP tasks, no lexicon-based approach has been implemented for the Luxembourgish language. The first step of implementing a lexicon-based annotation for Luxembourgish was to find or create a suitable dictionary containing as many words as possible with their respective sentiment score. Unfortunately, no such dictionary had already been established for the Luxembourgish language. As creating a new list from scratch would have been very labor- and time-intensive, the decision to go down a different path was made. An already established sentiment dictionary for German was taken, the PolArt lexicon (Klenner et al. (2009)), and translated to Luxembourgish using a list provided by the Lëtzebuerger Online Dictionnaire [Luxembourgish Online Dictionary] (Zenter fir d'Lëtzebuerger Sprooch (2020a)). The translation was carried out by matching the German sentiment dictionary with the Luxembourgish list containing translations of all entries to German and to French. This list includes 13,671 entries for nouns, 540 for adverbs, 136 for pronominal adverbs, 210 for pronouns and 39,132 for adjectives together with their German translation. This approach to translation does not mean that I consider German and Luxembourgish to be the same language. However, a certain similarity cannot be denied due to a related history of the two languages. For a while, Luxembourgish was even perceived as a dialect of German (Gilles, in press, p. 4). It is therefore assumed that using and translating a German sentiment dictionary to Luxembourgish is more adequate than taking, for instance, an already established English-language sentiment dictionary. This translation process results in a new Luxembourgish PolArt dictionary which consists of 22,105 entries in total. Each entry has several parts, namely first of all the German word, the polarity and its intensity and the POS given in the original PolArt lexicon. Furthermore, each entry also includes a Luxembourgish lemma, POS taken from the original Luxembourgish list and frequently also the plural form(s) or other word forms of the given lemma. Those do not appear everywhere, as not every entry of the original Luxembourgish list included all this information. Currently, the POS information is not used for any analysis, but having both the Luxembourgish and the German POS could potentially be interesting for future work. Imaginable would be to build a hybrid sentiment detection system which would primarily work with the lexicon-based information that was stored in the Luxembourgish PolArt dictionary and take the POS tags as additional features.

Table 3.7 shows eleven exemplary entries of the Luxembourgish PolArt lexicon for *Relief* '(sculptural) relief', *illimitéiert* 'unlimited', *gewot* 'daring', *handfest* 'solid', *irresponsabel* 'irresponsible', *wüüst* 'wild', *eegesënneg* 'stubborn', *üppeg* 'sumptuous', *lukrativ* 'lucrative',

*Aversioun* 'aversion' and *Ofneigung* 'aversion'. There are several interesting factors in this lexicon which are worth pointing out. First, there are two different entries for the *GLOSS_GERMAN Abneigung* (Aversioun - Ofneigung). This is not unusual and a recurrent phenomenon in the lexicon. The reason for this is that the original Luxembourgish list does not summarize those in one column but rather has a separate entry for *Aversioun* and for *Ofneigung*. For now, this factor is not considered when working with the Luxembourgish PolArt lexicon for providing sentiment labels for the corpus. The system that was built just takes the first entry it finds and uses this to determine the sentiment. This could certainly be improved in future versions of the approach. Second, it can be noticed that on the sentiment level (denoted as "polarity" in table 3.7), PolArt does not only distinguish between *negative* (NEG), *neutral* (NEU) and *positive* (POS) but also notes different intensities of said sentiment. This intensity ranges from 0.5 up to 5. * Besides those three labels, the dictionary also includes *INT* and *SHI* as additional polarity categories. *INT* stands for "intensifier" and can amplify the sentiment of a given word. Examples for such an intensifier are *vill* 'much' and *manner* 'less'. *SHI* denotes "shifter", i.e. all those words that can invert the polarity of an instance. Examples for shifters are *manner* 'less' and *keen* 'none, no, no one, nobody'. Such shifters are important to mark, as they would usually reverse positive words or whole sentences to carry a negative sentiment.

Table 3.7 – Eleven random examples extracted from the Luxembourgish PolArt lexicon.

| GLOSS_GERMAN | Polarity | POS_polartlexicon | LEMMA | POS_LOD | PLURAL | PLURAL.1 | WORD |
|---|---|---|---|---|---|---|---|
| Abbildung | NEU=1 | nomen | Relief | NOUN | Relieffen | Relieffer | / |
| unbegrenzt | POS=0.7 | adj | illimitéiert | ADJ | / | / | illimitéierter |
| gewagt | NEG=0.7 | adj | gewot | ADJ | / | / | gewoten |
| stichhaltig | POS=1 | adj | handfest | ADJ | / | / | handfesten |
| unverantwortlich | NEG=1 | adj | irresponsabel | ADJ | / | / | irresponsabelsten |
| wüst | NEG=0.5 | adj | wüüst | ADJ | / | / | wüüstst |
| eigensinnig | NEG=0.7 | adj | eegesënneg | ADJ | / | / | eegesënneger |
| üppig | POS=0.7 | adj | üppeg | ADJ | / | / | üppegsten |
| lukrativ | POS=0.7 | adj | lukrativ | ADJ | / | / | lukrativer |
| Abneigung | NEG=0.7 | nomen | Aversioun | NOUN | Aversiounen | / | / |
| Abneigung | NEG=0.7 | nomen | Ofneigung | NOUN | Ofneigungen | / | / |

### 3.5.3 Labeling for Sentiment on Word Level

The Luxembourgish PolArt lexicon which was created is, of course, already interesting in itself and gives insights about the *negative, neutral* or *positive* characteristic of a single word token. While the final goal of the creation of this lexicon was to obtain a sentiment label for the whole sentence, it was first of all used to annotate all of the data for sentiment on word level. Crucial for this part was not only the new Luxembourgish sentiment dictionary of 22,105 entries but also the preliminary work of using Purschke (2020b)'s spellux tool to correct the spelling of the corpus data. This smoothing of spelling variation was important due to Luxembourgish texts often being written in very diverse ways which do not always follow the official orthographic rules of the time. As explained in chapter 2.10, one of the

*(Klenner et al., 2009, p. 237) write that the polarity strength is still in development and ranges from 0 to 1. When this analysis was carried out in October 2020, sentiment strength had been extended to scores from 0.5 to 5.

reasons for this creativity is that the educational system focuses more on the other two official languages of the country, French and German, than on Luxembourgish. Correcting this large variety in spelling was helpful, as the Luxembourgish PolArt lexicon follows the orthographic rules of Luxembourgish and does not include entries for all sorts of possible spelling variants that speakers could have potentially come up with. An example for this variation in spelling is negation forms. The official form for the English particle *not* is *net* in Luxembourgish which could be written as *net, nët, nett, nit, nitt, nik, nek, nék, nöt, nött, ned, në̈d* in the corpus. The indefinite pronoun *no, none* should be correctly written as *keen, keng or keent* in Luxembourgish. However, it could also appear as *kee, kéng, këng, kéin* in the corpus. After having mapped as many spelling forms as possible to their orthographically correct form with spellux (Purschke (2020b)), I consequently used the Luxembourgish PolArt sentiment dictionary to iterate through each of the user comments files of the RTL news article and user comments corpus used for this thesis. That way, 702,122 *negative*, 1,317,683 *neutral* and 955,107 *positive* words as well as 756,616 *shifters* and 348,482 *intensifiers* were labeled.

### 3.5.4 Moving from Sentiment of the Word to Sentence Sentiment

Subsequently, the annotation on word level was used to calculate an overall sentiment on sentence level to obtain as many training examples as possible for the algorithms. I iterated over all sentences in each user comment of the corpus and counted the word level sentiment annotations in each sentence. The idea of this labeling process was to see sentiment on a scale. If the output of the computation was equal or more than three, the sentiment of the sentence was *positive*. In case of a score of minus three or more, the sentence was to be labeled *negative*. Last but not least, all sentences with a score between those were tagged with a *neutral* label. How did I end up with this final score that lead to the annotation on sentence level? First of all, it should be noted that sentiment in this thesis is seen as being on a range from *negative* (negative numbers) to *positive* (positive numbers). Neutrality is a concept that lies between *negative* and *positive* on this scale. It can thus respectively shift an initial *negative* or *positive* sentiment towards 0. To account for the different behavior of the shift in sentiment in *positive* and *negative* cases, two formulas were first of all implemented and a smoothing approach shown in figure 3.7 was used. Formula one subtracts the amount of *neutral* annotations from the number of *positive* ones found in a sentence. Formula two respectively adds the number of *neutral* annotations to the amount of *negative* labels in the sentence. Subsequently, smoothing was carried out. If the output of the first formula was a negative number, it was changed to zero and if the output of formula two was a positive number, it was also changed to zero (see figure 3.7). This act of smoothing was necessary to adequately be able to calculate a sentiment for *neutral* sentences. For instance, a sentence that contains three *neutral*, one *positive* and one *negative* word would be expected to be labeled as *neutral* by the algorithm. If no smoothing is used, however, the result of the calculation in this case would be minus 6, hence resulting in a *negative* sentiment label.

**Algorithm 1:** Calculation of the sentiment score of the sentence.

```
first = positive - neutral ;
second = neg + neutral ;
if first < 0 then
    first = 0 ;
if second > 0 then
    second = 0 ;
score = first - second ;
```

Figure 3.7 – Calculation of the sentiment score of the sentence.

Furthermore, the occurrence of *intensifiers* and *shifters* in a sentence were taken into account (see figure 3.8). The output of the initial calculation presented in figure 3.7 was used as input to this second operation shown in figure 3.8. If at least one word of a sentence was labeled as *intensifier*, the initial score of the algorithm presented in figure 3.7 was multiplied by the number of *intensifiers* of the sentence. In case of the occurrence of one or more *shifters* in the sentence, the initial score was multiplied by the negative number of *shifters* of the sentence. The reason for multiplying by minus *shifter* was that *shifters* are mostly negation particles or indefinite pronouns such as *keen* 'none (can also be translated with no, no one, nobody)' that are used in language to reverse the sentiment of an instance. Last but not least, the output value of those two algorithms was then used to decide about the overall sentiment the sentence was to be labeled with. A score of equal or greater than +3 lead to a *positive* sentiment label, a score of less than or equal -3 to *negative* and any score between those numbers to *neutral*. Looking at an example from the corpus makes this calculation clearer: The sentence *Eis Enseignaten verbrengen en enormen (INT) Deel (NEU) vun Zeit (NEU) hirer mat Verbesseren an vill (INT) Schüler (NEU) werfen herno just e flüchtege Bleck op (POS) hir verbessert Kopie.* 'Our teachers spend a lot of their time grading and many students just briefly look at their graded exam.' (RTL corpus, 04/2009, sentence ID 275ae) was labeled with two intensifiers (*INT*), three neutral words (*NEU*) and one positive (*POS*). The first part of the calculation is shown in figure 3.7 and leads to 0 in this case. As a next step, the existence of the *intensifiers* in the labeled instance is taken into account resulting in an overall value of 0. Subsequently, the overall label of this sentence is *neutral*. On a side note, the *POS* label of this example sentence shows one of the difficulties that are part of working with automatically translated sentiment dictionaries. *Op* can either be an adverb or a preposition in Luxembourgish meaning either 'open' or 'on'. In this sentence, the *op* is a preposition. In consequence, it seems unusual to label with a *positive* value. This feature can easily be explained when studying the translations in the Luxembourgish PolArt. This sentiment dictionary has only one entry for *op* translating the Luxembourgish adverb "op" with the German adjective *offen* 'open'. It certainly makes sense to assume that the German adjective *offen* 'open' can carry a *positive* sentiment, as it can describe different things such as the character of a person. The preposition *op* 'on', however, does not contribute strongly enough to the sentence meaning to be an individual sentiment carrier. It could therefore certainly be beneficial to include POS tags as part of

the lexicon-based labeling process in future work. In total, 7,903 *negative*, 105,484 *neutral* and 9,643 *positive* sentences were labeled, hence significantly increasing the number of labeled sentiment annotations in the RTL corpus.

**Algorithm 2:** Impact of shifter and intensifier on the labeling of sentences.

```
value = 0 ;
if intensifier != 0 then
    value = score * intensifier ;
else
    value = score
if shifter != 0 then
    value = score *-shifter ;
else
    value = score
```

Figure 3.8 – Impact of shifter and intensifier on the labeling of sentences.

### 3.5.5 Challenges of Working with a Lexicon

As described above, using a lexicon-based approach for annotation resulted in a large increase of sentences in the corpus that were enriched with a *negative, neutral* or *positive* sentiment label. The advantage of this is that the amount of training examples as input for a classification system can be substantially increased without relying on human annotators. However, it should not be forgotten that using a lexicon-based approach also comes with disadvantages that require attention. One of those is that lists are only as good as they were made to be. While creators of sentiment dictionaries do their best to include as many words as possible, it is impossible to include every possible word there is that could potentially carry some sort of sentiment. Also, lists are not good at considering context although this would often be very useful for grasping the sentiment intended by the speaker. A Luxembourgish word like *flott* could, for instance, be used as an adjective meaning 'nice, smart, pretty' in English. Depending on the context, *flott* would be used by speakers to express *positive* sentiment (*O flott! Elo hunn ech endlech méi Pai kritt.* 'Oh nice! Now I've finally received more salary!') or *negative* sentiment (*Flott ... Elo ass hie scho rëm ze spéit komm ...* 'Great ... Now he is late again ...'). *Flott* can also be a noun which would translate to 'fleet' in this case. The adjective *flott* and the noun *Flott* thus are homographs meaning different things and carrying different sentiments. For a human being capable of reading Luxembourgish, understanding the difference between *flott* and *Flott* should usually not be a problem due to their capability of understanding and incorporating context in the reading process. A lexicon in its basic form, however, does not have any context information that could help seeing the different possible meanings of *flott*. If the sentiment dictionary contains only one entry for *flott* with, for instance, a *positive* sentiment score, all three of the above mentioned examples would be labeled as *positive* if found in a corpus although the second example sentence clearly intends a *negative* sentiment. This example illustrates the limitations of only using a lexicon or sentiment dictionary to annotate a

corpus in an unsupervised way on word level. Besides, the sentiment dictionary could only find those words in the corpus that could be adequately detected and corrected by the spellux tool (Purschke (2020b)) or those that were already written correctly in the first place. As (Purschke, 2020a, p. 6) noted, there are still 680,300 unique words in the RTL corpus that spellux cannot normalize due to a number of different reasons. The lexicon-based approach for this analysis relies heavily on spellux. All words that cannot be corrected are likely to not be found in the sentiment dictionary leading to less word-based annotations for sentiment in the corpus. This has a direct impact on the number of sentences which were obtained carrying a *negative, neutral* or *positive* tag, because the word annotations were the input to the calculations shown in figure 3.7 and 3.8. The rules defined here were created after intensive and careful examination of the corpus to carry out this unsupervised annotation of larger syntactic units. Nevertheless, rules are always static and cannot incorporate every possible case of language use. It is therefore highly likely that not every single sentence of the corpus was adequately annotated with the right sentiment label.

The question hereby is: How can an unsupervised or semi-supervised annotation be made better? Possibly, incorporating context information could be helpful to be able to differentiate between the different usages of a word or to be able to see if the usage is intended in a literal or in a sarcastic way. One possible way of doing this would be to use word embeddings (Mikolov et al. (2013b)). This method converts words to vector representations that are capable of understanding context to some extend. For instance, (Mikolov et al., 2013b, p. 2) showed that the addition of vectors such as the ones for "Germany" and "capital" result in a vector close to the one for "Berlin". Combining a lexicon-based approach with word embeddings for context could be an interesting starting point for future research, due to context information possibly being able to improve the adequate detection of sentiment in difficult cases such as when sarcasm is expressed.

### 3.5.6 Overview of All Annotations

To sum up, several annotation procedures were used in order to provide the first Luxembourgish corpus enriched with sentiment labels. They are portrayed in table 3.8. First, one linguistically trained annotator labeled 2,142 sentences, 1,389 adjectives and 448 comments for sentiment. Second, 26 annotators from the public collected a total of 4,206 sentiment labels for sentences. Third, a lexicon-based approach was implemented and ended up with 123,030 labels on sentence- and 4,080,010 labels on word-level. While a certain amount of annotations were collected using very different approaches (single-person annotation, crowdsourcing, lexicon-based annotation) which lead to a large amount of labeled instances, it should not be forgotten that a large amount of annotations does not necessarily equal to a lot of high quality labels. The usage of those annotations as gold standard, i.e. training data for the ML experiments, needs to be carefully monitored in order to understand as well as possible the impact those different approaches might have on the learning process of an algorithm. Accordingly, it was opted to see the annotations collected as belonging to different data sections and all of the ML experiments presented

in chapter 4 were run several times with different parts of the corpus. Those different data sections and what they include exactly will be presented in the following chapter (3.6).

Table 3.8 – Number of all annotations collected with all different labeling methods.

| Type of Annotation | Number of Annotations |
|---|---|
| Sentences (1 annotator) | 2,142 |
| Adjectives (1 annotator) | 1,389 |
| Comments (1 annotator) | 448 |
| Sentences (crowdsourcing) | 4,206 |
| Sentences (lexicon-based) | 123,030 |
| Words (lexicon-based) | 4,080,010 |

## 3.6 Description of the Sections of the Corpus

So far, several approaches have been described that were used to annotate the data set that was then taken and applied to various ML experiments in different setups. The results of those experiments will be described in chapter 4. Before diving right into the different techniques that were applied to the data set, I need to have a closer look at the distribution of sentiment in every part of the annotated corpus.

### 3.6.1 Reasons for Splitting the Corpus into Segments

The explanation of why it was opted for this close study is that very different approaches were used to augment the amount of labeled data instances that could be applied as input to training various ML algorithms. Those different techniques lead to a large amount of labeled sentences (as well as comments in one part), but taking such diverse paths can potentially have an impact on the quality of the labels. One part of the corpus consists of sentences (and user comments) that were hand-labeled for sentiment. The quality of this segment could be checked by calculating the IAA for the crowdsourcing part and by following the hypothesis that my intrinsic motivation was high enough to annotate as well as possible for those sentences and user comments which I annotated. The second part contains labels that were collected with different automatic means. This part is bigger in size than the first one and checking the quality is a lot harder, as reading and manually judging more than 123,000 sentences (see table 3.8) is a very time-consuming and labor-intensive task. To take those differences into account, the corpus was split into different segments and all ML experiments were executed on those parts separately.

The different annotations were collected by the following techniques: I (see chapter 3.6.2.1) and crowdsourcing (see chapter 3.6.2.2) provided hand-labeled annotations and the lexicon-based method (see chapter 3.6.2.3) lead to sentences labeled for sentiment following specific predefined rules. Additionally, two deep learning (DL) approaches implemented in the

*STRIPS* project lead to two more kind of labels for the data (also see chapter 3.6.2.3). The first DL annotation was implemented using Luxembourgish word embeddings and my hand-labeled annotations as training data and a Recurrent Neural Network. The second labels worked with Luxembourgish word embeddings, my hand-labeled annotations, the crowdsourced annotations and spelling-corrected word tokens received through the spellux tool (Purschke (2020b)) as training input for the Recurrent Neural Network. Despite various differences in how the labeling process for all those different corpus sections took place, the output always remained the same, i.e. labels of either *negative, neutral* or *positive* manner. No further fine-grained labels such as determining the degree of *positivity* or *negativity* were undertaken. However, it should be noted that this would be an interesting starting-point for future research, as simply putting statements into *negative, neutral* or *positive* categories is very frequently far from real-life situations.

Applying such different approaches, like the ones shown here, certainly leads to a larger amount of annotations than could be achieved only working with human annotators. While this is an advantage and gives more training data to train the algorithms, it should not be forgotten that it also leads to very different kinds of annotations, as some originated solely from human annotators (crowdsourcing and those labeled by the author of this thesis) and some from ML/rule-based processes (*STRIPS* project, lexicon-based approach). Therefore, it was decided to carry out all experiments in a four-fold approach to grasp the differences due to those disparate methods of labeling. The different segments of the data that were worked with will be further described in the following chapter (3.6.2).

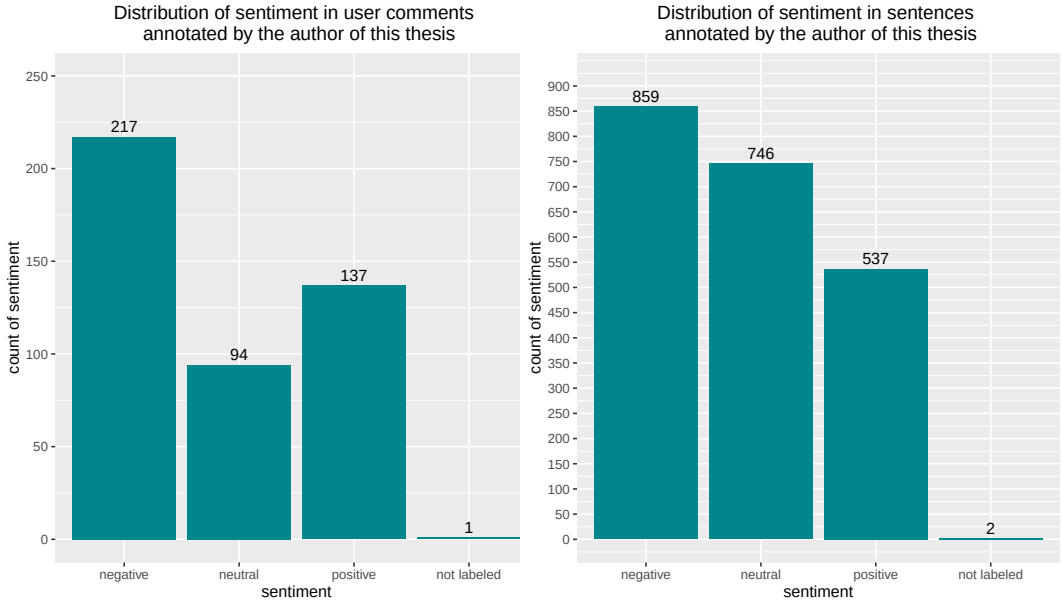
### 3.6.2 Distribution of Sentiment in the Different Sections of the RTL Data

Four separate segments in the annotated RTL corpus were defined and later on taken as input for the training of the different ML experiments presented in chapter 4. In order to get a better overview of what they include, it will first be outlined how each corpus segment is defined and how many *negative, neutral, positive* and *undecided / unlabeled* instances each includes.

#### 3.6.2.1 Data Sections Annotated by the Author of the Thesis - Corpus1 and Corpus2

The first data set consists of my annotations (see chapter 3.3.1). 449 comments, 2,142 sentences and 1,389 adjectives were labeled with a *negative, neutral* or *positive* sentiment. This part of the data set received the most fine-grained annotations, i.e. is the only section of the data that did not only get labeled on sentence-, but also on comment- and on word (adjective)-level. The corpus section containing my sentiment annotations on user comment level are denoted as *corpus1* and the sentences annotated with a sentiment label as *corpus2*. This more detailed annotation was done due to the decision to use those corpus sections as a starting-point for the linguistic feature analysis presented in Sirajzade et al.

(2020b) which was further explored in chapter 4.3.4. Having had those corpus segments only labeled by one annotator comes with positive and negative consequences. First of all, it should be underlined that the annotator is the author of this thesis. As I am personally involved in the project and benefit the most from this sentiment corpus, it can be assumed that my intrinsic motivation is very high. Ideally, this should lead to very consistent annotations. However, one important fact should not be forgotten: When only annotated by one annotator, there is no objective criteria such as the calculation of the IAA described in chapter 3.4 to measure the quality of the annotations collected. This is clearly a downside, because algorithms should be trained to learn general patterns of sentiment expression and not be biased towards the way a certain person perceives *negative, neutral* and *positive*. Figure 3.9 shows the distribution of sentiment in the user comments (on the left) and the sentences (on the right) annotated by the author of this thesis. Both the user comments and the sentences include a majority of *negatively* labeled instances followed by *positive* (for the comments) or *neutral* (for the sentences) annotations. Each data set has a small number (one respectively two) of non-labeled data points likely resulting from a mistake during the annotation process such as accidentally not filling in the value *negative, neutral* or *positive* while labeling.




a User comments.

b Sentences

Figure 3.9 – Distribution of sentiment in the user comments (left, *corpus1*) and sentences (right, *corpus2*) annotated by the author of this thesis.

### 3.6.2.2 Crowdsourcing Corpus Section - Corpus3

The second part of labels was provided by 26 annotators (see chapter 3.3.2) leading to a total of 4,206 labeled instances. Out of those, 783 were annotated by three annotators *. Only those were used to implement the upcoming experiments described in chapter 4 to ensure as much objectivity as possible. This corpus section will be referred to as *corpus3* for the rest of this thesis. While this choice of only considering sentences annotated three times comes with the downside that it omits quite a few labeled sentences from the potential training set, I still believe it to be the adequate method, as it is the best way to bring objectivity to this data section. With two or three labels for one and the same sentence, the IAA can be calculated (see chapter 3.4) and henceforth the quality of the annotation can be measured. Contrary to the first approach, the crowdsourced annotators were only asked to come up with sentiment labels on sentence-level which is why no annotations on comment- or adjective-level were collected.

An overall sentiment for each sentence was computed by looking at the three sentiments given by the annotators for each sentence. That way, one label per sentence was obtained which was essential for training the ML algorithms. If at least two sentiments were of one kind, this sentiment was attributed as the overall sentiment of the sentence. For instance, a sentence which was labeled as *positive* by two annotators and as *neutral* by the third one received the overall sentiment score *positive*. An instance with a *positive*, a *negative* and a *neutral* sentiment annotation would be marked as having no overall agreeing label provided during annotation (*undecided* in figure 3.10). Figure 3.10 displays the distribution of the aggregated sentiment annotated by three people from the crowd for each sentence. This distribution shows, similarly to the one presented in figure 3.9, that *negative* sentences are most present in the data set. The smallest group in the aggregated sentence labels obtained from the calculation is the *undecided* one.

*Those were the sentences annotated three times until 01/28/2021. As the annotation tool remained open beyond this data, this number is subject to change at a future point in time.

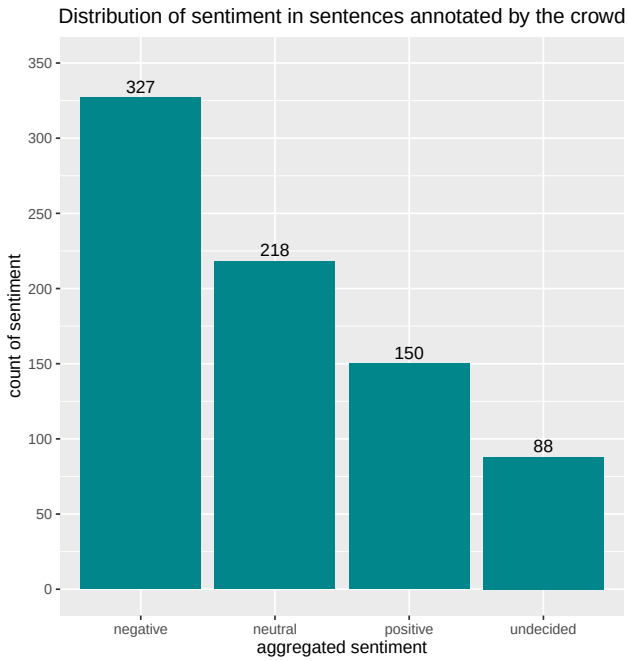



Figure 3.10 – Distribution of sentiment in sentences annotated by the crowd (*corpus3*).

### 3.6.2.3 Lexicon-based and Neural Network Corpus Section - Corpus4

For the third part of the data setup, all sentences saved in the eXist-db database were extracted and were annotated with a lexicon-based label (see chapter 3.5) and with a label induced from the two DL methods applied in the *STRIPS* project (see chapter 3.6.1). This third part contains a total of 113,589 annotated sentences making it the largest contribution to the labeled instances. Nevertheless, it needs to be pointed out that all labels collected for this segment of the data are either retrieved from rule-based (lexicon) or ML (DL) procedures making them very different from the manually-labeled sections (crowdsourcing and annotated by the author of the thesis). This is due to the approach being less reliant on human input and more on automatic procedures. I will refer to this corpus section as *corpus4* for the remaining of this thesis.

Similar to the crowdsourcing part, this section also has several labels per sentence, four in total for each annotated instance, that were used to calculate an overall sentiment. If at least three out of four labels were, for instance, *negative*, the sentence received an overall *negative* sentiment. No overall agreeing label was attributed if two or less equal labels were given to an instance. In this case, the label *undecided* was given. Interestingly, the distribution of this part of the data setup (see figure 3.11) is different from the distribution of the crowdsourced data (*corpus3*) and the part only labeled by the author of this thesis (*corpus2*). While those two find a majority of *negative* sentiment in the data, the combination of the automatic approaches yield a large amount of *undecided* labels (58,656), followed by *neutral* (35,996) and only then by *negative* (12,121). A potential reason for this could be that all initial labels (four per sentence) were achieved using either rule-based

or ML methods with very limited human interaction. As this approach is different from close human reading of every sentence or comment, it could indicate that *negativity* and also *positivity* are harder to grasp for machines than for human readers. Another possible explanation could be that the human-labeled data set is small, especially compared to the amount of data labeled using the rule-based and the ML techniques (*corpus4*). I do not know how representative of the whole RTL corpus the high amount of *negativity* found in the crowdsourced data (*corpus3*) as well as the user comments (*corpus1*) and sentences (*corpus2*) labeled by the author of this thesis is. It would be interesting to study this point further.

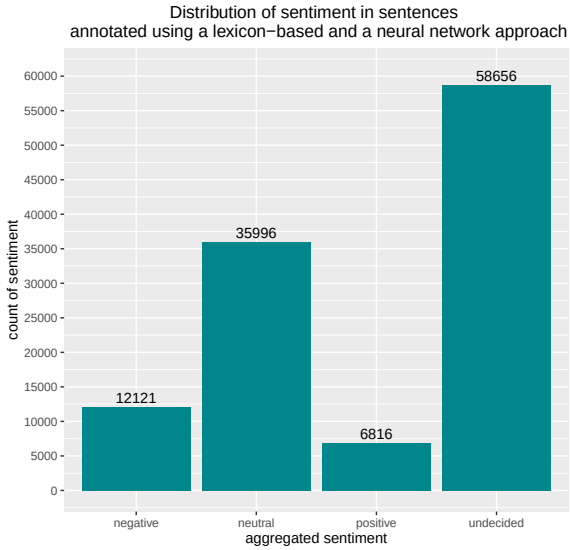



Figure 3.11 – Distribution of sentiment in sentences annotated using a lexicon-based and a neural network approach (*corpus4*).

### 3.6.2.4 Corpus Section of all Annotated Sentences - Corpus5

In a last step, all labeled sentences from all segments of the RTL corpus which were shown in this chapter were combined to one section. This resulted in a size of 116,516 sentences in this data setup (see figure 3.12, denoted as *corpus5*). The comments (*corpus1*) or the adjectives annotated by the author of this thesis were not considered for this combined corpus section, as they are all of different length and scope than sentences. Due to the ML and rule-based part of the data being the largest (*corpus4*), it is not surprising that a combination of all three previously presented corpus sections yields similar results to only looking at the data segment presented in figure 3.11. Nevertheless, it was opted to run the experiments with this setup as well to examine what kind of potential impact the quality of annotation and also the amount of input data might have on various algorithms.

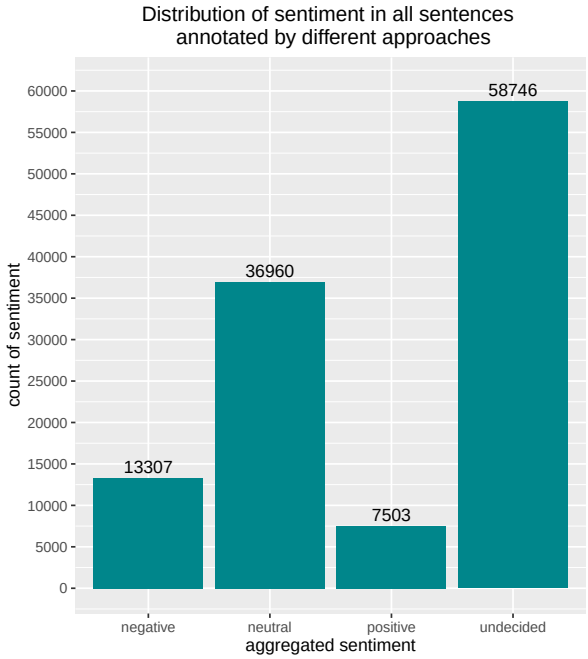



Figure 3.12 – Distribution of sentiment in all sentences annotated by different approaches (*corpus5*).

### 3.6.2.5 Distribution of Sentiment Classes in the Corpus Sections

Looking at all the distributions of sentiment presented in this chapter exposes a problem that should not be left unattended: While all different parts of the annotated corpus include *negative, neutral* as well as *positive* and *undecided/not labeled* sentences, no part is completely balanced. This becomes most evident in the two biggest corpus segments presented in figure 3.11 and figure 3.12 (*corpus4* and *corpus5*). In those two corpus sections, the difference between the biggest class *undecided* and the second most frequent class *neutral* is 22,660 sentences (*corpus4*) or 21,786 sentences (*corpus5*). Such large gaps between different classes are crucial to point out, as they can potentially have an impact on the performance of the various algorithms that were tested with those four setups. It would not be surprising, for instance, if a classifier learns to predict the bigger *undecided* class far better than the *neutral* one. One option for smoothing such imbalances in a training set is to perform naive random oversampling (see chapter 4.1.1.3) which creates copies of data points of minority classes to reduce the existing disproportion.

Another point that should be mentioned in this context is that I am working with sentences extracted from user comments published on RTL Luxembourg's website (RTL Luxembourg (2020)). All of those texts were produced by the Luxembourgish-speaking community registered as users on *www.rtl.lu* that wanted to express their opinion about an article on that website or as reaction to another comment they had seen there. The question that needs to be asked in this context is whether it can even be assumed that there is some kind of balance in the corpus I am working with, as every comment written was composed with

the intention of spreading a personal opinion. This opinion and the intensity behind it might depend on various factors such as the topic of the article, the dynamic of the overall discussion in the comments section and also personal preferences or even personal mood at the moment of writing.

### 3.6.3 Annotation of Sentiment Using the Google Natural Language API

In the previous chapter 3.6.2, the different corpus segments tagged with a *negative, neutral* or *positive* label were introduced. They were used as input to a large variety of ML algorithms of which the results and setups will later be reviewed in chapter 4. In order to check the quality of the annotations collected, a second approach was used to automatically collect annotations for the data set. More precisely, I applied the pre-trained model included in the *Google Natural Language API* (Google) to determine the sentiment of each sentence and user comment that had previously been translated to English using the *Google Translate API* provided by lushan88a (2020). At this point, the discerning reader might ask himself why the same corpus segments were intentionally labeled for sentiment yet again, and this time even in another language. The reason for working with an automatically translated version of the corpus is simple: Luxembourgish being a low-resource language, no ready-to-use version of a sentiment classifier exists openly available on the market. Applying the *Google Natural Language API* (Google) is consequently a great way to handle this challenge and to obtain sentiment labels in an efficient way relying on the models of a big company like Google. The way the *Google Natural Language API* calculates sentiment is more fine-grained than the different approaches used to come up with the *negative, neutral* and *positive* labels presented in chapter 3.6.2. The *API* produces sentiment in a range of different scores. In order to better compare the *Google Natural Language API*'s output with the Luxembourgish corpus segment parts, I consequently simplified this ranging scale assigning every value of 0.25 or above to *positive*, of -0.25 or below to *negative* and all other values to *neutral*. This way, both the Luxembourgish corpus sections and the English ones carry the same kind of annotations and can thus be used to gain further insight into the quality of the labeling process presented in chapter 3.6.2.

#### 3.6.3.1 Agreement of the Labels between the Google Natural Language API Corpus and the Luxembourgish Corpus Sections

Table 3.9 shows the number of agreed sentence annotations by label. This means, for instance, the amount of *negative* labels that both the *Google Natural Language API* and the different approaches used to annotate the Luxembourgish language data set (see figure 3.12) agreed upon. In total, 13,307 *negative*, 30,622 *neutral* and 7,503 *positive* sentences were labeled in the same way by the *Google Natural Language API* and the different annotation approaches. This amounts to a total of 51,432 common labels which is roughly half of the whole amount of annotations collected (116,516). At first glance, this seems

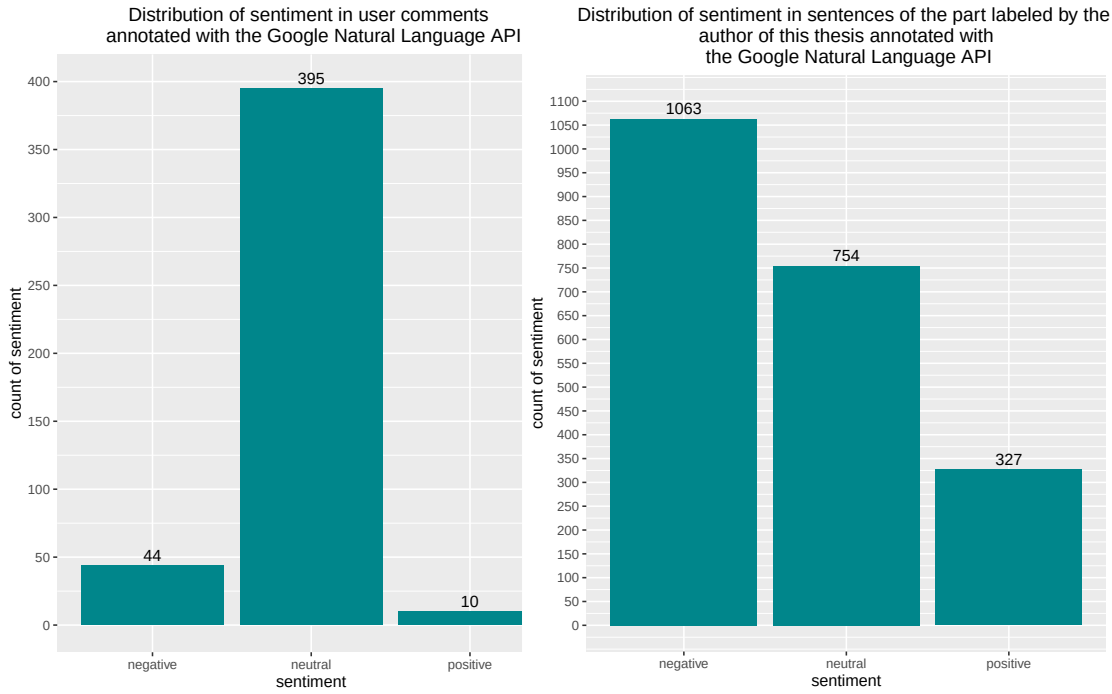
to be a big difference. However, we need to keep in mind that 58,746 instances of the Luxembourgish data set were annotated with an *undecided* label (see figure 3.12) whereas the English part labeled with the *Google Natural Language API* (Google) only applied *negative, neutral* and *positive* as annotation options. Also, all sentences tagged with a *negative* or *positive* label were annotated in the same way by both the Luxembourgish language approaches and the *Google Natural Language API* (Google) one (compare figure 3.12 and figure 3.16) suggesting that both ways to annotate perform in a similar way. This finding is surprising, but at the same time a good indication for the quality of the Luxembourgish corpus segments.

Table 3.9 – Number of annotations that the *Google Natural Language API* and the different annotation approaches summarized in figure 3.12 agreed on.

| Label | Number of Annotations |
|---|---|
| Negative | 13,307 |
| Neutral | 30,622 |
| Positive | 7,503 |

### 3.6.3.2 Sentiment in the Different Corpus Sections

This also becomes apparent when looking at the next step in which the distribution of sentiment in each corpus segment was plotted separately. Contrary to the plots described in chapter 3.6.2, the sentences and comments were translated to English and the *Google Natural Language API* (Google) was used in a simplified form to come up with the overall sentiment of each instance. Figure 3.13 shows the first corpus segment of the user comments (*corpus1*) and sentences labeled by the author of this thesis (*corpus2*), figure 3.14 the crowdsourcing part (*corpus3*), figure 3.15 the lexicon-based and ANN section (*corpus4*) and figure 3.16 all annotated sentences together (*corpus5*). If those are compared with the distribution of sentiment of the original Luxembourgish language corpus introduced in figures 3.9, 3.10, 3.11 and 3.12, a similarity in the allocation of sentiment in both cases can be perceived. Naturally, there are some differences, also due to the lack of the *undecided/not labeled* class in the *Google Natural Language API* part, but the overall picture is similar. The crowdsourcing data (*corpus3*), the sentences originally annotated by the author of this thesis (*corpus2*) and the overall distribution of sentiment in all sentences (*corpus5*) even have the exact same range of sentiment. This goes from *negative* to *neutral* to *positive* for the sentences and crowdsourcing and from *neutral* to *negative* to *positive* for all data points.




a User comments.

b Sentences.

Figure 3.13 – Distribution of sentiment in the user comments (left, *corpus1*) and sentences (right, *corpus2*) part annotated by the author of this thesis (*Google Natural Language API*).

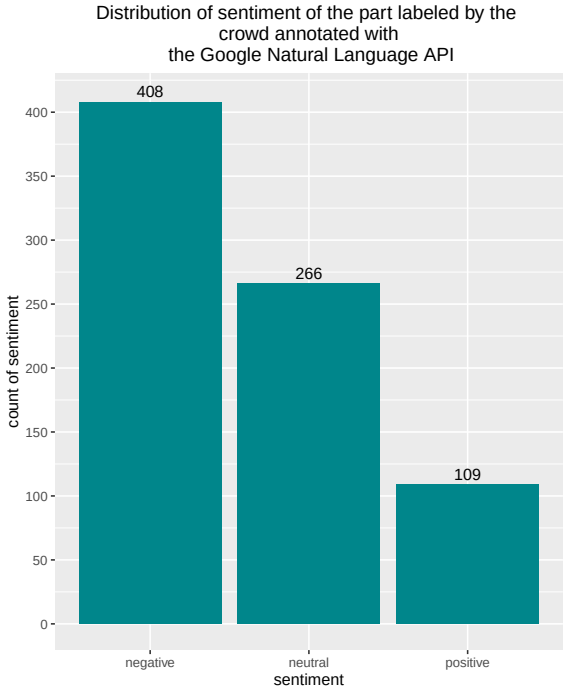



Figure 3.14 – Distribution of sentiment in sentences annotated by the crowd (*corpus3*) (*Google Natural Language API*).

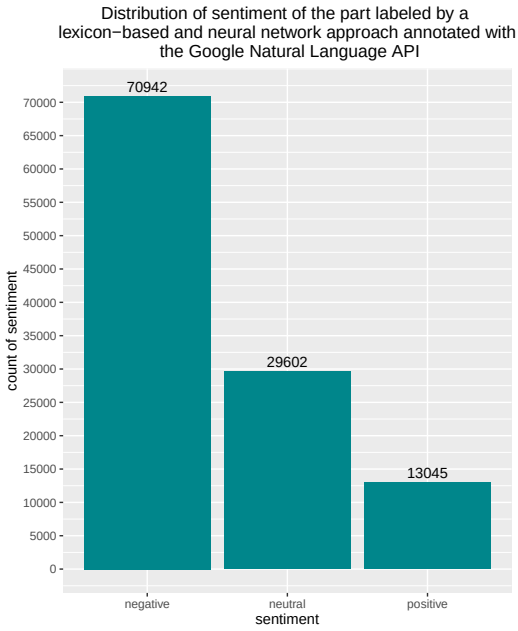



Figure 3.15 – Distribution of sentiment in sentences annotated using a lexicon-based and a neural network approach (*corpus4*) (*Google Natural Language API*).

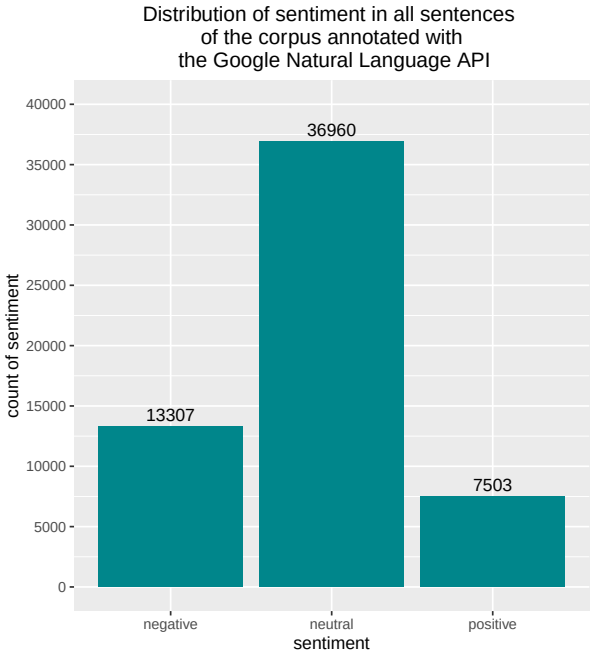



Figure 3.16 – Distribution of sentiment in all sentences annotated by different approaches (*corpus5*) (*Google Natural Language API*).

### 3.6.4 Conclusion to Be Drawn from Comparing the Luxembourgish and the English Corpus Sections

What can we learn from this comparison of different annotation approaches? First of all, it indicates that the labeling process is well-adapted to the data at hand. This is supported by the fact that both the *Google Natural Language API* (Google) and the various techniques presented in chapter 3.6.2 labeled in a very similar way. Second, the similarity in annotation underlines the underlying challenge of class imbalance in the corpus which can potentially lead to bias in the training process. Possible ways to tackle this problem are portrayed in the results section of the experiment shown in chapter 4.1.1.1. One of those approaches is naive random oversampling to smooth the imbalance and remove some potential bias.

In this section, a translated version of the corpus using *Google Translate* (lushan88a (2020)) was shown, which was labeled for sentiment with the help of the *Google Natural Language API* (Google). Comparing the annotation of those corpus segments with the ones previously presented in chapter 3.6.2 clearly lay out that the task of attributing *negative, neutral* and *positive* was satisfactorily completed by the variety of annotation methods used on the Luxembourgish data. In the following ML experiments (chapter 4), the Luxembourgish corpus sections were the only input, notwithstanding the knowledge that the English corpus segments could also be useful for such an analysis. Future work could move the comparison of both kinds of corpus sections ergo one step further and see if there are differences of performances of the algorithms depending on whether they were trained on labels provided by the *Google Natural Language API* (Google) or one of the annotation approaches presented in chapter 3.6.2.

Chapter 4

# Detecting Sentiment Using Machine Learning

Up to this point, the focus was on preprocessing and enriching of the unstructured corpus data provided by RTL Luxembourg for querying and retrieval purposes in a way that made it accessible for the next step of the work, the ML experiment part. As explained in chapter 2.4, ML is used to learn how to automatically detect patterns in data and apply this knowledge to new cases (Shalev-Shwartz and Ben-David, 2014, p. 2). In the case of sentiment detection, this means having a corpus of data instances that are annotated with a sentiment label and are taken as input for learning how sentiment is expressed in that specific language. This is useful on one hand for classifying the sentiment of new unseen sentences and on the other hand for having better insights of language patterns for building sentiments.

To put it into more precise words and adapted to this thesis, chapter 3 explained the various preparation steps required to transform the RTL Luxembourg corpus to an annotated version containing POS tags, split into word segments and partially annotated with *negative, neutral* and *positive* sentiment on adjective-, sentence- and comment-level. In the upcoming sections, it will now be explained how this previously prepared data was useful for learning to automatically detect sentiment in Luxembourgish language instances by training and developing different algorithms. The setup of the ML experiments is multi-layered and performed in several iterations. The idea of this is to study whether or not different factors have an impact on how well an ML algorithm can learn to predict *negative, neutral* or *positive* sentiment expressions in textual data produced in Luxembourgish.

The ML experiments are structured in the following way: First, the tasks were split into three main sections all containing the sentiment annotated parts of the RTL Luxembourg corpus data described in chapter 3.6. Those corpus sections are named *corpus1* (user comments labeled for sentiment by the author of this thesis), *corpus2* (sentences annotated with a sentiment by the author of this thesis), *corpus3* (sentences annotated with a sentiment label by crowdsourcing), *corpus4* (sentences labeled for sentiment using a lexicon-based and two ANN approaches) and *corpus5* (all sentences of *corpus2, corpus3* and *corpus4*

carrying a sentiment label). This corpus was once translated to English (see chapter 4.1, denoted as *setup 1*) and once to German (see chapter 4.2, considered as *setup 2*). The translations were carried out using lushan88a (2020)'s Google Translate API. The original Luxembourgish version of the corpus was also used for the experiments (see chapter 4.3, named *setup 3*). At first glance, working with automatic translations to German and English might seem unusual, especially as the same sentiment annotations for all languages were kept despite the awareness that expressions of sentiment might be cultural dependent. However, it should not be forgotten in this context that Luxembourgish is a low-resource language and both German and English are not. Therefore, the amount of resources and (preprocessing) tools available for those languages by far exceed the ones available for Luxembourgish making it easier to work with corpus instances translated to German and English.

In each section, several experiments were carried out in the same sequence of events: First, word embeddings were taken as input vectors for training various supervised classifiers as well as an ANN. Second, a *BERT* model was trained on the different corpus sections. The third experiment of each section takes *tf-idf* vectors to several supervised classifiers. Each experiment in each section is carried out with each corpus segment once in a preprocessed and once in an unpreprocessed form. That way, the impact of preprocessing techniques on the performance of the different algorithms used can be studied. The performance of all those experiments is evaluated with the $F_1$ score (see chapter 2.8) to be able to compare objectively and come up with recommendations of which techniques might be most suitable for developing a sentiment detection system for the RTL corpus. Table 4.1 shows an overview of the different experimental setups. The Luxembourgish experiments include one more approach, i.e. the usage of linguistic features for sentiment analysis. This is only implemented for the Luxembourgish version of the RTL corpus. In the next section, the setup and results working with English translations will be described, followed by the German experiments. The chapter ends with the analysis of the Luxembourgish original data.

Table 4.1 – Overview of the setup of all ML experiments.

| Setup | Vectorization Approach | Corpus Section | Preprocessing |
|---|---|---|---|
| All (see 4.1, 4.2, & 4.3) | Word embeddings (see 4.1.1, 4.2.1, 4.3.1) | Corpus1 | yes<br>no |
| | | Corpus2 | yes<br>no |
| | | Corpus3 | yes<br>no |
| | | Corpus4 | yes<br>no |
| | | Corpus5 | yes<br>no |
| | BERT (see 4.1.2, 4.2.2, 4.3.2) | Corpus1 | yes<br>no |
| | | Corpus2 | yes<br>no |
| | | Corpus3 | yes<br>no |
| | | Corpus4 | yes<br>no |
| | | Corpus5 | yes<br>no |
| | tf-idf (see 4.1.3, 4.2.3, 4.3.3) | Corpus1 | yes<br>no |
| | | Corpus2 | yes<br>no |
| | | Corpus3 | yes<br>no |
| | | Corpus4 | yes<br>no |
| | | Corpus5 | yes<br>no |
| Only Luxembourgish data (see 4.3) | Linguistic features: tf-idf & one-hot encoding (see 4.3.4) | Corpus1 | yes<br>no |
| | | Corpus2 | yes<br>no |
| | | Corpus3 | yes<br>no |
| | | Corpus4 | yes<br>no |
| | | Corpus5 | yes<br>no |

## 4.1 Setup 1 - Working with Translations to English

As Luxembourgish is a low-resource language for the NLP community, the amount of research done and tools developed so far is very limited, especially compared to bigger languages such as English. One of the ways to deal with this lack of resources is to translate the initial data set you want to be working with to a resource-rich(er) language and use this instead of developing all sorts of new tools for your language (Luxembourgish in this case). While this is not the most linguistically grounded approach for doing research in NLP, more recent advances in neural machine translation (NMT) have lead to significant improvements of automatic translations (Johnson et al. (2017)). This thesis uses the Google Translate API for Python proposed by lushan88a (2020) to translate sentences from Luxembourgish to English. Google's NMT model consists of three parts: an encoder, a decoder and an attention network (Wu et al., 2016, p. 3). Both the encoder and the decoder network are Long Short-Term Memory recurrent networks (LSTM RNNs) with eight layers (Wu et al., 2016, p. 2). The existence of the attention network makes it possible for the decoder to focus on different parts of the encoded sentence in the course of decoding (Wu et al., 2016, p. 3). This system has been further extended to be able to translate between a variety of languages in Johnson et al. (2017).

Before introducing the experiments and results that worked with those translations, the study of a couple of sentences from the corpus made it possible to get a first intuition of the quality of Google's NMT. Table 4.2 includes ten sentences translated from Luxembourgish to English using the Google Translate API for Python provided by lushan88a (2020). The translation of the first sentence in this table is very close to the Luxembourgish original with a word by word translation from one language to the other. A similar case can be observed with the fifth and ninth sentences. While all three are not the most idiomatic translations possible, they can also not be considered as being completely incorrect. Another challenge for the automatic translation can be seen in sentence three. Due to the non-manual sentence splitting during the preprocessing of the RTL corpus (see chapter 3.1), some sentences are split in unexpected places after a period. In this case, the Luxembourgish sentence ends with *asw.* 'and so on, etc.' which is not the actual semantic end. In order to overcome this shortcoming, Google NMT changes the meaning of the unit and adds the verb "is" to create a complete syntactically correct sentence which, however, has a different meaning from the source text. As explained in chapter 2.10, a particularity of the Luxembourgish language is that the orthographic rules are not as in-depth taught in schools as the other two official languages of the country, i.e. German and French, and that this leads to a large variety of spelling variation in the text production of Luxembourgish speakers. Sentence number seven is a perfect example of how this can be a challenge for translations. The author of the sentence used the adjective *wëll* 'wild' instead of the conjunction *well* 'because' leading to an incorrect English sentence. Such words that are almost identical in writing and are therefore easy spelling mistakes are particularly challenging for any kind of automatic language processing. Additionally, the style of writing can also lead to mistakes in translations. An example for this is the second sentence in the table. The last part of the sentence, *wou bleift do Logique* 'Where remains there logic, i.e. where is

the logic?' is actually its own syntactic unit. This is very difficult for a machine to grasp as the author did not provide any punctuation mark to separate it from the rest of the comment. Syntax in general seems to be a problem in some cases for Google's NMT from Luxembourgish to English. This can be observed for sentence ten, for instance, in which the subject of the sentence was wrongly identified or for sentence six in which the relative clause was not deciphered correctly. Last but not least, some incorrect translations on word basis can be found in those example sentences shown in table 4.2, such as *Bettel*, the prime minister of Luxembourg, being translated with 'beggar' or *dafstomm* 'deaf-mute' becoming 'dumbfounded' in the English sentence. To sum up, this analysis shows that automatic translations from Luxembourgish to English are still far from being perfect. Nevertheless, the translations are of sufficient quality to serve as input to the various ML algorithms, as the general structure of the translations is not bad and the number of translations by automatic means is far bigger than what could be manually translated in that amount of time. Therefore, it should be noted that working with Google NMT does not yield flawless results, but the advantage of having a large number of translated instances as training input outweighs the disadvantage of inaccuracy of translations.

In consequence, the planned experiments with those English translations as corpus input were implemented. They were used for three experiments which are described in the following chapters. Each experiment was carried out in several iterations taking one of the four corpus segments presented in chapter 3.6.2. To be able to examine the impact of preprocessing on this translated corpus, stopwords were removed and lemmatization with *NLTK* (Bird et al. (2009)) as well as lowercasing were done to create a preprocessed version of the English sentiment corpus sections. Those preprocessing steps transform a sentence like the first one of the ten random sentences shown in table 4.2 from "What is that then for a sarcastic commentary." to "sarcastic commentary .". Another example of how preprocessing worked on the data is the second sentence in this table 4.2, in which "Such a camp should be centrally located in the country which brings massive traffic with it now the jemp has to drive from Elwen with his fuel truck to Bascharage to find the one where logic remains." was preprocessed to "camp centrally located country brings massive traffic jemp ha drive elwen fuel truck bascharage find one logic remains .". In addition to the preprocessing of the sentence or user comment in its respective corpus segment, some changes on the sentiment labels themselves were also carried out. Instead of a string, each (overall) sentiment was converted to a numerical value. *Neutral* was changed to 0, *negative* to 1, *positive* to 2 and *no label/undecided* to 3. *

*As described in section 3.6.2, both the *corpus3* and the *corpus4* data had three (*corpus3*) or four (*corpus4*) annotations per sentence which were used to calculate an overall sentiment per instance. The user comments and sentences labeled by me (*corpus1* and *corpus2*) only had one annotation per labeled instance, i.e. did not require the calculation of an overall sentiment.

Table 4.2 – Ten random Luxembourgish sentences extracted from the data set and their translation to English using Google Translate.

| Sentence in Luxembourgish | Sentences in English | Sentence ID |
|---|---|---|
| 1.) Waat ass daat do dann fir e sarkastesch commentaire. | What is that then for a sarcastic commentary. | 24424ao |
| 2.) E sou e Lager soll central am Land leien dat do brengt massif Verkéier mat sech elo muss de jemp vun Elwen mat sengem Mazoutskamion bis op Käerjeng fueren fir déen ze fellen wou bleift do Logique. | Such a camp should be centrally located in the country which brings massive traffic with it now the jemp has to drive from Elwen with his fuel truck to Bascharage to find the one where logic remains. | 37652bb |
| 3.) Et steht awer neirens wat hei am Land en Haus Appartement asw. | However, there is no such thing as a house, apartment, etc. in this country. | 23446bt |
| 4.) Ee gudden Drëttel vun den Deputéierten schéngt dafstomm ze sinn gesäit een dach wéi se während de Sitzungen apathesch an hire Still hänken a nëmmen de Fanger beweege fir ofzestëmmen ouni dass jemols een eenzegt Wuert ze verlauschteren ass also Matleefer. | A good third of the deputies seem to be dumbfounded as they hang apathetically in their chairs during the sessions and only move their finger to vote without ever hearing a single word. | 26313cc |
| 5.) " Wat den Här Bodry also ee Vertrieder vun der Koalitioun domadder mengt seet hien dann hei : | "So what Mr Bodry, a representative of the coalition, means by this, he says here: | 26383cc |
| 6.) Eng privat Entreprise verkeeft en Gebai dat hier gehéiert un eng aner privat Entreprise. | A private company sells a building belonging to another private company. | 11221cs |
| 7.) Et geet ëm de Prinzip net ëm méi Souen a wëll et Proffen sin. | It's about the principle not about more money and wanting to be teachers. | 13257cu |
| 8.) Gewerkschaften hunn schonn virun e puer Deeg formell ofgestridden dat et en Accord ginn ass dat haten mir jo och schonn bei der Caritas wou den Bettel gesot huet esou guer Caritas ass op eiser Säit déi hunn dat och formell ofgestridden. | A few days ago, trade unions formally denied that there was an agreement, and we already had that at Caritas, where the beggar said that even Caritas is on our side, and they formally denied this. | 13400cu |
| 9.) Alkohol gehéiert op kee Fall an den Indexkuerf an "d" TVA um Alkohol misst op mannst op 25 % gehuewen gin egal wéivill "d" Horesca jéimert. | Alcohol is by no means included in the index basket and "VAT on alcohol should be raised to at least 25% no matter how much" Horesca complains. | 13539cu |
| 10.) Kengem Betrib hei am Land schued TVA Erhéijung direkt. | No business in this country is directly harming VAT increase. | 13856cu |

The three experiments that are described in the following sections were structured in the following way: First of all, pre-trained English word embeddings (Mikolov et al. (2013a)) as input to several supervised classification algorithms and one single-layered ANN (see chapter 4.1.1 and 4.1.1.1) were used. A single-layered ANN was taken for simplicity reasons, but future work could also be carried out with more hidden layers in order to take advantage of the range of possibilities DL provides. Second, an experiment with *BERT* (*Bidirectional Encoder Representations from Transformers*) (Devlin et al. (2019)) was carried out of which the setup and results are shown in chapters 4.1.2 and 4.1.2.1. Third, I applied *tf-idf* vectorization to attain input vectors for training several supervised classifiers (see chapters 4.1.3 and 4.1.3.1).

### 4.1.1 Google Word Embeddings as Input Vectors

The first experiment carried out with the different corpus sections in their English version uses a 300-dimensional Google word embeddings model (Řehůřek and Sojka (2010)) as vector input to training several supervised learning algorithms. The inspiration for this was taken from Galeshchuk et al. (2019) whose approach was closely followed and adapted to the data set when it seemed necessary. Galeshchuk et al. (2019) worked with automatic translations from several Slavic languages to English. They used a large pre-trained word2vec model for the English language to obtain averaged vector representations of the words in their sentiment instances and plugged those into several supervised algorithms as well as one ANN (Galeshchuk et al. (2019)). The intuition behind such an implementation is that word2vec incorporates context and therefore provides more additional information to a training process than simply obtaining vectors from the corpus itself. This would be the case for vectorization techniques such as *tf-idf* (see chapters 4.1.3, 4.2.3 and 4.3.3) which were used to directly create a document-term matrix from the corpus data (see chapter 2.6.3). Galeshchuk et al. (2019)'s approach applies the powerful technique of word2vec (Mikolov et al. (2013a)) that creates a vector space capturing meaning and similarities between words by means of a shallow ANN. More precisely, they apply Řehůřek and Sojka (2010)'s implementation of Google's 300-dimensional pre-trained word2vec model for the English language. The advantage of using this pre-trained word2vec model over creating your own is that it has been trained on a corpus of about 100 billion words meaning that a large amount of terms will have appeared several times and in various contexts (Galeshchuk et al., 2019, p. 122). This contributes to the quality and variety of the vectors achieved. Those vectors can then be taken as input to classification tasks.

For this experiment, Galeshchuk et al. (2019)'s method was followed for each of the corpus segments. Two iterations for each corpus section were carried out, once with preprocessed English sentences/user comments and once without preprocessing in order to study the influence of preprocessing on the classification task. For each execution of the experiment, the first step was to aggregate a 300-dimensional vector for the whole sentence or user comment. This was done by averaging the word vectors taken from the pre-trained word2vec model. If a word existed in the user comment or sentence but not in the word2vec model's vocabulary, a vector of zero for that specific word was obtained. The averaged sentence (or user comment) vectors were then taken as input to a decision tree (DT), a random forest (RF), a support vector machine (SVM), a k nearest neighbors (KNN), a logistic regression algorithm (LRM) as well as to a single-layered neural network (ANN). As far as the SVM algorithm is concerned, a polynomial kernel was used except for the larger corpus segments (*corpus4* and *corpus5*), as a linear kernel yielded a significant improvement in training time for larger data sets. 20 % of the respective corpus segment (*corpus1, corpus2, corpus3, corpus4* or *corpus5*) were taken as test data and 80% as training data.

#### 4.1.1.1 Results of the Word Embedding Experiment for English

In this chapter, the results achieved for the sentiment classification experiment with word embedding in the different setups explained above will be discussed. For the purpose of evaluating the performance of the training procedures, the overall $F_1$ score for training with each algorithm is taken into account. $F_1$ score is defined as the weighted average of precision and recall. The calculation was carried out with scikit-learn's implementation (Pedregosa et al. (2011)) following the mathematical notation of:

$$F_1 = 2 * \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.1}$$

with values reaching from 0 (very low $F_1$ score) to 1 (perfect $F_1$ score). As there are four potential labels, i.e. *negative, neutral, positive* or *undecided*, the $F_1$ score calculation used macro averaging. This method first calculates a separate $F_1$ score for each label and then averages the results to come up with an overall score to evaluate the performance of the classifier in its whole. The definition of $F_1$ score, as well as the precision and recall measures linked to it, are explained in greater detail in chapter 2.8. In order to facilitate comparison of the results of the different experiments, this $F_1$ measure is the basis for analysis of all experiments that follow in the rest of chapter 4, no matter which language or input vectors were used for training.

Table 4.3 shows the $F_1$ scores for the performance of algorithms trained with different corpus sections, both with and without having preprocessed the data first. All consecutive tables in this chapter follow the same structure to facilitate the comparison of the different experiments and setups. The first column contains the different corpus sections described in chapter 3.6.2, i.e. *corpus1, corpus2, corpus3, corpus4* and *corpus5*. The potential labels $y$ for this classification task were *negative, neutral, positive* and *undecided*. The second column shows whether the training and testing of the algorithms were carried out using a preprocessed or an unpreprocessed version of the respective corpus section. The subsequent columns contain the $F_1$ scores that result from evaluation. Each of those results was rounded to the second decimal after the period. The figures presented in bold print are the highest scores for each corpus segment. The results presented in table 4.3 indicate a slight tendency towards ANN being the most suitable algorithm, as it mostly performs best for the classification tasks described here with the highest performance being an $F_1$ value of 0.45. While this value denotes the best performing algorithm for the setup training with the RTL corpus data translated to English and with Google word embeddings as input vectors, it should not be forgotten what this value actually means. As stated above, the $F_1$ measure can be between 0 and 1. Having an evaluation of the best performing algorithm with an $F_1$ measure of 0.45 therefore suggests that randomly guessing the class of a user comment or sentence could even lead to a larger proportion of correct decisions taken, as the algorithm does not even manage to attribute the adequate sentiment label in more than 50% of the cases, i.e. does not exceed an $F_1$ value of 0.5. This is rather problematic, as building an automatic sentiment analysis system should ideally perform better than randomly guessing the correct sentiment.

Table 4.3 – $F_1$ score of all corpus sections with and without preprocessing and with English word embeddings.

| Corpus Section | Preprocessing | DT | RF | SVM | KNN | LRM | ANN |
|---|---|---|---|---|---|---|---|
| Corpus1 | yes | 0.27 | 0.36 | 0.26 | 0.25 | 0.37 | 0.29 |
| Corpus1 | no | 0.31 | **0.40** | 0.31 | 0.35 | 0.34 | 0.35 |
| Corpus2 | yes | 0.39 | 0.40 | 0.34 | 0.37 | 0.40 | 0.40 |
| Corpus2 | no | 0.39 | 0.41 | 0.35 | 0.38 | 0.43 | **0.45** |
| Corpus3 | yes | 0.26 | 0.21 | 0.15 | 0.23 | 0.30 | 0.24 |
| Corpus3 | no | 0.19 | 0.21 | 0.19 | **0.31** | 0.21 | 0.21 |
| Corpus4 | yes | 0.34 | 0.37 | 0.32 | 0.36 | 0.42 | **0.44** |
| Corpus4 | no | 0.34 | 0.36 | 0.33 | 0.34 | 0.41 | 0.43 |
| Corpus5 | yes | 0.38 | 0.36 | 0.31 | 0.35 | 0.41 | **0.42** |
| Corpus5 | no | 0.34 | 0.35 | 0.32 | 0.35 | 0.41 | 0.41 |

#### 4.1.1.2 Further Interpretation of the Results

What could be the reason for no $F_1$ value exceeding 0.5 in this first experiment? In order to investigate this question and gain further insights, also for the following experiments in this chapter, the labeled user comments and sentences of the RTL corpus need to be studied in more detail to find possible hints that go beyond simply showing an overall $F_1$ score for each algorithm and corpus section.

For this purpose, let us first have a look at the different graphs presented in chapter 3.6.2 which display the distribution of all classes in the different data sections used for training and testing. Those graphs have one important aspect in common, namely that they exhibit a high class imbalance in each of the segments. For instance, the user comments and the sentences labeled solely by the author of this thesis (*corpus1* and *corpus2*) contain a lot more *negative* than *positive* labels (see figure 3.9) which is also the case in the *corpus3* segment (figure 3.10). Furthermore, the sentences collected by automatic means (*corpus4*, see figure 3.11) include far more *undecided* labels than *positive* ones. Such large imbalances in all corpus segments influence the way the algorithms learn and can lead to a wide spectrum of $F_1$ score values. This becomes apparent when examining the $F_1$ scores separately for each class in each data section of the corpus, as they are very different in smaller as well as in larger corpus parts.

Those two examples underline this occurring pattern. * The first example is the corpus section containing all sentences labeled by the author of this thesis (*corpus2*) which consists of 859 *negative*, 746 *neutral*, 537 *positive* and 2 *unlabeled* sentences. For training and testing purposes, this corpus section was split into 1,715 sentences for training and 429 for testing. † Overall, *negative* was the class that was best learned by all classifiers with SVM scoring the highest. The preprocessed labeled sentences received an $F_1$ score of 0.60 for the *negative* class, 0.29 for the *neutral* and 0.14 for the *positive* class while the unpreprocessed labeled

*Due to readability, only two examples from the whole experimental setup are discussed at this point.

†The label *undecided/not labeled* did not appear in the test set.

instances obtained 0.59 for the *negative*, 0.34 for the *neutral* and 0.13 for the *positive* instances. As shown in table 4.3, this leads to the overall $F_1$ score for the *corpus2* section trained with SVM being 0.34 for the preprocessed and 0.35 for the unpreprocessed version of the data. This big difference in the $F_1$ score of different classes in the SVM experiments shows that imbalance is a true challenge in the corpus having an impact on the learning process and the overall performance of a classifier. Ideally, a classifier should be capable of detecting the sentiment for each class equally well and there should not be such a big dispersion as in this case. This first indication towards learning biases becomes even more apparent in bigger parts of the corpus like *corpus4*. It consists of 90,871 training and 22,718 test sentences of which 12,121 are labeled *negative*, 35,996 *neutral*, 6,816 *positive* and 58,656 as *undecided*. Similar to the results achieved on the labeled sentences of the first example (*corpus2*), SVM is one of the best performing classifiers. For preprocessed instances, the $F_1$ score for *negative* is 00.00, 0.55 for *neutral*, 0.01 for *positive* and 0.70 for *undecided*. The unpreprocessed sentences achieved $F_1$ values of 00.00 (*negative*), 0.54 (*neutral*), 0.06 (*positive*) and 0.70 (*undecided*). The biggest class (*undecided*) was hence learned the best while the two smallest classes, *positive* and *negative*, had a very low $F_1$ score on the test data set.

To sum up, class distribution and the amount of labeled instances algorithms have at their disposal for learning patterns, seem to have an impact on the performance of those algorithms. The examples mentioned above indicate that there can be a big difference in how well an algorithm is capable of learning to detect a class, even ranging as far as from 00.00 to 0.70 in $F_1$ score. This matter is a serious problem when we think beyond research and about potential real-life applications of such a system. Ideally, a system should be trained that is capable of equally well detecting the sentiment in every possible class. Putting a system on the market that is very bias towards the largest class, for instance *undecided* in the second example, would not be helpful. Such a bias could lead to a system tagging cases like *De Manuel ass haut ganz frou.* 'Manuel is very happy today.' with an *undecided* sentiment instead of a *positive* one which is not helpful and would rather lead to confusion in users of such an application.

#### 4.1.1.3 Naive Random Oversampling

Is there a way to tackle this problem of bias towards different classes? How can this big discrepancy between the classes be smoothed? And does this indeed make the learning better and thus more adequate for transferring the learned system to real-world applications? For this thesis, all results are mostly discussed in the research context. Ideally, the findings of this work would eventually pave the way towards a sentiment detection system which could be used by actual users such as the partner RTL Luxembourg of the *STRIPS* project. Therefore, this chapter is dedicated to analyze a potential way of overcoming class imbalance.

One technique to help with such an imbalanced corpus is oversampling. This method creates new samples following specific rules to ensure that no class is underrepresented in the training sample. Different ways to carry out this approach are possible. In this case,

a naive random oversampling approach was applied. This method artificially duplicates examples in the classes that do not contain the majority of sentiment instances. If, for example, the *negative* class is the biggest one in a data set with 20,000 entries, all other classes will artificially be increased to 20,000 entries as well making it an (artificially) perfectly balanced corpus. The practical implementation of this oversampling task was performed with the *Imbalanced-learn* toolkit proposed in Lemaître et al. (2017).

The $F_1$ scores achieved for all different corpus sections and algorithms using naive random oversampling are shown in table 4.4. Most results laid out in table 4.4, compared to the ones in table 4.3 that were obtained using training without oversampling, show an improvement in $F_1$ score. The highest $F_1$ score was for the unpreprocessed user comments annotated by the author of this thesis (*corpus1*) using an ANN for training. This setup got an $F_1$ value of 0.50 compared to 0.35 in the not oversampled case shown in table 4.3. Despite the fact that the results are still rather low with $F_1$ scores mostly below 0.5, the boost in the values demonstrate a clear tendency towards a positive effect of applying oversampling techniques for such highly unbalanced data sets. Future work could therefore include applying more fine-grained oversampling techniques such as SMOTE (Lemaître et al. (2017)) to this corpus in order to potentially observe even greater impact on the different $F_1$ scores and subsequently the quality of performance. Albeit oversampling, even in its most simplest form of naive random oversampling, shows promising tendencies of improving the classification results, more research needs to be done so as to improve the corpus itself in a sustainable way. The reason for this is that the RTL corpus comprises a big variety of topics, language use and spelling varieties written by a multitude of authors. Therefore, it is not considered to be sufficient to (artificially) balance the corpus by simply duplicating random examples of user comments and sentences that were labeled during the various annotation processes. The reason for this is that it cannot be guaranteed that there is an adequate distribution of authors and topics and henceforth it would have to be examined closely how such an oversampled classifier would perform on completely unseen and new data. For those arguments mentioned here, oversampling was not performed during the other experiments that were implemented for the thesis.

Table 4.4 – $F_1$ score of all corpus sections with and without preprocessing and with naive random oversampling and English word embeddings.

| Corpus Section | Preprocessing | DT | RF | SVM | KNN | LRM | ANN |
|---|---|---|---|---|---|---|---|
| Corpus1 | yes | 0.27 | 0.38 | 0.33 | 0.29 | 0.47 | 0.49 |
| Corpus1 | no | 0.26 | 0.44 | 0.44 | 0.38 | 0.48 | **0.50** |
| Corpus2 | yes | 0.28 | **0.43** | 0.32 | 0.28 | 0.32 | 0.32 |
| Corpus2 | no | 0.26 | **0.43** | 0.36 | 0.38 | 0.32 | 0.42 |
| Corpus3 | yes | 0.21 | 0.18 | 0.22 | 0.20 | **0.28** | **0.28** |
| Corpus3 | no | 0.22 | 0.21 | 0.23 | 0.24 | 0.21 | 0.24 |
| Corpus4 | yes | 0.34 | 0.39 | 0.43 | 0.35 | 0.43 | **0.45** |
| Corpus4 | no | 0.34 | 0.39 | 0.43 | 0.35 | 0.43 | **0.45** |
| Corpus5 | yes | 0.34 | 0.39 | 0.43 | 0.35 | 0.43 | 0.46 |
| Corpus5 | no | 0.34 | 0.40 | 0.43 | 0.35 | 0.43 | **0.48** |

The first conclusion which can be drawn from working with the word embeddings setup is that translations to bigger languages such as English indeed provide more potential language resources. However, this does not automatically lead to promising results, which is shown by the $F_1$ score in all sections never exceeding 0.50 (see tables 4.3 and 4.4). The approach of applying pre-trained word embeddings as input vectors to various classification algorithms were used two more times for this thesis, once for the German translations of the corpus and once for the Luxembourgish original data. Those results will be discussed in chapters 4.2.1.1 and 4.3.1.1. Before moving on to those different languages, two more experimental setups for the English translations will be presented in the upcoming sections.

### 4.1.2 English BERT

The main idea of this second experiment with the English translation of the RTL corpus was to examine how a very important, if not the most significant model at the moment of writing this thesis, performs with the different corpus sections of the Luxembourgish user comments. Luxembourgish is part of a multilingual *BERT* model (see chapter 4.3.2), but to the best of my knowledge, no *BERT* model exclusively trained for Luxembourgish exists yet. In consequence, I decided to take advantage once again of the automatically translated corpus segments and use an English *BERT* model. *BERT* stands for *Bidirectional Encoder Representations from Transformers* and was first proposed for the English language in Devlin et al. (2019). It pre-trains "deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers" (Devlin et al., 2019, p. 4171). The advantage of this system is that the pre-trained model "can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks" (Devlin et al., 2019, p. 4171). *BERT* was described in more detail in chapter 2.7.6.

In order to apply the *BERT* structure to the translated corpus, I adjusted Valkov (2020)'s code to the needs of this analysis. Valkov (2020) uses the Transformers library by Hugging Face (Wolf et al. (2020)) and PyTorch (Paszke et al. (2019)) to perform a sentiment analysis task on reviews written in English with the *BERT* model. The English *BERT* model for this experiment (*bert-base-cased*) was introduced in Devlin et al. (2019). The corpus sections presented in chapter 3.6.2 were taken and a model was fine-tuned for each of those, once with the data in its preprocessed and once in its unpreprocessed form. The training of the model was done for four epochs for the user comments and sentences annotated by the author of this thesis (*corpus1* and *corpus2*) as well as the crowdsourced data (*corpus3*). The segments labeled with a lexicon-based and an ANN approach and also the whole collection of all sentences labeled by different means were trained for two epochs (*corpus4* and *corpus5*). For each corpus segment, 10% of the data were taken as test data and 90% as training data. All training ran with a batch size of 16. In the following chapter, the results achieved for this training setup will be presented.

#### 4.1.2.1 Results of the English BERT Experiment

This experiment with the English *BERT* model (*bert-base-cased*) (Devlin et al. (2019)) was run with the different corpus sections which were discussed in chapter 3.6.2. Each corpus segment was once taken in its preprocessed and once in its unpreprocessed form to study whether preprocessing had an impact on the performance of the experimental setup. Table 4.5 includes the $F_1$ scores for each data section. The highest $F_1$ values per corpus section are portrayed in bold print.

If the results in both tables are compared to the previously discussed word embedding experiment, one aspect instantly comes to the fore: Neither the regular word embeddings experiment (table 4.3) nor the additional adding of the naive random oversampling technique (table 4.4) manage to train a sentiment analysis system for which testing results in an $F_1$ value of more than 0.5. In contrast to this, fine-tuning the *BERT* model to this sentiment task exceeds a measure of 0.5 in four cases (see table 4.5) suggesting that the power of the language knowledge incorporated in the *BERT* model might be helpful for creating a sentiment engine. Examining the results of the evaluation of the *BERT* experiment itself without associating it with other ML trainings additionally leads to further insights: First of all, observing all results of the *BERT* setup in table 4.5 makes apparent that preprocessing the training and testing data overall seems to have a negative effect on the performance of the fine-tuned *BERT* or no effect at all. For the overall $F_1$ score per corpus segment shown in table 4.5 for instance, no change in the *corpus1* and *corpus2* parts can be observed with $F_1$ values remaining at 0.24 or 0.48 respectively. All other segments in the table experience a drop in $F_1$ score when using the preprocessed instead of the unpreprocessed form of the corpus section. Table 4.5 additionally suggests that the bigger the amount of training data, the better the *BERT* model performs. The biggest corpus sections (*corpus4* with 113,589 labeled sentences and *corpus5* with 116,516 labeled examples) lead to an overall $F_1$ score ranging between 0.52 and 0.59 on respective test data and the next bigger section (*corpus2* with 2,142 labeled sentences) to 0.48. The smaller corpus segments (*corpus1* with 448 labels and *corpus3* with 783 labeled sentences) attained significantly lower $F_1$ scores in testing ranging from values between 0.18 and 0.24.

Table 4.5 – $F_1$ score of all corpus sections per sentiment class with and without preprocessing using English *BERT*.

| Corpus Section | Preprocessing | English *BERT* |
|---|---|---|
| Corpus1 | yes | **0.24** |
| Corpus1 | no | **0.24** |
| Corpus2 | yes | **0.48** |
| Corpus2 | no | **0.48** |
| Corpus3 | yes | 0.18 |
| Corpus3 | no | **0.27** |
| Corpus4 | yes | 0.52 |
| Corpus4 | no | **0.59** |
| Corpus5 | yes | 0.52 |
| Corpus5 | no | **0.57** |

To conclude, fine-tuning *BERT* still outperforms the usage of Google's word embeddings which were tested and explained in chapter 4.1.1.1. In that experiment, I first noted the high imbalance of classes inside the different corpus sections and how this could lead to a potential bias in learning. Naive random oversampling was proposed as a potential solution to overcome this bias and its evaluation results are shown in table 4.4. However, even those oversampled corpus sections only exceeded the 0.5 mark for $F_1$ in one case whereas the results of *BERT* for the bigger corpus parts (*corpus5* and *corpus4*) always exceed the 0.5 mark or come at least close to it. For this reason, using a *BERT* model and fine-tuning this for the sentiment task seems to be a better way of working with the RTL corpus than taking the language information incorporated into Google's word embeddings.

### 4.1.3 Classical ML Techniques Using tf-idf for English

The third and final experiment with the English translation of the RTL corpus is based on yet a different method for transforming textual data into vectors which the various algorithms can take to train and learn patterns from. *Tf-idf* designates the relationship between the frequency of a term $t$ and its inverted document frequency *idf*. The main idea of *tf-idf* is that the more frequent a word appears in a specific text and not in others, the more relevant it is for this document (Neumann, 2010, pp. 588-589).

*Tf-idf* is denoted by the formula

$$tf\text{-}idf_{t,d} = tf_{t,d} \times idf_t \tag{4.2}$$

(Manning et al., 2008, p. 109). It is discussed in greater detail in chapter 2.6.3 of this thesis. In contrast to the word embeddings and *BERT* experiments which were shown in previous chapters, *tf-idf* vectorization does not have any previous knowledge of the language acquired through pre-training. Instead, the input vectors for training the different ML algorithms are solely created by weighting terms in the corresponding corpus segment itself. For each respective corpus segment (*corpus1, corpus2, corpus3, corpus4* and *corpus5*), 25% were used as testing data and 75% as training data.

#### 4.1.3.1 Results of the tf-idf Experiment for English

Table 4.6 includes all results for the different algorithms taking vectors created through *tf-idf* vectorization as input. Each corpus section is once used in its preprocessed English translation and once in its unpreprocessed form for training DT, RF, SVM, KNN and LRM. The *not labeled* class was of insignificant size for both *corpus1* and *corpus2*. For this reason, this class was omitted before training. The values in bold print in table 4.6 denote the highest $F_1$ score for each corpus section. Interestingly, there is no discernible tendency towards either preprocessed or unpreprocessed data sections being better as training data and leading to higher results which were noted for the previously presented *BERT* and word embeddings experiments (see chapters 4.1.1 and 4.1.2). The two best results, i.e. an $F_1$ score of 0.51, were obtained using the two largest corpus sections in their unpreprocessed form (*corpus4* and *corpus5*) for testing.

Table 4.6 – $F_1$ score of all data sections with and without preprocessing and using *tf-idf* and the English data.

| Corpus Section | Preprocessing | DT | RF | SVM | KNN | LRM |
|---|---|---|---|---|---|---|
| Corpus1 | yes | 0.38 | 0.33 | 0.24 | **0.47** | 0.33 |
| Corpus1 | no | 0.42 | 0.25 | 0.24 | 0.31 | 0.27 |
| Corpus2 | yes | 0.40 | **0.45** | 0.29 | 0.30 | 0.40 |
| Corpus2 | no | 0.38 | 0.41 | 0.27 | 0.24 | 0.43 |
| Corpus3 | yes | 0.25 | 0.20 | 0.15 | **0.27** | 0.22 |
| Corpus3 | no | **0.27** | 0.25 | 0.15 | 0.25 | 0.19 |
| Corpus4 | yes | 0.38 | 0.38 | 0.38 | 0.30 | 0.47 |
| Corpus4 | no | 0.39 | 0.36 | 0.41 | 0.23 | **0.51** |
| Corpus5 | yes | 0.37 | 0.38 | 0.39 | 0.21 | 0.46 |
| Corpus5 | no | 0.38 | 0.36 | 0.41 | 0.26 | **0.51** |

In this and the previous sections, various ML experiments with an English translation of the RTL corpus were discussed. Comparing those with each other is interesting to examine which kind of approach is best suitable for learning to detect sentiment in the RTL corpus. Generally speaking, no real preference as to which kind of algorithm or method is best suitable can be observed. Working with word embeddings (see chapter 4.1.1), for instance, did not lead to significantly better results than *tf-idf* vectorization despite the fact that the word embeddings taken as input vectors incorporated prior language knowledge and the *tf-idf* vectors were simply created on the respective corpus segment itself. Even *BERT*, one of the most influential models in the NLP community at the time this thesis was written (see chapter 2.7.6), only performed significantly better for the larger corpus segments *corpus4* and *corpus5*. It is therefore hard to give an adequate recommendation as to which of the methods and algorithms discussed in this and previous chapters is truly suitable for working with an English translation of Luxembourgish user comments. In the next chapter, German translations of the same corpus will be taken as data for further experiments in order to examine the potential differences of using a corpus in different translated versions for sentiment detection.

## 4.2 Setup 2 - Working with Translations to German

In chapter 4.1, the setup for working with translations to English was introduced. While English is one of the resource-heaviest languages in the NLP community, it is less close to Luxembourgish than German from a linguistic point of view. This is due to shared historical origins of German and Luxembourgish (see chapter 2.10). Sentiment is, at least to some extend, culture and also language dependent. To examine whether or not this hypothesis is true, it was decided to repeat the experiments explained in chapter 4.1, but this time with a German translation of the Luxembourgish RTL corpus and not an English one. For this purpose, the same Google NMT API as for the English translations was applied (lushan88a (2020)). Table 4.7 shows the ten sentences and their translations

which were already presented in table 4.2. Interestingly, the challenges the Google NMT system faces for the Luxembourgish - German language pair are sometimes similar, but sometimes also different from the Luxembourgish - English translations. For instance, the English version of sentence one was rather good whereas this sentence was a challenge for the German translation. In this case, the Luxembourgish adverb *dann* 'then' was seen as temporal by the model and translated as *dann* 'then' [at that point] instead of 'then' [in that case] which does not lead to a correct translation. Additionally, *e sarkastesch commentaire* 'a sarcastic comment' was translated to the accusative form 'einen sarkastischen Kommentar' which is grammatically correct, but not semantically in this context. Like for the English Google NMT task, variety in spelling of Luxembourgish seems to be a problem. For the English version of sentence seven, the model treated *wëll* as the Luxembourgish adjective for 'wild'. This time, Google NMT considered it to be a form of the verb *wëllen* 'to want'. Similar to the English translation setting, syntax appears to be challenging for Google NMT. This becomes apparent in sentence ten. As for the English translation, the subject of the sentence was incorrectly identified leading to a semantically wrong German sentence. Another resemblance between the Luxembourgish - English and the Luxembourgish - German language pairs is that some mistakes already occurred on word level. *Jemp* 'name of a person' became *Jemm*, *Haus Appartement* 'house apartment' *Hauswohnung* 'house-flat', *jéimeren* 'to whine' was translated to *verwendet wird* 'get used' and *dafstomm* 'deaf-mute' as *verblüfft* 'stunned'. *Bettel*, the prime minister of Luxembourg, is a named entity that does not appear to be known to the model. In the English translation, it was translated to 'beggar' and in the German one to *Bettler* 'beggar'. Despite all those shortcomings of automatically translating correctly from Luxembourgish to German, one very interesting observation can be made in sentence six indicating that the syntactic closeness of German and Luxembourgish can help with the translation task. In this sentence, the relative clause *dat hier gehéiert* 'what belongs to it' was omitted during translation leading to a correct output which did not work in the English case. To sum up, neither English nor German translations using Google NMT yield perfect results. Nevertheless, it is a convenient way for obtaining a large amount of translated instances as training input for the ML experiments which outweighs the disadvantage of having inaccuracy in some of the translations.

Table 4.7 – Ten random Luxembourgish sentences extracted from the data set and their translation to German using Google Translate.

| Sentence in Luxembourgish | Sentences in German | Sentence ID |
|---|---|---|
| 1.) Waat ass daat do dann fir e sarkastesch commentaire. | Was ist das dann für einen sarkastischen Kommentar. | 24424ao |
| 2.) E sou e Lager soll central am Land leien dat do brengt massif Verkéier mat sech elo muss de jemp vun Elwen mat sengem Mazoutskamion bis op Käerjeng fueren fir déen ze fellen wou bleift do Logique. | Ein solches Lager soll zentral im Land liegen, was massiven Verkehr mit sich bringt. Jetzt muss der Jemm von Elwen mit seinem Tankwagen nach Bascharage fahren, um das zu finden, in dem die Logik bleibt. | 37652bb |
| 3.) Et steht awer neirens wat hei am Land en Haus Appartement asw. | Es gibt jedoch nirgendwo in diesem Land eine Hauswohnung etc. | 23446bt |
| 4.) Ee gudden Drëttel vun den Deputéierten schéngt dafstomm ze sinn gesäit een dach wéi se während de Sitzungen apathesch an hire Still hänken a nëmmen de Fanger beweege fir ofzestëmmen ouni dass jemols een eenzegt Wuert ze verlauschteren ass also Matleefer. | Ein gutes Drittel der Abgeordneten scheint verblüfft zu sein, da sie während der Sitzungen apathisch auf ihren Stühlen hängen und nur ihren Finger bewegen, um abzustimmen, ohne jemals ein einziges Wort zu hören. | 26313cc |
| 5.) " Wat den Här Bodry also ee Vertrieder vun der Koalitioun domadder mengt seet hien dann hei : | "Also, was Herr Bodry, ein Vertreter der Koalition, damit meint, sagt er hier: | 26383cc |
| 6.) Eng privat Entreprise verkeeft en Gebai dat hier gehéiert un eng aner privat Entreprise. | Eine private Firma verkauft ein Gebäude einer anderen privaten Firma. | 11221cs |
| 7.) Et geet ëm de Prinzip net ëm méi Souen a wëll et Proffen sin. | Es geht um das Prinzip, nicht um mehr Geld und darum, Professoren werden zu wollen. | 13257cu |
| 8.) Gewerkschaften hunn schonn virun e puer Deeg formell ofgestridden dat et en Accord ginn ass dat haten mir jo och schonn bei der Caritas wou den Bettel gesot huet esou guer Caritas ass op eiser Säit déi hunn dat och formell ofgestridden. | Vor einigen Tagen bestritten die Gewerkschaften förmlich, dass es eine Einigung gab, und das hatten wir bereits bei der Caritas, wo der Bettler sagte, dass sogar die Caritas auf unserer Seite ist, und sie bestritten dies förmlich. | 13400cu |
| 9.) Alkohol gehéiert op kee Fall an den Indexkuerf an "d" TVA um Alkohol misst op mannst op 25 % gehuewen gin egal wéivill "d" Horesca jéimert. | Alkohol ist keinesfalls im Indexkorb enthalten, und die Mehrwertsteuer auf Alkohol sollte auf mindestens 25% erhöht werden, unabhängig davon, wie viel Horeca verwendet wird. | 13539cu |
| 10.) Kengem Betrib hei am Land schued TVA Erhéijung direkt. | Kein Unternehmen in diesem Land schadet der Mehrwertsteuererhöhung direkt. | 13856cu |

Subsequently, I decided that the German translations were of sufficient quality to be used as input for the various experiments. The setup of those was equal to the one described in chapter 4.1 for the English version of the data. The reason for this was that the intention was to make the English and the German experiments as comparable as possible. First of all, preprocessing was carried out and each experiment was performed twice, once with an unpreprocessed and once with a preprocessed version of the respective corpus segments introduced in chapter 3.6.2. Python was used for lowercasing the sentences and user comments in the corpus segments. Additionally, I took spaCy (Honnibal et al. (2020)) to lemmatize the data and NLTK to remove stopwords from the sentences and comments

(Bird et al. (2009)). Furthermore, each (overall) sentiment was converted to a numerical value, i.e. *neutral* to 0, *negative* to 1, *positive* to 2 and *no label/undecided* to 3. The first experiment implemented used German word embeddings (see chapters 4.2.1 and 4.2.1.1), followed by an experiment with *BERT* (see chapters 4.2.2 and 4.2.2.1) and *tf-idf* vectorization (see chapters 4.2.3 and 4.2.3.1).

### 4.2.1 Word Embeddings as Input Vectors for German

The first experiment working with the German translations of the RTL corpus applied the same method that was described in chapter 4.1.1, i.e. taking a pre-trained word embeddings model in order to include additional context information into the input vectors used for training. As the algorithms and the sequence of steps were exactly the same, the explanation will not be repeated here. One crucial difference needs to be noted at this point though. The experiment explained in chapter 4.1.1 worked with the English translation of the RTL corpus and therefore required word embeddings pre-trained on a corpus written in English. Consequently, those word embeddings were replaced with 300-dimensional embeddings trained on the German Wikipedia data with the Wikipedia2Vec tool proposed in Yamada et al. (2020). It applies the skip-gram model (Mikolov et al. (2013a) with an extension in order to obtain a common vector space for words and entities together (Yamada et al., 2020, p. 3). Except for this crucial difference, this experiment was carried out in exactly the same way as the one explained in chapter 4.1.1 to allow for comparison between the two setups and their results. The training data consisted of 80% of the respective corpus segment and 20% were taken as test data.

#### 4.2.1.1 Results of the Word Embedding Experiment for German

In table 4.8, the results for all corpus setups using German word embeddings as input for training are portrayed. The $F_1$ scores range from 0.16 when taking the unpreprocessed *corpus3* section as training input for an SVM up to 0.46 for the preprocessed *corpus1* section trained with LRM. In contrast to most other experiments described in chapter 4, it is clearly visible here that taking the preprocessed version of the corpus sections as training and testing input yields higher results than when the unpreprocessed ones are employed. This becomes most apparent in the corpus section containing the user comments annotated by the author of this thesis (*corpus1*) when training is carried out with the LRM algorithm. Here, the gap between the $F_1$ for the preprocessed and the unpreprocessed sections is quite high with the $F_1$ score for the preprocessed version of the data being 0.46 whereas the unpreprocessed data obtains a value of 0.31 in evaluation. Generally speaking, it is interesting to observe that leveraging the same algorithms and the same approach of pre-trained word vectors as input can end up in different results. This can be seen clearly when comparing the word embedding set up with the English input data (see chapter 4.1.1) to the experiment described in this chapter. A potential explanation for this peculiarity is the different corpus foundations that were used for creating the respective word embeddings. More precisely, the English word embeddings were initially trained on a Google News data

set while the German embeddings took Wikipedia data as input. As will be described later in chapter 4.3.1, the Luxembourgish word embeddings used for the analysis were again trained on a different kind of corpus. In any case, neither English nor German word embeddings as input vectors lead to $F_1$ scores above 0.45/0.46. This suggest that simply relying on larger corpora as external resources is no panacea to providing a perfect solution for the automatic sentiment detection problem.

Table 4.8 – $F_1$ score of all corpus sections with and without preprocessing and with German word embeddings.

| Corpus Section | Preprocessing | DT | RF | SVM | KNN | LRM | ANN |
|---|---|---|---|---|---|---|---|
| Corpus1 | yes | 0.38 | 0.34 | 0.37 | 0.34 | **0.46** | 0.26 |
| Corpus1 | no | 0.37 | 0.39 | 0.23 | 0.38 | 0.31 | 0.23 |
| Corpus2 | yes | 0.37 | 0.39 | 0.39 | 0.40 | **0.45** | 0.43 |
| Corpus2 | no | 0.34 | 0.39 | 0.37 | 0.36 | 0.42 | 0.39 |
| Corpus3 | yes | 0.27 | 0.22 | 0.19 | 0.27 | **0.29** | 0.23 |
| Corpus3 | no | **0.29** | 0.24 | 0.16 | 0.23 | 0.22 | 0.25 |
| Corpus4 | yes | 0.33 | 0.34 | 0.31 | 0.34 | 0.38 | **0.40** |
| Corpus4 | no | 0.33 | 0.34 | 0.29 | 0.33 | 0.36 | 0.37 |
| Corpus5 | yes | 0.33 | 0.34 | 0.30 | 0.34 | 0.38 | **0.39** |
| Corpus5 | no | 0.33 | 0.34 | 0.29 | 0.34 | 0.35 | 0.36 |

### 4.2.2 German BERT

As written in chapter 4.1.2, *BERT* is a very powerful model which pre-trains "deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers." (Devlin et al., 2019, p. 4171). This pre-trained model is then fine-tuned for a specific task like sentiment analysis. The advantage of such a *BERT* model is that it incorporates knowledge about the language itself and can apply this in a beneficial way to the job in question. It was applied to all three versions of the corpus, i.e. using the translations to English, to German and the original Luxembourgish corpus. In chapter 4.1.2 and 4.1.2.1, I discussed the setup of the *BERT* experiments and the results for fine-tuning the *bert-base-cased* (Devlin et al. (2019)) to the corpus sections (see chapter 3.6.2) in their English translation. For the German version of the RTL corpus, I decided to use a pre-trained German *BERT* model (*bert-base-german-cased*, Deepset (2019)). It was trained with the German Wikipedia dump (6GB of raw txt files), the OpenLegalData dump (2.4 GB) as well as news articles (3.6 GB) (Deepset (2019)). To study how using a pre-trained German *BERT* model (*bert-base-german-cased*, Deepset (2019)) performs on the German corpus segments, the model was fine-tuned on each corpus segment separately. The segments were once taken in their preprocessed and once in their unpreprocessed form. Precisely the same parameters were employed on the English, the German and the Luxembourgish version of the corpus segments to ensure the best possible comparability between the results following Valkov (2020). Fine-tuning ran for four epochs for the *corpus1*, *corpus2* and also for the *corpus3* sections. Two epochs was the duration of fine-tuning for the

*corpus4* and *corpus5* sections. 10% of the corpus segment data were taken as testing data and 90% as training data. The batch size in each case was 16.

#### 4.2.2.1 Results of the German BERT Experiment

Table 4.9 shows the $F_1$ score for the fine-tuned German sentiment *BERT* model, divided by corpus section and preprocessing/no preprocessing. The highest $F_1$ scores are shown in bold print in table 4.9. Similar to what was already observed for the English setup, fine-tuning BERT to the sentiment task leads to better results than working with word embeddings as input vectors. The highest $F_1$ value for the German word embedding setup was 0.46 for *corpus1* in their preprocessed form trained with LRM (see table 4.8) while BERT's best result was 0.59 for the unpreprocessed *corpus4* corpus section. This suggests again, as already laid out in chapter 4.1.2.1, that the language knowledge integrated in BERT might be more helpful to training a sentiment detection system than the language knowledge part of pre-trained word embeddings. Looking at table 4.9 that comprises the overall $F_1$ score for each corpus section once fine-tuning the German *BERT* on a preprocessed version of the translated data and once on an unpreprocessed version gives additional insights. It indicates that preprocessing seems to have a negative impact on the $F_1$ score, as training and testing with the unpreprocessed versions of the data always outperform the preprocessed ones. Besides, the size of the corpus segment seems to influence the success of the learning process. For instance, the highest $F_1$ score was achieved with the unpreprocessed version of the *corpus4* segment with a value of 0.59 whereas the smallest corpus section in its unpreprocessed form, i.e. the *corpus3* one, only received a value of 0.29. Learning to generalize patterns thus seemed to work better for bigger corpus segments.

Table 4.9 – $F_1$ score of all corpus sections per sentiment class with and without preprocessing using German *BERT*.

| Corpus Section | Preprocessing | German *BERT* |
|---|---|---|
| Corpus1 | yes | 0.30 |
| Corpus1 | no | **0.31** |
| Corpus2 | yes | 0.34 |
| Corpus2 | no | **0.51** |
| Corpus3 | yes | 0.28 |
| Corpus3 | no | **0.29** |
| Corpus4 | yes | 0.51 |
| Corpus4 | no | **0.59** |
| Corpus5 | yes | 0.50 |
| Corpus5 | no | **0.57** |

In the previous section, the German translations of the RTL corpus were already used for training different ML algorithms taking external language knowledge of word embeddings as input vectors. The results of the evaluation of this experiment are to be found in table 4.8. The major aspect to be learned from comparing those results to the ones of the *BERT* experiment in table 4.5 is that *BERT* clearly outperforms the word embeddings setup for

the larger corpus sections *corpus4* and *corpus5*. Depending on the algorithm, taking word embeddings as input vectors leads to $F_1$ scores between 0.29 and 0.40 for those parts of the corpus whereas fine-tuning the German *BERT* model always results in a value of 0.50 or higher. Interestingly, the situation is different when smaller corpus sections, i.e. *corpus1, corpus2* and *corpus3*, are used. For those sections, there is no real discrepancy between the usage of word embeddings or *BERT*. The performance is similar. Both word embeddings and *BERT* incorporate prior knowledge about the language (in this case German) into the training process. It is therefore fascinating to perceive a difference in performance depending on the size of the corpus section and also the kind of language information.

Due to the data taken for this experiment being the same as for the first *BERT* experiment with translations to English (see chapter 4.1.2), I decided to compare both outputs as well, i.e. the $F_1$ scores of table 4.5 with the ones in table 4.9. The only difference between the two corpus sets is that it was once translated to English and once to German with the Google Translate API of lushan88a (2020). All sentiment labels remained the same and were directly taken from the original annotation task on the Luxembourgish sentiment corpus. Additionally, both the English and the German versions of the corpus were preprocessed with the same preprocessing steps, i.e. removal of stopwords, lemmatization, tokenization and lowercasing. Studying and comparing all $F_1$ scores of both tables with each other exposes interesting insights into the data. The best performing models for each corpus segment in both languages were the ones which worked with unpreprocessed training data. In two cases (*corpus1* and *corpus2*) with English *BERT*, preprocessing did not make any difference and lead to the same evaluation result. In NLP projects, preprocessing techniques are frequently used (Camacho-Collados and Pilehvar, 2018, p. 40) and can be seen as an essential part of the framework of conventional text classification (Uysal and Gunal, 2014, p. 104). The impact of various text preprocessing steps in neural models, on the other hand, has not been as extensively studied in the literature (Camacho-Collados and Pilehvar, 2018, p. 40). Nevertheless, (Camacho-Collados and Pilehvar, 2018, p. 41) state that preprocessing steps such as lemmatization and lowercasing might influence the performance in a negative way. Both the German (*bert-base-german-cased*, (Deepset (2019))) and the English (*bert-base-cased*, (Devlin et al. (2019))) version of the BERT model which the author of this thesis worked with included cased tokens, an aspect that got removed from the preprocessed RTL corpus data in its English as well as its German version. While more research would have to be done, this could be a potential reason as to why fine-tuning and evaluating the performance on an unpreprocessed version of the respective corpus section generally yields higher results.

To sum up, I fine-tuned a case-sensitive German *BERT* model (Deepset (2019)) for different translated sections of the corpus. Compared to the first experiment implemented on an English version of the RTL corpus, a rise in $F_1$ score was mostly perceived. This suggests that *BERT* and all the prior knowledge about the German language it includes is very helpful for detecting sentiment in the data. However, comparing the results of *corpus4* and *corpus5* for the English and the German *BERT* (see tables 4.5 and 4.9) shows that there is almost no difference in the $F_1$ score. The fine-tuning of *BERT* therefore seems to be promising, but a clear remark whether English or German, which is closer to Luxembour-

gish, is better suitable for sentiment detection on the RTL corpus cannot be made. As was briefly described in chapter 4.2.2, the hyper-parameters defined by Valkov (2020) were not changed and fine-tuning was only carried out for two or four epochs respectively. Future work could therefore include examining the impact of tuning different hyper-parameters of the network. Furthermore, fine-tuning could be undertaken for more epochs to find the most adequate fine-tuned model for the data. That way, the performance of the trained system could potentially be boosted even higher. However, it should also not be forgotten at this point that all analyses carried out with German texts, as well as with English data, were implemented on automatically translated versions. I have studied some of the translations to German (chapter 4.2) and English (chapter 4.1) and especially the Luxembourgish to English translations seem to be of sufficient quality. Nevertheless, manually reading just a couple of automatically translated sentences is not enough to be sure that all data instances are translated with a sufficiently high quality. The low quality of translations might potentially have a negative influence on the quality of the *BERT* system itself. In order to examine this further, a third experiment with *BERT* is implemented working with a multilingual *BERT* model including Luxembourgish to discuss the performance of *BERT* on untranslated sentences and comments. Those findings will be discussed in chapter 4.3.2.

### 4.2.3 Classical ML Techniques Using tf-idf for German

As written in chapters 2.6.3 and 4.1.3, *tf-idf* is a method to transform a textual corpus into computer-readable vectors based on the connection between the term frequency of $t$ and the inverted document frequency *idf*. Subsequently, a word/term's importance for a document depends on how often it appears in that text and not in others (Neumann, 2010, pp. 588-589). In the following section, the results for *tf-idf* vectorization for the different corpus sections of the RTL data previously translated to German are discussed. For each corpus section (*corpus1, corpus2, corpus3, corpus4* and *corpus5*), 25% were used as testing and 75% as training data.

#### 4.2.3.1 Results of the tf-idf Experiment for German

Table 4.10 shows the $F_1$ score for the *tf-idf* vectorization of the different corpus sections that were translated to German, once in a preprocessed and once in an unpreprocessed form. DT, RF, SVM, KNN and LRM were taken as algorithms for training and testing. The bold print values denote the best evaluation score per corpus section. The first observation to be made is that the $F_1$ scores calculated for the different corpus sections are in a wide range from 0.15 (*corpus3*, not preprocessed, SVM) to 0.51 (*corpus4*, not preprocessed, LRM). Additionally, it can be noted that there is a tendency towards the unpreprocessed version of a corpus section being more suitable for training a sentiment detection system than the preprocessed one. For three of the five possible corpus sections, performing sentiment detection with the unpreprocessed version of the data lead to higher $F_1$ scores.

A similar experiment taking an English translation of the same corpus sections was implemented and previously described in chapter 4.1.3.1. Comparing the $F_1$ scores obtained for

Table 4.10 – $F_1$ score of all data sections with and without preprocessing and using *tf-idf* and the German data.

| Corpus Section | Preprocessing | DT | RF | SVM | KNN | LRM |
|---|---|---|---|---|---|---|
| Corpus1 | yes | 0.31 | 0.30 | 0.24 | **0.41** | 0.34 |
| Corpus1 | no | 0.24 | 0.28 | 0.24 | 0.32 | 0.30 |
| Corpus2 | yes | 0.40 | 0.39 | 0.33 | 0.23 | 0.40 |
| Corpus2 | no | 0.38 | 0.40 | 0.27 | 0.24 | **0.42** |
| Corpus3 | yes | 0.25 | 0.28 | 0.17 | **0.30** | 0.24 |
| Corpus3 | no | 0.26 | 0.19 | 0.15 | 0.23 | 0.19 |
| Corpus4 | yes | 0.38 | 0.38 | 0.38 | 0.27 | 0.46 |
| Corpus4 | no | 0.38 | 0.36 | 0.41 | 0.26 | **0.51** |
| Corpus5 | yes | 0.38 | 0.39 | 0.38 | 0.20 | 0.47 |
| Corpus5 | no | 0.37 | 0.36 | 0.41 | 0.17 | **0.50** |

those tests with the ones for the exact same RTL corpus in its German translation shows little differences (see tables 4.6 vs. 4.10). On the other hand, comparing the *tf-idf* vectorization experiment described here with the German word embeddings (chapter 4.2.1) and the German *BERT* (chapter 4.2.2) one makes it possible to come up with recommendations as to which approach might be the most suitable for building a successful sentiment analysis system. The usage of word embeddings seems to lead to a drop in performance compared to *tf-idf* vectorization. This conclusion can be drawn from the observation that no $F_1$ value in table 4.8 exceeds 0.50. *BERT*, as shown in table 4.9, learns to detect sentiment patterns slightly better than a *tf-idf* vectorization approach, especially for the larger corpus sections *corpus5* and *corpus4*. For this reason, future work with the German translation of the RTL corpus segments could therefore consist of fine-tuning the parameters of *BERT* better to the needs of the project. One very important aspect should, however, not be forgotten in this context. So far, all experiments carried out and described were done on a translated version of the original corpus. While some of the tendencies shown lead into promising directions, translations always come with a price: the introduction of errors into the original data and potential loss of (linguistic) information. The last part of the experimental setup is subsequently dedicated to working with the original RTL corpus in its Luxembourgish form.

## 4.3 Setup 3 - Working with Luxembourgish Data

So far, various models have been trained and tested to examine the performance of a variety of different classification algorithms and I have discussed those results. However, none of those were implemented directly on the real corpus data obtained from the partner RTL Luxembourg of the *STRIPS* project. Instead, automatically translated instances obtained applying a Python implementation of Google NMT (lushan88a (2020)) for translating to English and German were used for *setup 1* and *setup 2* (see chapters 4.1 and 4.2). The reason for this choice was that languages that are not low-resource like Luxem-

bourgish have more resources at their disposal such as large pre-trained language models like *BERT* (Devlin et al. (2019)) and a variety of preprocessing tools. While studying different techniques and their output with translated corpus data is certainly interesting, every translation comes with a price of loosing some information or potentially changing the initially intended meaning of a claim. This point was illustrated in chapters 4.1 and 4.2 when examining the difficulties and mistakes in ten randomly chosen translated sentences from the corpus. In more general terms, concepts such as sarcastic affirmations or very cultural specific topics are hard to translate correctly, especially by automatic means that lack adequate training and cultural background knowledge. For those reasons, the experiments described in the preceding chapters using German and English translated instances of the corpus are not sufficient to adequately study the expression of sentiment in Luxembourgish and how this concept can be taught to be automatically detected by an ML algorithm.

Before diving into the setup of the ML experiments performed on the Luxembourgish original corpus data, the ten randomly drawn sentences from the RTL corpus which were already discussed in tables 4.2 and 4.7 will first be looked. One column presents the original ten random sentences of the corpus, one the preprocessed and one the manually corrected form done by a native speaker of Luxembourgish who is very familiar with the official spelling rules of the language (see table 4.11). In contrast to English and German, only two preprocessing tools exist for Luxembourgish, LuNa (Sirajzade and Schommer (2019)) and spellux (Purschke (2020b)). Due to the chosen data format being more easily compatible with spellux, this tool was chosen to preprocess the RTL corpus. Spellux was specially developed to help normalize Luxembourgish statements. It was used to remove stopwords, normalize the spelling and take care of cases in which the *-n* of a word should be omitted following the so-called *n-rule*. This rule, which is a challenging and yet also interesting particularity to the Luxembourgish language, is further explained in chapter 2.10. Furthermore, lowercasing of all sentences and user comments in the corpus was performed as preprocessing steps.

Table 4.11 – Ten random Luxembourgish sentences extracted from the corpus and their correction using spellux and lower casing.

| Sentence in Luxembourgish | Preprocessed Sentence | Sentence Corrected by a Native Speaker | Sentence ID |
|---|---|---|---|
| 1.) Waat ass daat do dann fir e sarkastesch commentaire. | wat dat sarkastesch commentaire. | Wat ass dat do da fir e sarkastesche Commentaire. | 24424ao |
| 2.) E sou e Lager soll central am Land leien dat do brengt massif Verkéier mat sech elo muss de jemp vun Elwen mat sengem Mazoutskamion bis op Käerjeng fueren fir déen ze fellen wou bleift do Logique. | e lager soll zentral land leie bréngt massiv verkéier elo jemp ëlwe mazoutskamion käerjeng fueren dée wëlle bleift logiciel. | Esou e Lager soll zentral am Land leien. Dat do bréngt massiv Verkéier mat sech. Elo muss de Jemp vun Ëlwen mat sengem Mazoutscamion bis op Käerjeng fueren, fir deen ze fëllen. Wou bleift do d'Logik? | 37652bb |
| 3.) Et steht awer neirens wat hei am Land en Haus Appartement asw. | et steet néierens land haus appartement asw. | Et steet awer néierens, wat hei am Land en Haus, Appartement asw. | 23446bt |
| 4.) Ee gudden Drëttel vun den Deputéierten schéngt dafstomm ze sinn gesäit een dach wéi se während de Sitzungen apathesch an hire Still hänken a nëmmen de Fanger beweege fir ofzestëmmen ouni dass jemols een eenzegt Wuert ze verlauschteren ass also Matleefer. | ee gudden drëttel deputéierte schéngt dafstomm gesäit während sitzungen apathesch hire still hänke fanger beweege ofzestëmmen dass jeemools eenzegt wuert verlauschtere matleefer. | Ee gudden Drëttel vun den Deputéierte schéngt dafstomm ze sinn. Gesäit een dach, wéi se wärend de Sitzungen apathesch an hire Still hänken an nëmmen de Fanger beweegen, fir ofzestëmmen, ouni dass jeemools een eenzegt Wuert ze verlauschteren ass. Also Matleefer. | 26313cc |
| 5.) " Wat den Här Bodry also ee Vertrieder vun der Koalitioun domadder mengt seet hien dann hei : | " wat här bodry vertrieder koalitioun domadder mengt seet: | Wat den Här Bodry, also e Vertrieder vun der Koalitioun, domadder mengt, seet hien dann hei: | 26383cc |
| 6.) Eng privat Entreprise verkeeft en Gebai dat hier gehéiert un eng aner privat Entreprise. | eng privat entreprise verkeeft gebai gehéiert aner privat entreprise. | Eng privat Entreprise verkeeft e Gebai, dat hir gehéiert un eng aner privat Entreprise. | 11221cs |
| 7.) Et geet ëm de Prinzip net ëm méi Souen a wëll et Proffen sin. | et prinzip sue wëll proffe sinn. | Et geet ëm de Prinzip, net ëm méi Suen a well et Proffe sinn. | 13257cu |
| 8.) Gewerkschaften hunn schonn virun e puer Deeg formell ofgestridden dat et en Accord ginn ass dat haten mir jo och schonn bei der Caritas wou den Bettel gesot huet esou guer Caritas ass op eiser Säit déi hunn dat och formell ofgestridden. | gewerkschafte viru puer deeg formell ofgestridden accord caritas bettel gesot guer caritas säit formell ofgestridden. | D'Gewerkschaften hu scho virun e puer Deeg formell ofgestridden, datt et en Accord ginn ass. Dat hate mir jo och scho bei der Caritas, wou de Bettel gesot huet, esouguer d'Caritas ass op eiser Säit, déi hunn dat och formell ofgestridden. | 13400cu |
| 9.) Alkohol gehéiert op kee Fall an den Indexkuerf an "d" TVA um Alkohol misst op mannst op 25 % gehuewen gin egal wéivill "d" Horesca jéimert. | alkohol gehéiert fall indexkuerf" d" tva alkohol misst mannst 25 % gehuewe ginn egal wéivill" d" horesca jéimert. | Alkohol gehéiert op kee Fall an den Indexkuerf an d'TVA um Alkohol misst op d'mannst 25% gehuewe ginn, egal wéi vill d'Horesca jéimert. | 13539cu |
| 10.) Kengem Betrib hei am Land schued TVA Erhéijung direkt. | kengem betrib land schued tva erhéijung direkt. | Kengem Betrib hei am Land schuet TVA-Erhéijung direkt. | 13856cu |

Comparing the first column (sentence in Luxembourgish) with the third column (sentence corrected by a native speaker) exhibits a couple of typical mistakes which are frequently found in text productions of Luxembourgish-speakers who have not studied the orthography of the language. The first one is the non-respecting of the n-rule (Zenter fir d'Lëtzebuerger Sprooch, 2019, pp. 46-47). Examples of this behavior can be found in sentence one (*dann* instead of *da* 'then'), sentence four (*Deputéierten* instead of *Deputéierte* 'deputy'), sentence six (*en* instead of *e* 'a') and sentence eight (*schonn* instead of *scho* 'already') and *haten* instead of *hate* 'have'). In contrast to German, Luxembourgish does not foresee a letter *h* to extend the length of the preceding vowel (Zenter fir d'Lëtzebuerger Sprooch, 2019, p. 32). This kind of spelling mistake can be found in the third (*steht* instead of *steet*) and fourth sentence (*während* instead of *wärend*). Other orthographic mistakes are the (non-) doubling of consonants (Zenter fir d'Lëtzebuerger Sprooch, 2019, p. 27-28), like in sentence seven (*sin* instead of *sinn* 'be'), eight (*dat* instead of *datt* 'that') or nine (*gin* instead of *ginn* 'to give, I/we/they give'), and the incorrect usage of diacritics, as for example in the second sentence (*brengt* instead of *bréngt* 'bring', *Elwen* and not *Ëlwen* 'Troisvierges: name of a Luxembourgish town', *déen* instead of *deen* 'that' and *fellen* instead of *fëllen* 'fill'). Furthermore, spelling mistakes resulting from the influence of German or French, final devoicing (Zenter fir d'Lëtzebuerger Sprooch, 2019, p. 30-31), homonyms and omitting of the article *d'* can be observed. Examples for those cases are *Logique* (correct spelling *Logik* 'logic') in the second sentence, *schued* (correct spelling *schuet* 'harm') in the tenth sentence, *hier* 'here' instead of *hir* 'hers' in sentence six and *Gewerkschaften* (instead of *D'Gewerkschaften* 'trade union') in the eight sentence. Examining the preprocessed versions of the respective sentence in the column in the middle shows that, in general, the spelling correction carried out with the spellux tool (Purschke (2020b)) worked well. Most spelling mistakes were identified and corrected. However, spellux does not work perfectly yet. For instance, it did not correlate *während* 'during' with the correct spelling *wärend* in the fourth sentence and *schued* 'damage' with the intended *schuet* 'harm' in the tenth sentence. Mistakes like the last one are especially difficult to automatically find though. As both *schued* and *schuet* are valid words in Luxembourgish, external knowledge would be required to adequately determine which candidate word was intended by the writer of the sentence. To sum up, studying those ten randomly drawn sentences from the RTL corpus illustrate the amount of spelling mistakes in the data used for the ML algorithms. Taking the spellux tool (Purschke (2020b)) leads to an improvement in correctness of spelling. It will therefore be interesting to see in the upcoming experiments whether or not this removing of some of the spelling variation can improve the way algorithms learn to detect sentiment in the corpus.

Like for the English and German translations, each experiment was implemented with each corpus segment (see chapter 3.6.2) twice - once with the segment in its unpreprocessed and once in its preprocessed form. Analogous to the English and German data instances, each (overall) sentiment in each corpus segment was converted from a textual to a numerical value. That way, *neutral* was transformed to 0, *negative* to 1, *positive* to 2 and *not labeled/undecided* to 3. The Luxembourgish experiments started with a word embedding setup (see chapters 4.3.1 and 4.3.1.1). The second kind of experiment used a Multilingual *BERT* (see chapters 4.3.2 and 4.3.2.1) and the third a *tf-idf* vectorization process (chapters

4.3.3 and 4.3.3.1). In contrast to the work previously discussed on the translated English and German corpus segments, the Luxembourgish experimental part also includes the study of the influence of various linguistic features on classification performance (chapters 4.3.4 and 4.3.4.2).

### 4.3.1 Word Embeddings for Luxembourgish

The first experiment with the RTL corpus in its original Luxembourgish form was, similar to the first experiments of the first and second setups with English and German translations of that same data, carried out with word embeddings to incorporate additional context information into the training process (see chapters 4.1.1 and 4.2.1). Once again, Galeshchuk et al. (2019)'s method was followed which was depicted in chapter 4.1.1. 20% of the respective corpus segment data were used for testing and 80% as training data. In contrast to the German and English experiments for which I took pre-trained word embedding models, this experiment required to train an own one for the Luxembourgish language. A corpus of roughly 70,000,000 tokens was taken including a diverse collection of Luxembourgish texts composed by the supervisor of this doctoral thesis to train a 64-dimensional Luxembourgish word2vec word embedding model (Mikolov et al. (2013a)).

#### 4.3.1.1 Results of the Word Embedding Experiment for Luxembourgish

Table 4.12 shows the results for the Luxembourgish version of the word embedding experiment using various corpus sections in preprocessed and unpreprocessed form and DT, RF, SVM, KNN, LRM and ANN as algorithms for training. For those word embeddings, unpreprocessed data works best, as those versions of the respective corpus section mostly perform better than the preprocessed ones. They are highlighted in bold print in table 4.12. Similar to the other word embedding experiments with English and German translations of the RTL corpus, the $F_1$ scores obtained are in a wide range from 0.16 (unpreprocessed *corpus3* section, training and testing with SVM) to 0.50 (unpreprocessed *corpus4* section, training and testing with ANN). A tendency towards the ANN algorithm being the best fit for training can be observed. What is most interesting about this version of the word embeddings experiment is that, despite the German and English translations of the RTL corpus disposing of more resources for preprocessing and also larger pre-trained word embeddings, the Luxembourgish experiment actually yielded better results. Whereas the highest values for the $F_1$ score were 0.45 for the English word embeddings experiment not using oversampling and 0.46 for the German word embeddings setup, using an unpreprocessed version of the *corpus4* section and training with an ANN lead to a value of 0.50 closely followed by taking *corpus5* in its unpreprocessed form as training input for an ANN ($F_1$ value of 0.49). This finding suggests that relying on the external knowledge from bigger corpora and incorporating this into the training approach through pre-trained word embeddings might not actually be the most successful way of dealing with automatic sentiment detection problems, at least for the Luxembourgish RTL corpus case.

Table 4.12 – $F_1$ score of all corpus sections with and without preprocessing and with Luxembourgish word embeddings.

| Corpus Section | Preprocessing | DT | RF | SVM | KNN | LRM | ANN |
|---|---|---|---|---|---|---|---|
| Corpus1 | yes | 0.32 | 0.39 | 0.36 | 0.31 | 0.44 | 0.39 |
| Corpus1 | no | **0.47** | 0.40 | 0.23 | 0.25 | 0.45 | 0.34 |
| Corpus2 | yes | 0.36 | 0.38 | 0.32 | 0.36 | 0.40 | 0.39 |
| Corpus2 | no | 0.36 | **0.42** | 0.39 | 0.40 | **0.42** | **0.42** |
| Corpus3 | yes | 0.20 | 0.22 | 0.17 | 0.22 | 0.25 | 0.23 |
| Corpus3 | no | 0.22 | 0.25 | 0.16 | 0.22 | **0.26** | **0.26** |
| Corpus4 | yes | 0.35 | 0.37 | 0.32 | 0.36 | 0.39 | 0.42 |
| Corpus4 | no | 0.37 | 0.39 | 0.35 | 0.38 | 0.42 | **0.50** |
| Corpus5 | yes | 0.34 | 0.37 | 0.31 | 0.37 | 0.37 | 0.41 |
| Corpus5 | no | 0.37 | 0.39 | 0.33 | 0.38 | 0.41 | **0.49** |

### 4.3.2 Multilingual BERT (Luxembourgish)

To the best of my knowledge and contrary to German and English, no *BERT* model solely pre-trained on the Luxembourgish language has been published yet. However, Luxembourgish is part of the *bert-base-multilingual-cased* model first proposed in Devlin et al. (2019). Multilingual *BERT* was "pre-trained on the concatenation of monolingual Wikipedia corpora from 104 languages" (Pires et al., 2019, p. 4996). It therefore only has one multilingual vocabulary for all languages (Pires et al., 2019, p. 4997). Interestingly, Multilingual *BERT* is even capable of generalizing across languages with different scripts to some extend (Pires et al., 2019, p. 5000). Working with a model that incorporates so many different languages seems challenging at first glance, as it means that information from very different language families with diverse scripts and grammars are taken as one single element of background knowledge for a task. This makes this version of the *BERT* experiment particularly interesting and different from the previously described English and German *BERT* setups (see chapters 4.1.2 and 4.2.2). The practical implementation was the same as for the previously discussed experiments with an English and a German version of *BERT* (see chapters 4.1.2, 4.1.2.1, 4.2.2, 4.2.2.1) with fine-tuning *BERT* for each of the corpus segments presented in chapter 3.6.2. This ensured the best possible comparability between the different *BERT* models for English, German and Luxembourgish. Fine-tuning ran for four epochs for the smaller corpus sections (*corpus1*, *corpus2* and *corpus3*) and for two epochs for the bigger corpus segments (*corpus4* and *corpus5*) with a batch size of 16. For each corpus segment, 10% were taken as testing data and 90% as training data.

#### 4.3.2.1 Results of the Multilingual BERT Experiment

Table 4.13 shows the overall $F_1$ score for each corpus section, once in its preprocessed and once in its unpreprocessed form. The highest $F_1$ scores per corpus section are presented in bold print. In the previous section, an experiment applying Luxembourgish word embeddings was described and evaluated. Comparing the results of the word embeddings experiment in table 4.12 with the ones of multilingual *BERT* in table 4.13 yields first interesting insights into the data. Evaluating the performance of the multilingual *BERT* for the *corpus3* section results in an $F_1$ value of 0.11 for both the preprocessed and the unpreprocessed version of the data. This value is lower than the one achieved for the same corpus section in its unpreprocessed form taking word embeddings as input vectors ($F_1$ score of 0.26, see table 4.12). Generally speaking, the multilingual *BERT* model does not outperform the word embedding experimental setup described in chapter 4.3.1, except when fine-tuning occurred with the unpreprocessed *corpus4* section which obtained a 0.59 $F_1$ value for multilingual *BERT* vs. 0.50 or lower for the word embeddings setup. This is in contrast to *setup 1* and *setup 2* in which I implemented both a word embeddings and a *BERT* experiment with a translated version (either to English or to German) of the RTL corpus and usually observed a better result for *BERT* than for word embeddings (see chapters 4.2.2.1 and 4.1.2.1).

Table 4.13 – $F_1$ score of all corpus sections with and without preprocessing using Multilingual *BERT*.

| Corpus Section | Preprocessing | Multilingual *BERT* |
|---|---|---|
| Corpus1 | yes | **0.24** |
| Corpus1 | no | **0.24** |
| Corpus2 | yes | 0.28 |
| Corpus2 | no | **0.37** |
| Corpus3 | yes | **0.11** |
| Corpus3 | no | **0.11** |
| Corpus4 | yes | 0.35 |
| Corpus4 | no | **0.59** |
| Corpus5 | yes | 0.42 |
| Corpus5 | no | **0.51** |

Taking a closer look at the evaluation results of multilingual *BERT* per corpus section reveals that the span of results is quite large, ranging from 0.11 (*corpus3* section) to 0.59 (*corpus4* in the unpreprocessed form). The bigger the corpus segment, the better the learning and thus generalization of sentiment structures seems to work. While the smallest corpus segments (*corpus1* and *corpus3*) obtained an $F_1$ score of 0.24 (*corpus1*) and 0.11 (*corpus3*), using the unpreprocessed form of the second largest corpus section (*corpus4*) for fine-tuning and testing lead to an $F_1$ value that was more than twice as high (0.59). As observed already in previous experiments, for instance for the German and the English *BERT* experiments (see chapters 4.2.2 and 4.1.2), a negative effect of preprocessing can be noted. The biggest negative impact was on the *corpus4* section. While fine-tuning and

testing with the preprocessed version got an overall $F_1$ score of 0.35, the unpreprocessed version of that same corpus part scores more than 0.2 higher with an $F_1$ of 0.59.

Most interesting about this examination is to compare its result with the ones obtained from fine-tuning the German and the English *BERT* to see whether the language model itself impacts the performance of the classifier carrying out the sentiment detection task. Those two experiments are discussed in chapters 4.1.2, 4.1.2.1, 4.2.2 and 4.2.2.1. In a first step, the $F_1$ scores of the English *BERT* (see table 4.5) and the Multilingual *BERT* (table 4.13) will be studied next to each other. Except for the smallest data section (*corpus1*) and the unpreprocessed version of *corpus4*, the English *BERT* always outperforms the multilingual version. This suggests the utility of a large pre-trained model on a single language. Examining the results of the German BERT experiment (see table 4.9) supports this claim as it outperforms the Multilingual *BERT* even further. The only exception to this is the unpreprocessed *corpus4* segment which receives the same $F_1$ score of 0.59 in both cases, a surprising finding which I could find no explanation for. It does not seem surprising that both the English and the German *BERT* models' performances surmount the Multilingual *BERT* one. Whereas the English and German *BERT* were exclusively trained on data of one language, Luxembourgish is part of a multilingual model that includes very different languages, of which some do not even have the same scripts or word orders (Pires et al., 2019, p. 4998-4999). Incorporating such a large span of information into one single model should naturally be harder than training for a single language with specific features. From a linguistic point of view, a drop in performance when working with the multilingual *BERT* therefore makes sense. Potential future work could therefore include the creation of a monolingual Luxembourgish *BERT* model and repeating the experiment presented in this chapter.

### 4.3.3 Classical ML Techniques Using tf-idf for Luxembourgish

This experiment is based on a different method for transforming a corpus of textual data into vectors for training and testing ML algorithms. The approach used for this experiment is *tf-idf* of which the main idea is to study the frequency of words in a text. The more frequent such a word is in a specific text, the more relevant it might be for characterising this document (Neumann, 2010, pp. 588-589), see also chapter 2.6.3). 25% were used as testing and 75% as training data for each corpus section.

#### 4.3.3.1 Results of the tf-idf Experiment for Luxembourgish

Table 4.14 displays the results of the different algorithms using vectors obtained from *tf-idf* vectorization. Like in all other experiments, each corpus section presented in chapter 3.6.2 was once taken in its preprocessed and once in its unpreprocessed form. The algorithms for training and testing were DT, RF, SVM, KNN and LRM with the same parameters as in the previous experiment with English word embeddings as input vectors (see chapter 4.1.1.1) as well as the other *tf-idf* experiments on English and German translations (chapters 4.1.3 and 4.2.3). The parameters were not changed in order to assure better comparability between

the results. As the *not labeled* class was very small for both the *corpus1* and *corpus2* sections, it was removed before training. The highest $F_1$ score per section is presented in bold print in the table.

Table 4.14 – $F_1$ score of all corpus sections with and without preprocessing using *tf-idf* and the Luxembourgish data.

| Corpus Section | Preprocessing | DT | RF | SVM | KNN | LRM |
|---|---|---|---|---|---|---|
| Corpus1 | yes | 0.31 | 0.29 | 0.24 | 0.35 | 0.32 |
| Corpus1 | no | **0.39** | 0.26 | 0.24 | 0.38 | 0.28 |
| Corpus2 | yes | 0.40 | 0.41 | 0.30 | 0.20 | 0.43 |
| Corpus2 | no | 0.36 | 0.40 | 0.26 | 0.27 | **0.45** |
| Corpus3 | yes | **0.32** | 0.22 | 0.15 | 0.20 | 0.19 |
| Corpus3 | no | 0.23 | 0.19 | 0.15 | 0.15 | 0.18 |
| Corpus4 | yes | 0.40 | 0.40 | 0.46 | 0.27 | 0.52 |
| Corpus4 | no | 0.43 | 0.39 | 0.59 | 0.24 | **0.60** |
| Corpus5 | yes | 0.40 | 0.40 | 0.45 | 0.10 | 0.52 |
| Corpus5 | no | 0.42 | 0.39 | 0.57 | 0.12 | **0.59** |

The results of the $F_1$ score in table 4.14 are in a wide range, from 0.10 (*corpus5*, preprocessed, KNN) to 0.60 (*corpus4*, unpreprocessed, LRM). What is especially interesting in this overview of results is that the best performing classifier trained per corpus section is usually one that was trained and tested with the unpreprocessed data. The only exception to this is the *corpus3* set in which DT trained on preprocessed data scored the highest (0.32). I did not expect this behavior before running the different trainings and on the contrary anticipated a positive behavior of preprocessing, as *tf-idf* vectors were created from the input texts of the respective corpus section and preprocessing them with spellux (Purschke (2020b)) as well as lower casing the corpus, as described in chapter 4.3, removed at least a part of the large variety of spelling variation which is so particular to the Luxembourgish language (see chapter 2.10). Why is it then that this reduction of variation does not lead to an improvement of classification results? Toman et al. (2006) analyzed the impact of word normalization techniques on both a Czech and an English corpus. They found out that word normalization has a negative and not a positive impact on performance and subsequently suggest that solely removing stopwords from a corpus and not applying normalization at all could be the best preprocessing approach (Toman et al. (2006)). Furthermore, Uysal and Gunal (2014) studied the impact of preprocessing in more general terms on classification of texts. They worked with both English and Turkish corpus sets. One of their findings was that the choice of preprocessing techniques can have a big influence on the accuracy of a classifier (Uysal and Gunal, 2014, p. 109). Additionally, they noted that stopwords should not necessarily be removed from a data set and that generally speaking, lowercasing as part of the preprocessing is a valuable idea (Uysal and Gunal, 2014, p. 110). Most importantly though, they stated that there is no combination of different preprocessing tasks perfectly adequate for every corpus and every kind of classification problem. Therefore, they infer that the choice of preprocessing should be carefully made by the researcher as this can change the performance of an algorithm

(Uysal and Gunal, 2014, p. 111).

With respect to the findings made by Toman et al. (2006) and Uysal and Gunal (2014), future research should certainly be made on the choice of preprocessing in order to ensure that it helps to improve the classification results and does not, on the contrary, harm performance due to a potentially inadequate preprocessing step for the corpus present. One first preprocessing step whose impact on performance could easily be examined is the lower casing or non-lower casing of the corpus. For this thesis, the decision was taken to lower case during preprocessing, may it be for the English, German or Luxembourgish version of the corpus. Nevertheless, it should be noted that casing can certainly carry sentiment. For instance, *Ech sinn esou genervt vun dir!* 'I am so annoyed by you!' is certainly *negative*, but *ECH SINN ESOU GENERVT VUN DIR!* 'I AM SO ANNOYED BY YOU!' even more so.

Another interesting finding from this experiment becomes apparent when comparing the results with the scores obtained in table 4.3 and 4.4. Those two tables show the results for the English experiment using word embeddings as input features. The same parameters and the same algorithms for the word embedding experiment working with English translations were used here. The crucial difference between the two ways of training was not only the choice of language (English vs. Luxembourgish), but more the method for transforming the input text to a vector representation that algorithms can understand and learn from. Intuitively, I would have expected the word embeddings of the English setup (see chapter 4.1.1) to outperform the *tf-idf* vector representations. The reason for this expectation was that the English setup averaged a large pre-trained word2vec model for the English language (Řehůřek and Sojka (2010)) which includes by far more knowledge about the language in itself than *tf-idf* vector representations. Last but not least, comparing the *tf-idf* results in table 4.14 with the word embeddings (table 4.12) and multilingual *BERT* (table 4.13) experiment suggests that *tf-idf* might be the best choice for working directly with the Luxembourgish version of the RTL corpus. In the next chapter, this *tf-idf* vectorization approach will be further extended by combining it with a variety of linguistic features to study their potential importance for carrying and transmitting sentiment to its readers.

### 4.3.4 Using Linguistic Features

The experiments described in the preceding chapters made use of translations to English (*setup 1*, see chapter 4.1), German (*setup 2*, see chapter 4.2), and different vectorization techniques (*tf-idf* (see chapters 4.1.3, 4.2.3, 4.3.3), *word2vec* (see chapters 4.1.1, 4.2.1, 4.3.1)) and pre-trained models (*BERT*, see chapters 4.1.2, 4.2.2, 4.3.2). In this chapter, an additional kind of features that has not been used in the experiments presented so far will be introduced: the class of linguistic features. Taking linguistic information for (automatically) detecting sentiment in a corpus has a long-standing tradition. Early research include working with adjectives (Hatzivassiloglou and McKeown (1997)), a POS that "were considered the most important features in sentiment analysis" (Farzindar and Inkpen, 2015, pp. 49-50). Other POS significant for expressing sentiment are nouns, verbs and adverbs (Taboada, 2016, p. 329). Additionally, sentiment shifters such as negation

words are interesting for expressing sentiment, as they can reverse the intended positivity or negativity (Liu, 2015, p. 116).

In the following experiment, those linguistic features will be used in different combinations in order to study their impact on the learning of expressions of sentiment in Luxembourgish. Building upon the component analysis of adjectives and their influence on the voicing of sentiment in Luxembourgish (Sirajzade et al. (2020b)), more features going beyond the simple adjective-level and data were applied in this experiment. A short introduction to them as well as to the Luxembourgish language itself was given in chapter 2.10. The POS information was automatically added as *XML* tags to the different corpus sections with the LuNa corpus tool (Sirajzade and Schommer (2019), see also chapter 3.1). Like for all other experiments discussed in the previous chapters, all the following experiments were carried out on the different corpus sections described in chapter 3.6.2. For each of those corpus segments, different feature combinations were taken to examine their impact on the detection of sentiment. More precisely, the features *SENT* (the whole sentence)*, *ADJ* (adjective(s)), *ADV* (adverb(s)), *V* (verb(s)), *N* (noun(s)), *NEGATION* (negation(s)), *ADJ_NEGATION* (adjective(s) and negation(s)), *SENT_NEGATION* (sentence and negation(s)), *V_NEGATION* (verb(s) and negation(s)) and *all features* (combining all the mentioned features) were taken as input representation for the DT, RF, SVM, KNN and LRM algorithms. No preprocessing steps were carried out on the different corpus segments. There were four possible labels, i.e. *negative, neutral, positive* and *undecided/not labeled*. The label *undecided/not labeled* was omitted for the *corpus1* and *corpus2* sections, as there were only one instance in *corpus1* and two in *corpus2* labeled in that way. Additionally, it should be noted that all corpus sections utilized the sentence's sentiment value as label except for the *corpus1* segment which took the user comment's sentiment as value. On a practical implementation note, *scikit-learn's* (Pedregosa et al. (2011)) *LabelEncoder* was used to encode the *negative, neutral, positive* and *undecided/not labeled* label spectrum. The *SENT* feature was encoded with the *tf-idf* vectorizer of *scikit-learn* (see chapter 2.6.3) and all other features with the *OneHotEncoder* (see chapter 2.6.2). For each corpus section, 20% of the data were used for testing and 80% for training purposes.

#### 4.3.4.1 Most Frequent POS

Before taking the different linguistic features and training a variety of algorithms (see chapter 4.3.4.2), different kinds of POS investigated in this work are demonstrated in order to get a better intuition of what kind of features were used as input for this experiment. As written in the previous chapter (see 4.3.4), the most important POS for voicing sentiment are nouns (N), adjectives (ADJ), adverbs (ADV) and verbs (V). In addition to those POS, negation (NEGATION) was also used as an additional feature, since it can flip the sentiment, e.g. from *positive* to *negative* (Liu, 2015, p. 116). The different tables below show the ten most frequent terms of each POS. The terms were automatically tagged with LuNa (Sirajzade and Schommer (2019)). Due to the large amount of tokens in the corpus,

*This applies for all corpus segments except for *corpus1*. In this case, the features used are the whole user comment and not a single sentence.

only the ten most frequent ones per POS are examined in more detail. Each of the tables below presents the number of occurrence as a percentage of the total number in the corpus. The highest percentages per POS are highlighted in bold print. As shown in chapter 3.6, the distribution of sentiment in the corpus is not balanced. Of the total amount of 116,516 sentences tagged with a *negative, neutral, positive* or *undecided* label, 13,307 were *negative*, 36,960 *neutral*, 7,503 *positive* and 58,746 *undecided*. In order to account for this imbalance, this percentage of the total number in the corpus was used allowing for easier comparison. Table 4.16, for instance, shows that 3.83% of the 116,516 sentences in the corpus include the adjective *einfach* 'easy'. Furthermore, it shows the reader that 6.94% of the *negative*, 1.92% of the *neutral*, 2.81% of the *positive* and 4.47% of the *undecided* sentences have the adjective *einfach* 'easy' as one of their tokens.

The first group of linguistic features that is important for the following ML experiments is negation particles. The most frequent ones, i.e. *net / nët / nöt* 'not', *keng / kéng / këng* 'none', *keen / kee* 'none', *näischt* 'nothing, none' and *ni* 'never', are shown in table 4.15. Those underline an interesting particularity of Luxembourgish mentioned in chapter 2.10, namely that there is a significant variation in spelling. Due to this variation, some of the most frequent negation particles appeared in different writing forms. They were added as one in this table. Not surprisingly, all negation particles mostly appeared in *negative* sentences. On the one hand, they were used as individual features for the ML experiments (see chapter 4.3.4.2). On the other hand, they were also used in combination with adjectives and verbs. I opted to work with negators both as individual features and together with other POS when using POS in the input, as the existence of POS in a context is capable of increasing the *negativity* or decreasing the *positivity* score of a word (see also (Kiritchenko and Mohammad, 2016, p. 47) for a discussion).

Table 4.15 – The most frequent negation particles and their sentence sentiment to be compared by the percentage of the total occurrence in the respective sentiment class.

| Negation Particle | All Sentences | Negative | Neutral | Positive | Undecided |
|---|---|---|---|---|---|
| net/ nët / nöt | 46.63 | **94.5** | 33.17 | 9.53 | 49.00 |
| keng / kéng / këng | 8.36 | **11.01** | 6.25 | 7.25 | 9.23 |
| keen / kee | 8.53 | **15.57** | 6.55 | 2.06 | 9.00 |
| näischt | 1.96 | **4.10** | 1.40 | 0.33 | 2.06 |
| ni | 0.58 | **1.32** | 0.49 | 0.03 | 0.54 |

Table 4.16 includes the ten most frequent adjectives, i.e. *einfach* 'easy', *besser* 'better', *richteg* 'right', *laang* 'long', *verschidden* 'different', *normal* 'normal', *kloer* 'clear', *egal* 'to be all the same', *gutt* 'good' and *wichteg* 'important'. *Einfach* 'easy', *laang* 'long', *egal* 'to be all the same' and *gutt* 'good' mostly appear in *negative* sentences. *Besser* 'better', *richteg* 'right' and *wichteg* 'important' mostly occur in *positive* contexts whereas *verschidden* 'different', *normal* 'normal' and *kloer* 'clear' show most frequently in *undecided* sentences. Interestingly, the majority of those adjectives appear in sentences that hold the same sentiment orientation. If annotated without any context by themselves, it is reasonable to expect *besser* 'better', *richteg* 'right' and *wichteg* 'important' to carry a

*positive* sentiment and *verschidden* 'different' and *normal* 'normal' to either be in the *undecided* or the *neutral* class. However, not all adjectives behave this way. Two of the most frequent adjectives in the corpus, *einfach* 'easy' and *gutt* 'good', are mostly in *negative* sentences whereas they would intuitively be expected to carry a *positive* sentiment. Albeit the importance of adjectives in carrying the *positive* or *negative* orientation of a text that was also noted by Taboada (Taboada, 2016, p. 328), taking this information to classify the whole sentence without context may not always be sufficient.

Table 4.16 – The 10 most frequent adjectives and their sentence sentiment to be compared by the percentage of the total occurrence in the respective sentiment class.

| Adjective | All Sentences | Negative | Neutral | Positive | Undecided |
|---|---|---|---|---|---|
| einfach | 3.83 | **6.94** | 1.92 | 2.81 | 4.47 |
| besser | 2.56 | 5.25 | 0.87 | **5.42** | 2.64 |
| richteg | 2.06 | 3.10 | 1.12 | **6.52** | 2.35 |
| laang | 1.79 | **3.11** | 1.16 | 1.04 | 1.99 |
| verschidden | 1.78 | 1.10 | 1.84 | 0.29 | **2.09** |
| normal | 1.75 | 1.75 | 1.44 | 0.76 | **2.06** |
| kloer | 1.73 | 1.28 | 1.69 | 1.24 | **1.92** |
| egal | 1.64 | **2.82** | 1.17 | 1.09 | 1.74 |
| gutt | 1.53 | **2.15** | 0.83 | 1.84 | 1.80 |
| wichteg | 1.34 | 2.21 | 0.38 | **3.49** | 1.48 |

In table 4.17, a second feature is added to the ten most frequent adjectives, i.e. the most frequent negation particles, *net/nët/nöt* 'not', *keng/kéng/këng* 'none', *keen/kee* 'none', *näischt* 'nothing, none' and *ni* 'never'. In contrast to the previously shown table (table 4.16), it shows the frequency of occurrence of the ten most frequent adjectives if one of those negation particle occurs in the same sentence instead of the absolute frequency of occurrence of adjectives in sentences in general. This illustration portrays that most adjectives, with the exception of *verschidden* 'different', appear mainly in *negative* sentences if at least one negation particle is included. This finding is not surprising, as research has shown that negation can reverse the sentiment of an instance (Taboada, 2016, p. 333). That way, adjectives such as *richteg* 'right' in *Dat ass richteg.* 'That is right.' would change from a *positive* to a *negative* sentiment if negated as in *Dat ass net richteg.* 'That is not right.'.

Table 4.17 – The 10 most frequent adjectives and negation particles and their sentence sentiment to be compared by the percentage of the total occurrence in the respective sentiment class.

| ADJ_NEGATION | All Sentences | Negative | Neutral | Positive | Undecided |
|---|---|---|---|---|---|
| einfach | 2.83 | **6.88** | 1.29 | 0.67 | 3.16 |
| besser | 1.60 | **5.52** | 0.42 | 0.97 | 1.53 |
| richteg | 1.46 | **3.34** | 0.83 | 0.24 | 1.58 |
| laang | 1.28 | **3.19** | 0.71 | 0.31 | 1.32 |
| verschidden | 0.96 | 1.01 | 0.82 | 0.05 | **1.15** |
| normal | 1.08 | **1.95** | 0.68 | 0.11 | 1.26 |
| kloer | 0.98 | **1.44** | 0.69 | 0.19 | 1.17 |
| egal | 1.09 | **2.93** | 0.60 | 0.27 | 1.09 |
| gutt | 0.99 | **2.23** | 0.41 | 0.19 | 1.17 |
| wichteg | 0.73 | **2.17** | 0.18 | 0.48 | 0.79 |

The third linguistic features which were used for the following ML experiments are adverbs. The ten most frequent ones in the corpus are portrayed in table 4.18. They are *och* 'also', *dann* 'then', *nach* 'more', *do* 'there', *méi* 'more', *esou* 'so', *hei* 'here', *awer* 'nevertheless', *wou* 'where' and *elo* 'now'. As the table shows, most of them most frequently appear in *negative* sentences when counting the percentage of their total occurrence in the different sentiment classes no matter if a negator exists in the sentence or not. In contrast to the above described features of adjectives (see tables 4.16 and 4.17), the existence of an adverb in a sentence alone does not lead to any conclusion on the connection between the usage of adverbs and the sentiment of the sentence. This finding was also explored in more detail in Dragut and Fellbaum (2014). The authors of this paper worked with intensifying adverbs and state that they do not carry sentiment themselves, but rather help to strengthen the sentiment of other words in a context. Additionally, Kiritchenko and Mohammad (Kiritchenko and Mohammad, 2016, pp. 49-50) note that the impact of the usage of certain adverbs on sentiment is not really high which underlines Dragut and Fellbaum (2014)'s finding of the role of adverbs for sentiment detection.

Table 4.18 – The 10 most frequent adverbs and their sentence sentiment to be compared by the percentage of the total occurrence in the respective sentiment class.

| Adverb | All Sentences | Negative | Neutral | Positive | Undecided |
|---|---|---|---|---|---|
| och | 33.72 | **44.93** | 23.54 | 27.46 | 38.38 |
| dann | 20.96 | **30.86** | 16.69 | 10.58 | 22.74 |
| nach | 18.61 | **27.80** | 15.52 | 10.85 | 19.47 |
| do | 14.70 | **21.50** | 11.20 | 8.82 | 16.10 |
| méi | 13.15 | **18.89** | 9.87 | 9.81 | 14.33 |
| esou | 13.02 | **19.16** | 7.84 | 9.57 | 15.33 |
| hei | 11.88 | **14.94** | 8.42 | 6.68 | 14.02 |
| awer | 11.73 | **17.11** | 8.46 | 8.13 | 13.03 |
| wou | 9.90 | **13.04** | 8.61 | 5.25 | 10.59 |
| elo | 8.30 | **10.44** | 7.71 | 5.13 | 8.60 |

The fourth linguistic feature group used for the analysis is verb forms. The ten most occurring ones are *geet* 'he/she/it goes', *maachen* 'to make, I/we/they make', *kommen* 'to come, I/we/they come', *soen* 'to say, I/we/they say', *kritt* 'he/she/it gets, you get', *goen* 'to go', *kréien* 'to get, I/we/they get', *fannen* 'to find, I/we/they find', *kennen* 'to know, I/we/they know' and *schaffen* 'to work, I/we/they work'. The percentage of occurrence in the whole corpus (all sentences) and in the *negative, neutral, positive* and *undecided* sentences are shown in table 4.19 and the percentage of those ten verbs under the condition that a negation particle occurs in the same sentence is portrayed in table 4.20. Interestingly, all verbs except for *kritt* 'get' always show mostly in *negative* sentence no matter if we examine the occurrence of verbs in general or of verbs with negation particles in the same sentence in specific.

Table 4.19 – The 10 most frequent verb forms and their sentence sentiment to be compared by the percentage of the total occurrence in the respective sentiment class.

| Verb | All Sentences | Negative | Neutral | Positive | Undecided |
|---|---|---|---|---|---|
| geet | 4.87 | **7.61** | 4.22 | 2.25 | 4.99 |
| maachen | 4.65 | **7.04** | 3.09 | 3.57 | 5.23 |
| kommen | 4.05 | **6.15** | 2.98 | 2.84 | 4.41 |
| soen | 3.64 | **6.10** | 2.02 | 2.21 | 4.28 |
| kritt | 3.49 | 3.40 | 3.41 | 1.73 | **3.64** |
| goen | 3.42 | **5.23** | 2.18 | 2.13 | 3.96 |
| kréien | 3.01 | **3.59** | 2.56 | 2.05 | 3.27 |
| fannen | 2.30 | **4.28** | 1.86 | 3.23 | 3.40 |
| kennen | 2.99 | **4.91** | 1.76 | 2.20 | 3.44 |
| schaffen | 2.89 | **4.29** | 2.25 | 1.35 | 3.17 |

Table 4.20 – The 10 most frequent verb forms and negation particles and their sentence sentiment to be compared by the percentage of the total occurrence in the respective sentiment class.

| Verb_NEGATION | All Sentences | Negative | Neutral | Positive | Undecided |
|---|---|---|---|---|---|
| geet | 3.46 | **7.80** | 2.38 | 0.51 | 3.53 |
| maachen | 2.91 | **6.55** | 1.59 | 0.84 | 3.18 |
| kommen | 2.65 | **6.05** | 1.59 | 0.63 | 2.81 |
| soen | 2.64 | **6.35** | 1.22 | 0.28 | 2.30 |
| kritt | 2.25 | **4.04** | 1.77 | 0.52 | 2.37 |
| goen | 2.39 | **5.56** | 1.24 | 0.67 | 2.60 |
| kréien | 1.90 | **3.60** | 1.34 | 0.55 | 2.04 |
| fannen | 1.99 | **4.31** | 1.09 | 0.55 | 2.21 |
| kennen | 2.11 | **5.10** | 1.00 | 0.61 | 2.32 |
| schaffen | 1.98 | **4.17** | 1.23 | 0.25 | 2.18 |

The fifth POS that is important for the following ML experiment is the feature group of nouns. Table 4.21 includes the ten most frequent nouns in percentage terms. The nouns portrayed there are *Leit* 'people', *Land* 'country', *Här* 'Mr.', *Staat* 'state', *Regierung* 'gov-

ernment', *Joer* 'year', *Euro* 'euro', *Auto* 'car', *Kanner* 'children', *Bierger* 'citizen, mountains' and *Zäit* 'time'. All nouns appear mostly in *neutral* or *undecided* contexts. The only exception is the noun *Leit* 'people' which is most frequently found in *negative* contexts.

Table 4.21 – The 10 most frequent nouns and their sentence sentiment to be compared by the percentage of the total occurrence in the respective sentiment class.

| Noun | All Sentences | Negative | Neutral | Positive | Undecided |
|---|---|---|---|---|---|
| Leit | 12.29 | **18.40** | 8.79 | 7.88 | 13.68 |
| Land | 11.18 | 5.88 | 9.98 | 3.03 | **14.18** |
| Här | 8.00 | 2.66 | **10.84** | 2.12 | 8.17 |
| Staat | 6.78 | 2.02 | **12.24** | 0.67 | 5.20 |
| Regierung | 5.05 | 2.01 | **7.64** | 0.75 | 4.66 |
| Joer | 5.03 | 4.85 | **6.53** | 2.64 | 4.43 |
| Euro | 4.33 | 0.57 | **9.67** | 0.24 | 2.34 |
| Auto | 4.30 | 2.07 | **5.66** | 0.56 | 4.43 |
| Kanner | 4.16 | 4.79 | 2.97 | 3.73 | **4.81** |
| Bierger | 3.31 | 1.77 | **3.97** | 0.65 | 3.59 |
| Zäit | 2.50 | 1.13 | **3.03** | 0.84 | 2.70 |

The overview carried out in this chapter shows the distribution of sentiment of different POS as a percentage of their total occurrence in the *negative, neutral, positive* or *undecided* classes and in the whole corpus of annotated sentences. It demonstrates that the existence of certain POS can have an impact on the overall sentiment of a sentence. In the examples discussed here, the impact is especially visible for adjectives (see tables 4.16 and 4.17). In the following chapter, the features in their full range were taken as input to train a variety of ML algorithms.

### 4.3.4.2 Results of the Linguistic Features Experiment

Table 4.22 shows the results of the experiment applying the linguistic features which were described in the previous two chapters. The highest $F_1$ score obtained from testing each corpus section is portrayed in bold print. As for all preceding ML experiments, training and testing was performed on all five corpus sections (see chapter 3.6.2 for a description) separately. The algorithms taken were DT, RF, SVM, KNN and LRM. For each corpus section, all training and testing were implemented with a range of features, namely the whole sentence (SENT), the adjectives (ADJ), the adverbs (ADV), the verbs (V), the nouns (N), the negators (NEGATION), the adjectives and negators (ADJ_NEGATION), the sentences and negators (SENT_NEGATION), the verbs and negators (V_NEGATION) and all of those features combined (all features). The labels for the corpus section *corpus1* were the sentiment annotations on user comment level. All the other corpus sections took the sentence annotations as training target. The possible labels were *negative, neutral, positive* and *undecided*.

Table 4.22 – $F_1$ score of all corpus sections without preprocessing and with different feature combinations on the Luxembourgish data.

| Corpus Section | Scenario | DT | RF | SVM | KNN | LRM |
|---|---|---|---|---|---|---|
| Corpus1 | SENT | 0.31 | 0.32 | 0.23 | 0.33 | 0.32 |
| | ADJ | 0.34 | 0.34 | 0.34 | 0.34 | 0.34 |
| | ADV | 0.34 | 0.34 | 0.34 | 0.32 | 0.34 |
| | V | 0.36 | 0.32 | 0.32 | 0.32 | 0.32 |
| | N | 0.23 | 0.23 | 0.23 | 0.23 | 0.23 |
| | NEGATION | 0.25 | 0.25 | 0.24 | 0.36 | 0.24 |
| | ADJ_NEGATION | 0.36 | 0.36 | 0.36 | 0.36 | 0.35 |
| | SENT_NEGATION | 0.24 | 0.23 | 0.23 | 0.36 | 0.24 |
| | V_NEGATION | 0.35 | 0.36 | 0.36 | 0.36 | 0.33 |
| | all features | 0.35 | 0.30 | 0.32 | 0.36 | **0.38** |
| Corpus2 | SENT | 0.36 | 0.39 | 0.20 | **0.41** | 0.40 |
| | ADJ | 0.37 | 0.37 | 0.37 | 0.27 | 0.37 |
| | ADV | 0.34 | 0.34 | 0.33 | 0.35 | 0.33 |
| | V | 0.28 | 0.27 | 0.27 | 0.29 | 0.27 |
| | N | 0.19 | 0.20 | 0.19 | 0.19 | 0.19 |
| | NEGATION | 0.19 | 0.20 | 0.20 | 0.22 | 0.20 |
| | ADJ_NEGATION | 0.37 | 0.38 | 0.38 | 0.38 | 0.38 |
| | SENT_NEGATION | 0.20 | 0.18 | 0.20 | 0.22 | 0.20 |
| | V_NEGATION | 0.31 | 0.30 | 0.30 | 0.33 | 0.27 |
| | all features | 0.33 | 0.31 | 0.30 | 0.37 | 0.36 |
| Corpus3 | SENT | 0.25 | 0.18 | 0.15 | 0.24 | 0.14 |
| | ADJ | 0.17 | 0.17 | 0.17 | 0.16 | 0.17 |
| | ADV | 0.22 | 0.23 | 0.23 | 0.24 | 0.20 |
| | V | 0.22 | 0.22 | 0.22 | 0.17 | 0.20 |
| | N | 0.16 | 0.16 | 0.16 | 0.16 | 0.16 |
| | NEGATION | 0.17 | 0.17 | 0.15 | 0.18 | 0.15 |
| | ADJ_NEGATION | 0.19 | 0.17 | 0.17 | 0.19 | 0.17 |
| | SENT_NEGATION | 0.15 | 0.15 | 0.15 | 0.18 | 0.15 |
| | V_NEGATION | 0.23 | 0.22 | 0.22 | 0.21 | 0.20 |
| | all features | **0.27** | 0.16 | 0.17 | **0.27** | 0.24 |
| Corpus4 | SENT | 0.43 | 0.37 | **0.59** | 0.40 | 0.58 |
| | ADJ | 0.31 | 0.31 | 0.31 | 0.23 | 0.29 |
| | ADV | 0.28 | 0.28 | 0.27 | 0.20 | 0.26 |
| | V | 0.29 | 0.29 | 0.29 | 0.17 | 0.26 |
| | N | 0.21 | 0.21 | 0.21 | 0.14 | 0.19 |
| | NEGATION | 0.18 | 0.18 | 0.18 | 0.23 | 0.18 |
| | ADJ_NEGATION | 0.33 | 0.33 | 0.32 | 0.27 | 0.30 |
| | SENT_NEGATION | 0.21 | 0.21 | 0.21 | 0.24 | 0.21 |
| | V_NEGATION | 0.29 | 0.29 | 0.29 | 0.26 | 0.27 |
| | all features | 0.33 | 0.29 | 0.34 | 0.30 | 0.33 |
| Corpus5 | SENT | 0.42 | 0.37 | **0.57** | 0.39 | 0.56 |
| | ADJ | 0.31 | 0.31 | 0.31 | 0.29 | 0.29 |
| | ADV | 0.28 | 0.28 | 0.28 | 0.27 | 0.26 |
| | V | 0.29 | 0.29 | 0.28 | 0.27 | 0.26 |
| | N | 0.21 | 0.21 | 0.21 | 0.19 | 0.19 |
| | NEGATION | 0.18 | 0.18 | 0.18 | 0.21 | 0.18 |
| | ADJ_NEGATION | 0.32 | 0.31 | 0.31 | 0.26 | 0.30 |
| | SENT_NEGATION | 0.21 | 0.21 | 0.21 | 0.23 | 0.19 |
| | V_NEGATION | 0.30 | 0.29 | 0.29 | 0.24 | 0.28 |
| | all features | 0.33 | 0.29 | 0.34 | 0.28 | 0.34 |

One first observation that can be made from studying the results in table 4.22 is that only two $F_1$ scores are above 0.50. Additionally, there is no obvious difference in the $F_1$ value when utilizing different linguistic features and algorithms on each corpus section. The best results are achieved either taking all features and performing *one-hot encoding* or using *tf-idf* to transform the input sentences (or comments in the case of the *corpus1* section). The relatively good performance of *tf-idf* was already demonstrated in chapter 4.3.3.1. Subsequently, this experimental setup suggests that simply using linguistic features to detect the overall sentiment of a sentence (or user comment) is not sufficient to obtain significant results. In the case of the experiment presented here, all linguistic features are the result of automatic POS tagging with LuNa (Sirajzade and Schommer (2019)). Therefore, possible mistakes happened during the tagging process that could potentially have an influence on the quality of the linguistic features. For instance, some of the most frequent tokens tagged as *N* (noun) were *d'* 'the', *sin* 'to be, I am' and *gin* 'to give, I give' although they are not nouns but rather a definite article and two verbs respectively. Therefore, they were replaced with the next most frequent POS that were actually nouns (see table 4.21). Additionally, no parameter tuning during the training was done meaning that the algorithms used might not have performed at their best. Sirajzade et al. (2020b) followed a different approach in their component analysis of adjectives for sentiment detection on Luxembourgish data. In contrast to the experiment presented here, they worked solely with hand-labeled sentences and adjectives. As the annotation was carried out by a trained linguist (the author of this thesis), it is expected that the annotations are comparably more consistent and reliable. Overall, the $F_1$ scores observed by Sirajzade et al. (2020b) when applying different algorithms were better than the scores presented in this chapter. Consequently, the quality of the POS used as input is likely to have a crucial impact on the performance of the sentiment detection, and simply automatically annotating as much data as possible might not be sufficient. This observation could be further examined in future work by hand-labeling a larger amount of data and then using the labeled data as training samples for the algorithms presented in this chapter.

To sum up, working with Luxembourgish data annotated for sentiment performs best on the biggest corpus sections (*corpus4* and *corpus5*), as training on those generally leads to the highest $F_1$ scores. This shows that the quantity of input can have an impact on the performance of an ML system. Unexpectedly, using linguistic features as additional input to different algorithms did not improve the performance. Providing linguistic information can be henceforth not as important as taking a large quantity of examples when learning to detect sentiment patterns in texts. In the following chapter, the corpus sections used for carrying out the different automatic sentiment detection processes will be closely examined from a different angle by going beyond simple $F_1$ scores.

## 4.4 Going beyond $F_1$ Score

So far, different ML experiments working with the different corpus sections of the RTL sentiment corpus in their English and German translations as well as their Luxembourgish original were presented and discussed. Each of those corpus sections were taken both in a preprocessed and unpreprocessed form to study the influence of preprocessing on the training success. For all of those different experiments, one thing remained the same: Evaluation of the performance was always accomplished by simply studying and comparing the $F_1$ score of the validation or test set of the corresponding corpus segment. Evaluating a system in this way certainly gives valuable insights into the capacity of performance. Nevertheless, it is not always sufficient to truly understand why algorithms learn to make certain decisions. In order to better infer conclusions going beyond the $F_1$ score, two analyses were carried out which will be discussed below. The first one examines the raw frequency of words in each class and the second one a ranked distribution of words in the *negative* and *positive* classes.

### 4.4.1 Raw Frequency of Words in each Sentiment Class

This chapter is dedicated to studying the raw frequency of words in each sentiment class, i.e. the occurrence of the most common terms in the *negative, neutral, positive* and *undecided* classes. Figure 4.1 shows the ten most frequent words per class in the original Luxembourgish corpus. The corpus was lower-cased and the most frequent non-sentiment bearing words were removed beforehand to facilitate interpretation. The corpus was not normalized for this analysis which is why spelling mistakes are still part of the data. They were *dat* 'the, that', *dass* 'that', *an* 'and, in', *wat* 'what', *fir* 'for, to', *d* (sic!) 'the', *well* 'because, already, wave', *di* 'the', *déi* 'that, the', *mee* 'but (however), May', *vill* 'a lot', *méi* 'more (than), effort, mow (imperative)', *elo* 'now', *ech* 'I', *wei* (sic!) 'when', *gett* (sic!) 'give' and *ëmmer* 'always'. At first glance, most words do not seem to carry any sentiment and there does not seem to be a big difference in the words used in different classes. The ten most frequent words for the *neutral* class are *staat* 'state', *euro* 'euro', *här* 'Mr.', *leit* 'people, he/she/it suffers, you suffer', *land* 'country', *ass* 'he/she/it is', *regierung* 'government', *joer* 'year', *sinn* 'to be', *ginn* 'to give, I/we/they give'. The ten most frequent *negative* words are *leit* 'people, he/she/it suffers, you suffer', *ass* 'be', *si* 'she, to be', *sinn* 'to be, I am, we/they are', *ginn* 'to give, I/we/they give', *mol* 'times', *gi* 'give', *keng* 'no, none', *et* 'it' and *ganz* 'very, whole'. For the *positive* class, the ten most frequent words are *besser* 'better', *ganz* 'very, whole', *leit* 'people, he/she/it suffers, you suffer', *sinn* 'be', *si* 'she, be', *nët* (sic!) 'not', *ass* 'be', *respekt* 'respect', *gudde* 'good', *ginn* 'to give, I/we/they give'. The most frequent words in the sentences that were labeled with an *undecided* sentiment are *leit* 'people, he/she/it suffers, you suffer', *land* 'country', *ass* 'be', *ganz* 'very, whole', *sinn* 'be', *ginn* 'to give, I/we/they give', *här* 'Mr.', *mol* 'times' and *gi* 'give'.

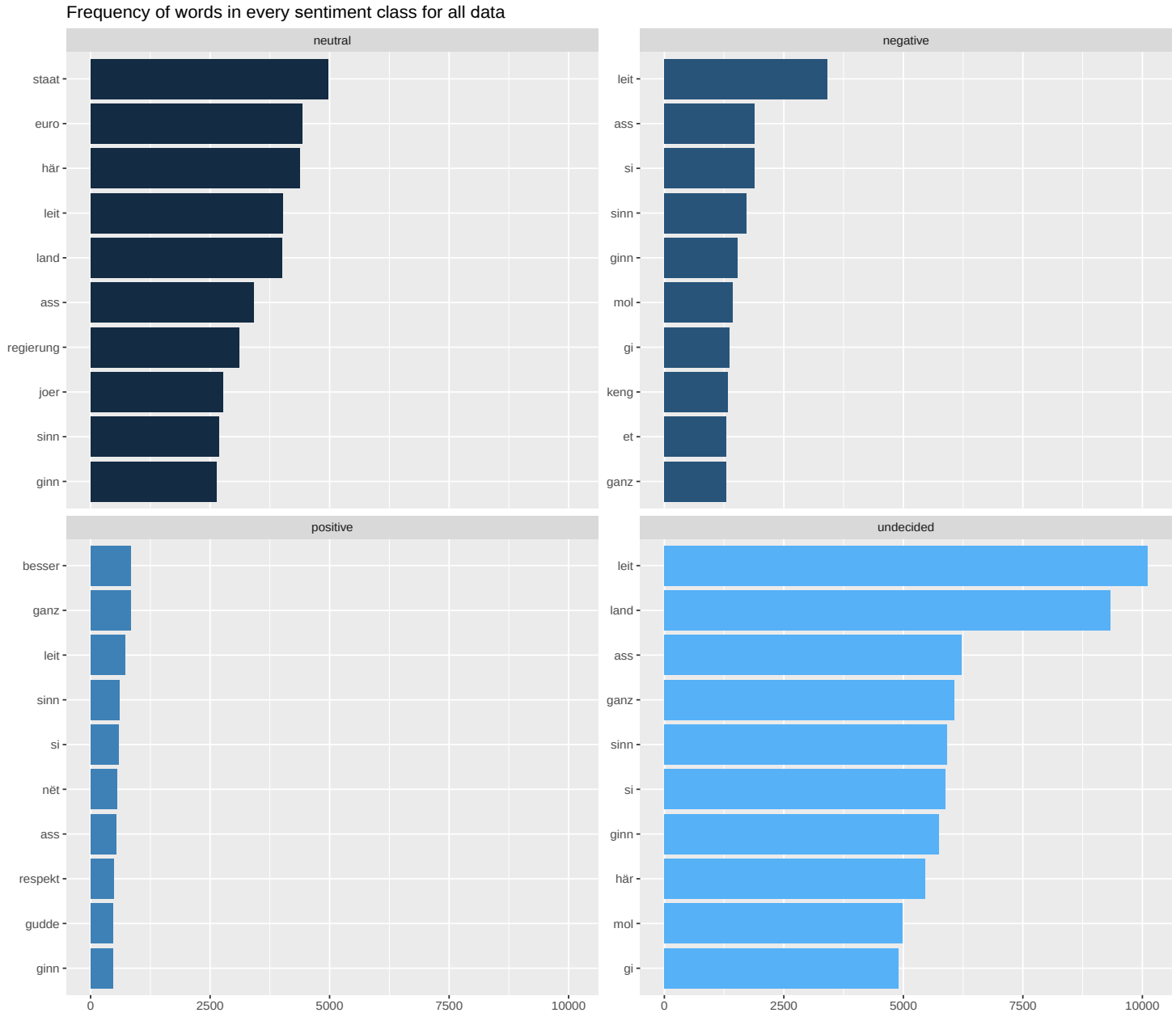



Figure 4.1 – The 10 most frequent words in every sentiment class in the whole corpus in Luxembourgish.

Most of those items do not carry any sentiment that might truly identify them as belonging to either the *negative, neutral, positive* or *undecided* class. The *negative* class contains one indefinite pronoun (*keng* 'no, none') that can be perceived as *negative* and the *positive* class the adjectives *besser* 'better' and *gudde* 'good' as well as the noun *respekt* 'respect' that could be seen as incontestably *positive*. At the same time, four of the ten most frequent words per sentiment class are identical for all classes, i.e. *leit* 'people, he/she/it suffers, you suffer', *ass* 'be', *sinn* 'to be, I am, we/they are', *ginn* 'to give, I/you/they give', and

the *positive* class includes the particle *nët* (sic!) 'not' as one of the most frequent terms although it could intuitively also be classified as *negative*. This overview of the most frequent words in each class suggests that simply looking at the most frequent words in a class, even after having removed some stopwords and non-sentiment bearing words, is not sufficient to determine the sentiment and does not give a good intuition as to how sentiment is expressed in all four classes. On the contrary, the most common words are rather similar. This implies that teaching a classifier to automatically distinguish between all those classes containing similar words might not be an easy task and that additional information beyond the word frequency level is required in order to successfully classify unseen instances into the *negative, neutral, positive* or *undecided* class options.

### 4.4.2 Contrasting Negative with Positive Sentiment Going beyond Word Frequencies Using Scattertext Plot

As shown in the previous section, removing stopwords and subsequently examining the most frequent words per sentiment class is not necessarily sufficient to adequately detect *negative, neutral, positive* or *undecided* in language. In this chapter, I present a different way of studying the distribution of words in sentiment classes or different corpora by creating a scattertext plot, as proposed by Kessler (2017). The idea behind this is to not use raw frequency of words to create a plot, as was shown in figure 4.1 in the previous section, but to rank the terms by frequency percentile. Kessler (2017) furthermore proposes the measure of scaled f-score, i.e. the harmonic mean of the precision and the recall measure of a word. The underlying intuition of this approach is that each term associated with a category has a high precision and term frequency in the category it occurs in. Kessler (2017)'s visualization package was used to create a scattertext plot for all preprocessed sentences in the RTL corpus either labeled with a *negative* or *positive* label. The sentences were preprocessed normalizing the spelling and removing stopwords with spellux (Purschke (2020b)). Furthermore, all sentences were lowercased before fed into the scattertext tool. Only the *negative* and *positive* sentences were used for this visualization and not the *undecided* and *neutral* ones. The main reason for this choice was that, as written in chapter 2.1, sentiment is perceived as being on a scale ranging from *negative* through *neutral* towards *positive* (see figure 2.1). According to this scale, *negative* and *positive* are on opposing ends suggesting that they might have the biggest difference of sentiment-bearing particles that can be found in the data.

Figure 4.2 shows the distribution of sentiment bearing words in the *positive* and the *negative* class of the corpus data. Blue is associated with the *positive* and red with the *negative* class. The most negative words are placed towards the right on the x-axis and the most positive words towards the top of the y-axis. On the right side of the graph, a list of the top *positive*, the top *negative* and the most characteristic words of all texts is depicted. Scattertext identified *bravo* 'bravo', *stolz* 'proud', *courage* 'courage', *zë* (sic!) 'too (too much)', *respekt* 'respect', *kéng* 'bold; likely to be the wrong spelling of the pronoun *keng* 'no, not a' ', *super* 'super', *dën* (sic!) 'the', *ech hoffen* 'I hope', *merci* 'thanks', *dër* (sic!) 'you, yourself, the', *all respekt* 'all respect', *éng* (sic!) 'a(n), one', *hoffen dass* 'hope that' as

the most *positive* words. *Keen* 'no, none', *manner* 'less', *nie* (sic!) 'never', *näischt* 'nothing', *domm* 'stupid', *nett* (sic!) 'not', *guer* 'guer net: not at all', *naischt* (sic!) 'nothing', *schlecht* 'bad', *neischt* (sic!) 'nothing', *dier* 'dead, dry, door', *net* 'not', *deier* 'expensive', *gett* (sic!) 'give' are the most *negative* items. Interestingly, the top *negative* and *positive* are by far more intuitively sentiment-bearing words than when I looked at the raw frequency of words in figure 4.1.

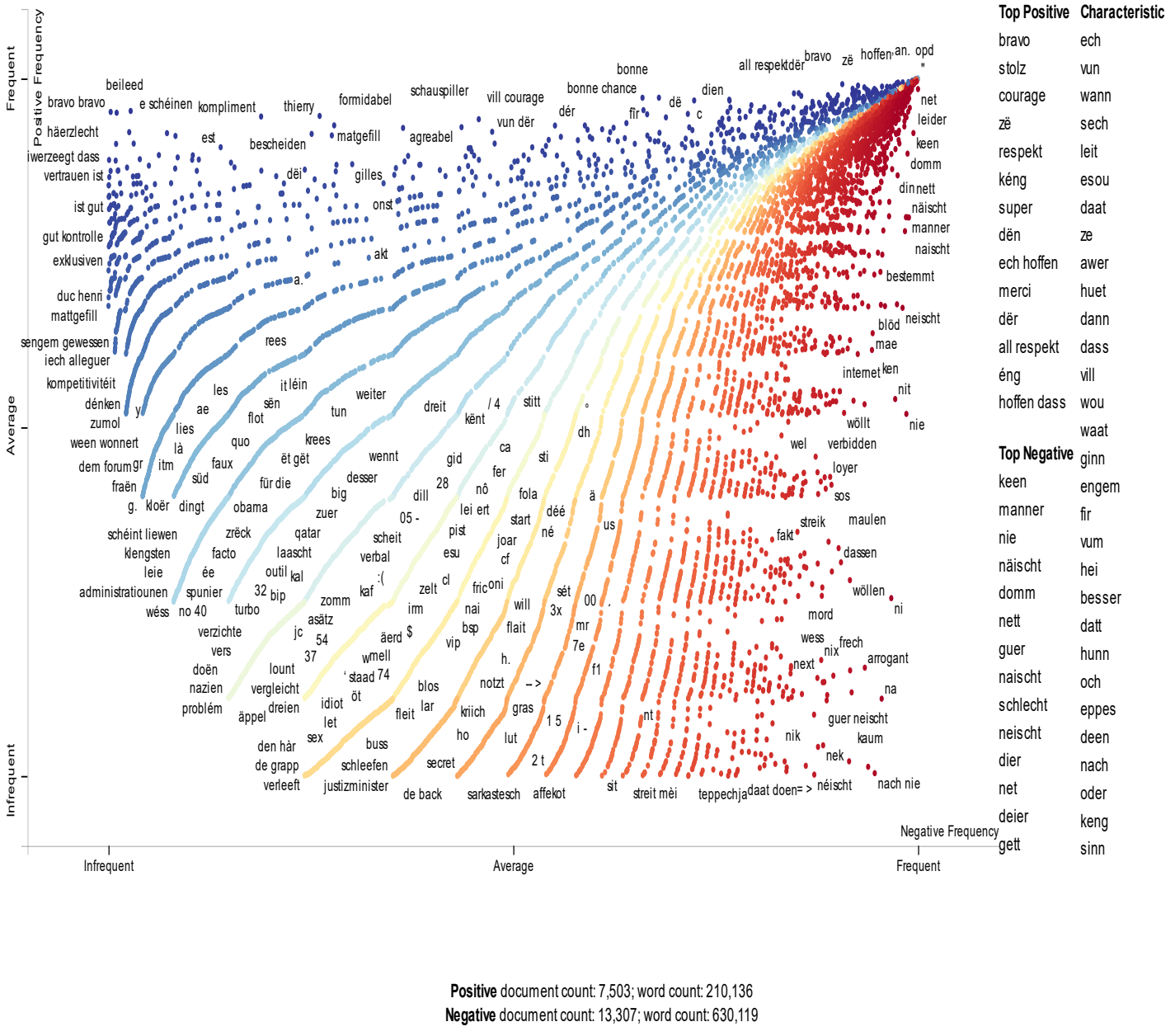



Figure 4.2 – A scattertext plot of the whole annotated corpus contrasting the *negative* and the *positive* class.

The comparison between the scattertext plot using a ranking technique (see figure 4.2) and the count of raw frequency distribution in the data (see figure 4.1) clearly show that it is possible to find differences in the different classes and to distinguish words that are rather *positive* or *negative* such as *bravo* 'bravo', *super* 'super', *stolz* 'proud' and *respekt* 'respect' for the *positive* and *domm* 'stupid' and *schlecht* 'bad' for the *negative* top words. However, not all of those can be identified by simply counting the frequency of words like in figure 4.1. This finding suggests that the raw frequency of words is not sufficient to learn about sentiment distribution in the RTL corpus. Additionally, it should be noted that those words shown in the graph are not the most common ones. This means that a classifier is likely to get more sentences with frequent words that often do not carry any sentiment than with these ones. Therefore, only a small number of sentences could be classified correctly if only the word-level and no additional other information would be considered for automatic sentiment detection.

Thinking again about all the experiments described in the preceding chapters using different corpus sections in different languages emphasizes this discovery. For instance, using translations to a bigger language such as English and taking word embeddings obtained from a large corpus (see chapter 4.1.1) did not automatically lead to promising results. This large corpus naturally included many occurrences of frequent words as it tries to depict the language in question as much as possible, but is not specialized for sentiment detection. For this experiment, none of the $F_1$ scores obtained exceeded 0.50 (see tables 4.3 and 4.4). Considering how well sentiment-bearing words such as *bravo* 'bravo', *super* 'super', *stolz* 'proud' (*positive*) and *domm* 'stupid' and *schlecht* 'bad' (*negative*) were discovered in the corpus using Kessler (2017)'s approach, future work dealing with a heterogeneous corpus such as the one used for this thesis could be to implement a hybrid approach that does not solely rely on vectors retrieved from general language settings such as word embeddings and labeled sentences/comments but also includes either a ranking information similar to what was proposed in Kessler (2017) and/or a sentiment dictionary to give more *negative, neutral, positive* or *undecided* weight to words found in specific classes.

## 4.5 Lessons Learned from Detecting Sentiment with ML Techniques

This part of the thesis described the multilayered approach of working with a variety of ML algorithms and a corpus in three different languages. The idea behind this complex approach was to show which kind of method works best for detecting sentiment in a heterogeneous corpus of Luxembourgish user comments. Several observations can be made from the experiments: First, automatically translating a corpus into a language with more resources such as English or German can be helpful despite the drawback of including new noise due to translation mistakes. As was shown for the ten sentences extracted from the RTL corpus (see tables 4.2 and 4.7), most words were translated correctly, but the semantic roles of the different POS were not always adequately identified.

The tendency towards the translations working sufficiently well can be seen from the experiments with German and English *BERT* models which result in satisfactory performance, especially for the largest corpus sections *corpus4* and *corpus5*. This finding, however, cannot be generalized to all cases, as the use of word embeddings (word2vec) worked better on the Luxembourgish sentences than on the German and English ones although the pre-trained word embeddings for both English and German were bigger than the Luxembourgish ones. Second, performing intensive preprocessing on the corpus does not always help to improve the training results. This can for instance be seen in the *tf-idf* experiment on Luxembourgish data (see table 4.14) for which the unpreprocessed versions performed significantly better than the preprocessed ones. Third, including linguistic features as additional input for training does not yield promising results notwithstanding the intuitive assumption that providing additional information about sentiment-bearing POS such as adjectives (like *traureg* 'sad' and *frou* 'happy') could be useful for automatically detecting sentiment in texts. Some other research on using linguistic features for sentiment analysis in other languages, such as Korean (Jang and Shin (2010)), however, showed promising results. Therefore, further work could be to improve the quality of the linguistic features leveraged as input, for instance by working with human expert annotators instead of automatically produced POS. Additionally, the class balance in further work should be improved. That way, potential biases in the training data can be reduced. Last but not least, an interesting path would be to take *spellux* (Purschke (2020b)) only for spelling correction and not for any other preprocessing task. By doing that, it could be examined whether removing parts of the large spelling variation in Luxembourgish can help to improve the results.

To sum up, I have herein applied several approaches to detect sentiment in user comments. While some indications point towards fine-tuning *BERT* (Devlin et al. (2019)) being a powerful and promising technique, no final recommendation as to which model works overall the best for the data can be given. The first step of further research should therefore be to improve the data quality, as having such large imbalance in the data set as well as annotations from several different sources make it challenging to proceed due to the inconsistency of annotations from different sources. Due to the nature of this work as

being the first one of its kind for the Luxembourgish language, shortcomings remain as discussed here. On the annotation level, it should be underlined again that several different approaches were applied to collect annotations ranging from labeling by one annotator to crowdsourced annotations to automatically labeled instances of sentiment (see chapters 3.3, 3.5 and 3.6). As the calculations of Fleiss' kappa and Krippendorff's $\alpha$ have shown (see chapters 3.4 and 3.4.2), the IAA for the crowdsourced sentiment labels were rather low suggesting that the quality of the annotations collected should be improved. A possible solution for this would be to improve the guidelines for annotation. One way of doing this could be to provide examples of how *negative, neutral* or *positive* is expressed in language or to train annotators before they start with the actual annotation task. Additionally, it should be noted that a large amount of the annotations taken for training in the different ML experiments were automatically collected, either by means of a lexicon-based approach or through the help of an ANN (see section 3.6.2.3). While this results in a fast acquisition of an adequate amount of sentences labeled with a *negative, neutral* or *positive* sentiment, it is very hard to carry out certain quality control on those automatic annotations. It is thus hard to objectively judge whether or not those automatic annotations are of good quality. Regarding the annotated corpus, it should also be noted that the setup of the data set was in a highly unbalanced manner (chapter 3.6). This can lead to a big difference in how well different classes were learned during training of the different ML algorithms. Ideally, a sentiment system should be able to detect each sentiment class equally well and not be bias towards a certain one. Additionally, this work applied translations of annotated Luxembourgish sentences to English and German. As they were automatic translations (lushan88a (2020)), noise was included and none of the translations were optimal (see tables 4.2 and 4.7 for ten randomly extracted example sentences). Also, the sentiment labels were translated from Luxembourgish to the two target languages as well although sentiment might be culture-dependent. This means that sentiment might be expressed differently in German, English and Luxembourgish. Those points have to be taken into account if the automatically translated corpus should be used for future research. On a more theoretical note, it was already discussed in chapter 2.1 that defining sentiment as being on a scale from *negative* through *neutral* to *positive* is a major simplification of the concept of sentiment and is insufficient to represent the term in its full spectrum. More research would be required to move towards a more realistic approach of sentiment detection without simplifying the problem overly.

## 4.6 Recommendations for Sentiment Analysis in Low-resource Settings

In chapter 4, several approaches to detecting sentiment were tested and limitations as well as lessons learned from those experiments were discussed. This leads to a couple of crucial points that should be further pointed out here in order to give some recommendations as to how sentiment analysis for a low-resource language could be conducted.

First, as already mentioned above, detecting sentiment with the help of the *BERT* model (Devlin et al. (2019)) has shown to be the most promising approach compared to building sentiment analysis systems using a *word2vec* or *tf-idf* approach. While fine-tuning *BERT* overall performs well, it does not always achieve better results than other methods such as *tf-idf*. This is surprising, as *BERT* incorporates knowledge about the language itself whereas *tf-idf* only counts the frequency of terms in the given texts used for training an algorithm. The experiments and their results discussed in the sections above have rather shown that the corpus size can be more important than which computational approach is used. This is supported by the observation that working with corpus sections *corpus4* and *corpus5*, which were the biggest corpus sections, overall lead to better performance than when *corpus1, corpus2* or *corpus3* were taken as input data. What could be the reason for the fact that only corpus size, but neither the vectorization approach nor the algorithms used for training had a noticeable impact as to which approach yields the best results? I believe that the most likely explanation for this is the lack of a corpus with sentiment annotations of high quality and consistency. Additionally, it should be noted that the RTL corpus covers a large variety of topics. Not having taken this sentiment distributions into account when collecting the annotations could also have an impact on the model performance. As was shown in chapter 3.4.2 with the calculation of the IAA on parts of the RTL corpus, it can be difficult for human annotators to agree upon a *negative, neutral* or *positive* label for a piece of text. Furthermore, a large part of the sentiment annotations that were taken as training and testing data for this thesis were labeled by automatic annotation means. This results in a large amount of annotations generated in a short period of time. Nevertheless, a major disadvantage of employing this approach is that it is difficult to assure the consistency and quality of those labels without human interference.

The discussion above leads to the second crucial question: Is it really worth building special corpora for new languages? Or does it work equally well to simply utilize textual resources of bigger languages such as English and transfer those findings to smaller settings? The approach of this thesis was three-fold and answers those questions to some extend. The RTL corpus was automatically translated to English and German and the same ML experiments were carried out three times, once using the English, once the German and once the Luxembourgish version of the same corpus. Studying the macro-averaged $F_1$ score (see chapter 4.1.1.1) for the different experiments shows that the performance remained similar, no matter which version of the corpus was used. From a linguistic and also cultural point of view, creating independent corpora with sentiment labels for a language has

its merit and accounts for the cultural and linguistic particularities in sentiment expressions. Recent examples of work on building such resources for lower-resource languages are Regatte et al. (2020) and Ali et al. (2021). Creating tailor-made sentiment resources for a specific language comes with one important drawback: It consumes a lot of time and effort. Ideally, human experts have to be recruited and closely supervised to create a data set of high quality that can be used as training data for an automatic sentiment detection system. As I have shown in this thesis, those efforts do not always automatically result in well-performing sentiment detection systems and working with automatic translations can work equally or comparably well as input for ML algorithms. This observation was also made by Barriere and Balahur (2020) and Balahur and Turchi (2012). Therefore, the choice to manually create high quality new sentiment annotations should always be made carefully, as it does not always lead to better results. For this reason, automatic translations of good quality sentiment annotations of a resource-rich source language such as English to a resource-lower target language such as Luxembourgish should be considered a prioritized alternative for low-resource language tasks, especially when time or budget restrictions apply and creating a large corpus with high quality annotations is not possible.

For the purpose of data collection and preparation for analysis, a variety of different annotation approaches were used. While corpus sections *corpus1* and *corpus2* were only labeled by one annotator and subsequently only had either a *negative, neutral* or *positive* label per sentence or user comment, all other corpus sections had several initial labels from different annotators that were combined to one overall label set. More precisely, *corpus3* contains the sentences of the RTL corpus that were annotated by three annotators. *Corpus4* consists of all sentences labeled with the lexicon-based approach and a labeling technique using AAA. Before using them as training data for the different classifiers previously described in this chapter, all sentences having more than one label went through a merging technique in order to have one label only per sentence (*negative, neutral, positive* or *undecided*). For this purpose, the rule of the majority label applied. For instance, if a sentence had two *positive* and one *neutral* label, it was attributed the overall label of *positive*. If the original annotations of a sentence were *positive*, *neutral* and *negative*, an *undecided* label would be given. As the calculation of the IAA on the human-labeled crowdsourced sentences (see chapter 3.4.2) showed, putting sentences into sentiment categories is a difficult task and disagreement occurred more frequently than agreement. In chapter 2.1, I already mentioned this by stating that the open definition of sentiment taken by this thesis relies on the personal intuition of the annotator to a great extend. Depending on the piece of text an annotator is asked to label, choosing *negative, neutral* or *positive* can be difficult and disagreement among different people reading the same piece of text can occur. Having an *undecided* category as additional label could therefore be a good option for the simplistic sentiment annotation approach as used in this thesis. That way, difficult cases could be grouped together under one label which could possibly lead to the other three labels (*positive*, *negative* and *neutral*) being more consistent. A similar approach was also proposed by Kenyon-Dean et al. (2018). However, introducing this fourth category of *undecided* to the annotation process is still no guarantee to high quality and consistent annotations, as putting text into either four (*negative, neutral, positive* and *undecided*) or three (*negative, neutral* and *positive*) categories always remains an approximation meaning that disagree-

ment among annotators needs to be settled to create one agreeing label. In any case, it is recommended to use a fourth category of *undecided* in an annotation setup. Nevertheless, when planning the annotation, researchers should be aware that this is not a perfect solution. Generally speaking, the perfect solution that is working for every sentiment analysis task does not exist (yet). The reason for this is that language is very versatile making it highly challenging to find the right way of labeling it. For instance, a sentence such as *Ech ginn elo bei de Bäcker.* 'I am going to the bakery now.' does not carry any kind of sentiment at first glance. If the author of this sentence is highly stressed and in a hurry at the moment of making this statement, it could also be interpreted in a *negative* way by implying thoughts such as 'Oh God! I totally forgot that grandma is coming over for dinner today! I need to get to the bakery before they close!'. For the reasons alluded to here, it is important for researchers to be aware of the challenges in annotation planning. The costs and benefits should be weighed to find the most suitable solution for the specific research project.

To conclude, doing sentiment analysis or any other kind of classification task with a low-resource language always remains a challenge. One crucial factor influencing the planning is often the time and budget constraints. The experimental findings shown in this chapter indicate that working with automatic translations and large pre-trained models such as *BERT* (Devlin et al. (2019)) are a promising starting point for research. Nevertheless, this can only be the first part of a research project. In order to account for the cultural and linguistic peculiarities of a language, a full sentiment analysis system should work with a high quality annotated corpus of the language in question and not simply with translations using a bigger corpus. The planning and carrying out of the annotation of such a corpus certainly requires a lot of time and effort, but will be beneficial for advancing NLP research with the low-resource language (Luxembourgish in this case) in its whole.

Chapter 5

# Conclusion

This thesis has explored different ways to deal with sentiment detection for the low-resource language Luxembourgish. In this chapter, the major findings of this work will be summarized as well as the contribution to the field of research of sentiment analysis for low-resource languages. Additionally, the importance of the work for NLP research for Luxembourgish will be discussed. Additionally, a potential starting point for future work to improve the proposed sentiment analysis approach will be introduced.

## 5.1 Major Findings and Research Contribution

To the best of my knowledge, this work is the first to work on sentiment detection for the low-resource language Luxembourgish making it subsequently a foundational work in this field. As the first one to explore the field, there are many challenges. In the course of this work, I aimed at answering the following research questions:

- *How can sentiment analysis be carried out in a low-resource setting? What are the challenges related to this task and how can they (partially) be overcome?*

The textual basis for this work was an RTL corpus of 585,358 user comments and 179,283 news articles written between 1999 and 2018. The corpus was provided by RTL Luxembourg, the partner of the *STRIPS* project (see chapter 3.1) in the context of which this thesis was written. The RTL corpus was a useful textual basis for implementing a sentiment analysis project, since it covers a wide range of topics and the user comments of the articles on the RTL website span over a long period of time. For a human reader, this corpus is full of expressions of sentiments. In order to use it as input for building an automatic sentiment detection system, several preparatory steps had to be undertaken. First, preprocessing including spelling normalization was carried out (see chapter 3.1). Spelling correction is an important part of preprocessing Luxembourgish texts, as the corpus is full of spelling variation since Luxembourg's school system largely neglects teaching Luxembourgish spelling (see chapter 2.10). Second, an annotation tool was created (see chapter 3.2) to be able to collect crowdsourced sentence-level annotations from more than 26 different Luxembourgish speakers (see chapter 3.3.2). As stated in Sirajzade et al. (2020a),

this annotation tool was linked to the Temporal Warehouse (Gierschek et al. (2019)) of the *STRIPS* project, the data backbone chosen "to separate the data itself from the various applications that are linked to it" (Sirajzade et al., 2020a, p. 173). The annotation tool was designed as a web interface with a direct access to the eXist-db database (Siegel and Retter (2014)). A lot of different annotation tools exist (e.g. Daudert et al. (2019), Eryiğit et al. (2013), Yimam et al. (2013)). However, a sentiment annotation tool that is specifically created for Luxembourgish has not been implemented before. Creating a tool specifically for the purpose of collecting annotations for the *STRIPS* project came with a great advantage, as it allowed for a tailor-made solution for annotating and storing the annotations directly in the required *XML* format in the corresponding eXist-db database. Third and last was the annotation of the corpus with *negative, neutral* and *positive* labels (see chapter 3.3). This was carried out in a multilayered process: One part of the corpus was labeled on sentence-, user comment- and adjective-level by the author of this thesis, one part on sentence-level by means of crowdsourcing and a third part on sentence-level working with a lexicon-based and ANN-based approach for automatically labeling with *negative, neutral* and *positive* tags. As part of the lexicon-based approach, I compiled a sentiment lexicon for the Luxembourgish language (see chapter 3.5.2) based on the German PolArt (Klenner et al. (2009)) sentiment dictionary. This new resource follows the underlying intuition that each word has a semantic orientation (Taboada et al., 2011, p. 267) and can therefore contribute to the overall sentiment of a larger semantic unit.

After having completed the initial step of creating resources and preparing the data for detecting sentiment in Luxembourgish user comments, I used a variety of ML algorithms to learn sentiment structures in Luxembourgish (see chapter 4). For this purpose, the annotated sentences were automatically translated from Luxembourgish to English and German resulting in three different versions of the same data set which could be used for comparative studies in the future. While the semantic roles were mostly incorrectly translated (see tables 4.2 and 4.7), the words themselves are correctly reworded. The tenth sentence (see table 4.2), *Kengem Betrib hei am Land schued TVA Erhéijung direkt.* 'No business in this country is harmed directly by the increase of the VAT.' (RTL corpus, 12/2014, sentence ID 13856cu), for instance, was translated with 'No business in this country is directly harming VAT increase.'. In this example, the object *kengem Betrib* 'no business' was translated as the subject and the actual subject *TVA Erhéijung* 'VAT increase' was registered as the object of the sentence. As the words are still correctly paraphrased despite the difficulty of adequately identifying the semantic roles, those translations are mostly suitable as input for bigger models that can compensate such mistakes with a large amount of input data.

The analyses carried out and described in chapter 4 are the first of their kind executed for Luxembourgish sentiment detection demonstrating a large variety of methods to show how training and testing can be done for sentiment detection in Luxembourgish user comments. Several vectorization techniques (*tf-idf*, *one-hot encoding*, *word embeddings*) were used to convert the text to input data for a range of algorithms. Additionally, *BERT*, a large pre-trained language model (Devlin et al. (2019)), was fine-tuned for the sentiment detection task. In chapter 4.5, I discussed the experiments carried out in detail and showed that fine-tuning *BERT* (Devlin et al. (2019)) tends to be the most promising approach. However,

most results are similar meaning that no clear and final proposition as to which algorithm performs the best could be given.

As shown here, carrying out sentiment analysis for a low-resource language such as Luxembourgish is a challenging task. When there is no existing resource for the language, a researcher needs to carefully plan the process of creating a new resource. The costs and benefits of each decision involved in the process must be evaluated. In this thesis, a multilayered approach of annotation combining crowdsourcing, labeling by a linguist and automatic labeling was chosen. While this approach produces a large amount of annotations, adequately evaluating the quality of the annotation is difficult. I tried to partially check the quality of the annotations by calculating the IAA (see chapter 3.4.2) for the part of the corpus which was labeled by crowdsourcing. This calculation showed low agreement scores suggesting that the quality of the annotations might not be satisfactory. Additionally, the ML experiments described in chapter 4 frequently lead to very similar results questioning whether the input data taken for training and testing was sufficiently well-labeled. It also demonstrates that there are no clear differences between applying the algorithms with an automatically translated version of the data to German or English or with the original Luxembourgish data. The results obtained in this thesis can therefore be seen as the foundation work for moving towards a well-functioning sentiment detection system for Luxembourgish. More research, especially on improving the quality of the input data, is required in order to build a system which could reliably predict the sentiment in most cases. This, however, was not feasible in the time constraints accompanying this research project underlining again that building a well-functioning classifier which can adequately determine whether a piece of text is *negative, neutral* or *positive* is not an easy task. This leads us to the second research question of this thesis, the question of how the performance of such classification systems can be evaluated.

*- What are the possibilities of evaluating the performance of different classification systems on the annotated corpus of this work? Are those evaluation measures sufficient to adequately determine the quality of the performance?*

The RTL corpus was annotated with *negative, neutral* and *positive* sentiment labels using different annotation techniques. Furthermore, some sentences of the corpus were assigned an *undecided* label. This was the case for the sentences that were labeled three times and averaging the three annotations did not lead to agreement. In the next step, the RTL corpus was taken as input data for a selection of ML algorithms using different vectorization techniques (see chapters 2.6.2, 2.6.3, 2.6.4 and 2.6.5) and one pre-trained language model (see chapter 2.7.6). In order to evaluate how well a system learned to detect sentiment structures in texts, one of the standard evaluation measures in ML, $F_1$ score, was used (see chapter 2.8). Although the $F_1$ score is useful for the evaluation, evaluating the RTL corpus comes with one important challenge - the quality of the sentiment annotations is doubtful. The reason for the doubt is that a large part of the corpus was annotated making use of automatic means (cf. *corpus4* and *corpus5* sections, chapter 3.6). Verifying whether the annotations are of sufficient quality or if they include a lot of noise is difficult, as the same sentences were not additionally annotated by human annotators that could be treated as gold standard for evaluation purposes. However, calculating the IAA for the sentences

that were indeed annotated by human showed low agreement (see chapter 3.4.2) indicating that labeling the RTL corpus with *negative, neutral* and *positive* is even a difficult task for (native) speakers of Luxembourgish.

To conclude, it can be said that using the $F_1$ score for evaluation is a valid approach. Nevertheless, I argue that it is not sufficient to evaluate the performance of the experiments presented in chapter 4, as the quality and consistency of the corpus input and its sentiment annotations cannot be assured. In chapters 2.1 and 2.2, I discussed that perceiving sentiment as a scale ranging from *negative* to *neutral* and to *positive* is difficult and does not always lead to agreement, even among (native) speakers. A potential reason for this could be that people have diverse ideas of what *positivity, negativity* or *neutrality* means or maybe people have different standpoints when reading the same piece of text. Also, it could be that the user comments this thesis works with do not include enough cases which clearly fit into those categories. Future research should therefore be done on providing clear definitions for the expression of sentiments in general and more precisely on the creation of a well-annotated Luxembourgish sentiment corpus, as the approach for sentiment annotation taken in this thesis might be too broad to adequately grasp the way sentiment is expressed in Luxembourgish. In the future, other categories such as sarcastic or conditional sentences, questions and facts could also be considered. In addition to simply working with sentiment labels on sentence- and user comment-level, this thesis also studied how linguistic features can contribute to the overall sentiment of a larger semantic unit. The aim of this was to move towards an answer for the following research question:

*- Does adding Part-of-Speech as additional linguistic features help to improve the automatic detection of sentiment?*

Research on sentiment analysis for other languages (e.g. Hatzivassiloglou and McKeown (1997), Farzindar and Inkpen (2015) and Taboada (2016)) has shown that specific POS like adjectives and nouns play an important role for the expression of sentiments. In order to study the impact of linguistic features for Luxembourgish sentiment utterances, I built upon the work done by Sirajzade et al. (2020b) and carried out a multilayered ML experiment taking adjectives, adverbs, verbs, nouns, negation particles and a combination of those POS as input features for training and testing with DT, RF, SVM, KNN and LRM. The intuition behind this experimental setup was to examine whether POS can be beneficial and boost the performance of classification. This would be especially convenient for a low-resource context, as manually labeling sentences or user comments is a labor- and time-intensive process that could benefit from adding high quality external resources to the sentiment training process. More precisely, the intuition behind this is that POS could be used to automatically label sentences for sentiment. Those automatically labeled sentences could be used as a way to pre-select sentences with high sentiment content for manual annotation. Unfortunately, using POS as additional features did not improve the classification performance for the experiment carried out as part of this research (see chapter 4.3.4.2). A factor having an impact on this could be that all POS used as features for the experiment were automatically tagged in the corpus with the LuNa tool (Sirajzade and Schommer (2019)). While this tool was specifically built for working with Luxembourgish texts, it does not perform perfectly and therefore includes some noise during the automatic

generation of POS tags. The experiment working with linguistic features thus showed that POS do not help to improve the automatic detection of sentiment. Choosing to add POS tags as input features for a low-resource NLP task should therefore be carefully considered, as it does not necessarily help and could consume human resources that would be better used in other parts of the process of an automatic sentiment analysis system.

To sum up, this thesis contributes to the research in the field of NLP by setting the cornerstone for sentiment analysis in the low-resource language Luxembourgish. Above, I discussed the results of my work with reference to the research questions. While all of the results stated here are interesting and relatable for working with low-resource settings, there are two major findings of those that could best be transferred to other low-resource languages: In one of the ML experiments, a variety of linguistic features were used as additional input features for training (see chapter 4.3.4). Those linguistic features were obtained from automatically labelling the corpus with POS tags using the LuNa tool (Sirajzade and Schommer (2019)). In comparison to other experiments, this one did not perform the best although it was the one providing the most explicit features. It can therefore be assumed that using linguistic features does not automatically lead to a better performance for such a classification system. Moreover, it was overall shown that taking larger corpus sections for training leads to higher $F_1$ scores in evaluation. Henceforth, collecting as much data as possible, even if it includes automatically annotated samples, could be a promising solution for lower-resource settings.

In this research, the first sentiment detection for Luxembourgish user comments was carried out. Aiming to tackle as many challenges as possible, several resources for sentiment analysis were built and a multilayered experimental setup was constructed to observe which kind of learning approach can be best suitable for Luxembourgish sentiment detection. The contribution of this work sets an important milestone for the field and has been extensively documented in this thesis in order to allow building upon my work. Due to the time constraint and the difficulty of the task, no perfectly working sentiment engine could be presented as the outcome of this thesis. In the following section, future work will be discussed that is seen as essential for improving the results of this thesis.

## 5.2 Future Work

As this thesis has performed the first ever sentiment analysis for the Luxembourgish language, challenges remain that could be resolved as part of future work. One crucial aspect is the improvement of the annotations. As stated above, a large part of the annotations that were used for this thesis were automatically collected by means of a lexicon-based or an ANN approach. Applying such methods comes with the crucial drawback that the quality of the annotations is difficult to control. Therefore, improving the annotation approach in general is important: First of all, the calculation of the IAA (see chapter 3.4.2) suggested that the annotation guidelines are not clear enough for annotators to agree on how a *negative, neutral* or *positive* instance is expressed. The guidelines for annotation should therefore be refined by moving away from the intuitive approach taken in this the-

sis that sentiment should be annotated from the perspective of the author (inspired by (Abdul-Mageed and Diab, 2011, p. 23)). It should be moved towards a more fine-grained definition, e.g. by giving more examples to the annotators as to how *negative, neutral* and *positive* are expressed in the language. This is especially important, as such a simplified approach does not always allow a clear distinction between the author's and the reader's perspective. A different way to improve the proposed sentiment analysis approach would be to introduce a more fine-grained scale for annotation, as was for example shown in the work by Kapukaranov and Nakov (2015) and Øvrelid et al. (2020). Kapukaranov and Nakov (2015) work with a scale of 11 parts ranging from 0 to 0.5, 1 and 4.5. Øvrelid et al. (2020) annotate their Norwegian corpus with polar expressions as well as targets and holders of the opinions expressed. There are a lot of different manners of expanding the sentiment scale from *negative, neutral* and *positive* to a larger span of potential sentiment values. Within the framework of the study with the RTL corpus, one additional approach to annotation could also be to add the layer of *objective* or *subjective* to the labeling process. An arbitrary sentence such as *Haut reent et.* 'It rains today.' could be labeled as *neutral_objective* and a sentence such as *Ech denken datt dat Wieder weeder gutt nach schlecht ass.* 'I think this weather is neither good nor bad.' as *neutral_subjective* to account for the fact that the speaker might have their own opinion about the weather situation.

Additionally, it was argued in chapter 2.1 that the definition of sentiment generally assumed by researchers is that language can be classified into categories, such as the category of *negative, neutral* and *positive* sentiment. In future work, it could be further examined whether this categorization is fair to represent the diversity of human language and expressions of feelings, sentiments and emotions. The calculation of the IAA for the sentences in this thesis (chapter 3.4.2) certainly suggests that *negative, neutral* and *positive* are not generally consentaneous concepts and disagreement within the categorization can occur even among speakers of the same language. The introduction and considerable size of the *undecided* class in the *corpus4* and *corpus5* sections also underline this observation (see chapters 3.6.2.3 and 3.6.2.4). Moreover, recent work has raised the argument that disagreement among annotators of the same pieces of texts is ubiquitous and comparing the predictions of a system against a gold standard is outdated (Basile et al., 2021, p. 15). Discarding the data associated with disagreement, like I did in this thesis, can result in better evaluation scores, "but [...] hides the true nature of the task we are trying to solve." (Basile et al., 2021, p. 16). Thus future work on sentiment analysis for Luxembourgish should not only study how to go beyond the simple approach of having *negative, neutral* and *positive* classes but also how disagreeing cases in annotation can be adequately dealt with.

As described above in section 5.1, one of the research contributions of this work was to create a lexicon-based sentiment annotation system that leads to the labeling of 123,030 Luxembourgish sentences of the RTL corpus with a *negative, neutral* or *positive* label (see chapter 3.5.6). The first step of this process was the translation of the German PolArt (Klenner et al. (2009)) to Luxembourgish. While both languages share some related history (e.g. Gilles, in press, p. 4), simply translating a whole dictionary with all its sentiment

values from one language to another could lead to problems, as expressions of emotive instances such as sentiment vary from culture to culture. For this reason, future work can include a carefully handcrafted Luxembourgish sentiment lexicon. This could be done, for instance, by recruiting native speakers and asking them to label POS such as adjectives and nouns with a predefined range of sentiment values. Additionally, it should be noted that a lexicon-based approach comes with one crucial drawback, namely that the change of sentiment of words affected by the context is not taken into account. If a lexicon-based system does not include negation rules, it is incapable of correctly processing sentences such as *Den Omar ass frou.* 'Omar is happy.' and *Den Omar ass net frou.* 'Omar is not happy.'. They both have the *positive* ADJ *frou* 'happy', but only the first sentence expresses the sentiment of *positive*. Such aspects could be added to a new Luxembourgish sentiment lexicon in form of handcrafted rules.

Another possibility to improve the proposed sentiment detection systems in this thesis goes beyond simply refining the annotations and moves towards a more fine-grained approach of sentiment analysis in general. For instance, an aspect-based sentiment system for Luxembourgish, as was briefly described in chapter 2.3.4, could be implemented (see e.g. Hoang et al. (2019) and Zhang et al. (2019)). That way, sentences such as *Haut ass d'Wieder schéin, mee ech fille mech awer schlecht.* 'The weather is nice today, but I still feel bad.' could be tagged with two different sentiment labels, one *positive* concerning the weather and one *negative* concerning the personal feeling of the speaker. Nevertheless, it should not be forgotten that even such fine-grained approaches like aspect-based sentiment analysis stay an approximation and may not capture every possible expression of sentiment in its whole. In Sirajzade et al. (2020b), the idea of setting up a hybrid system for Luxembourgish sentiment analysis relying on linguistic information and ML techniques was already raised. This linguistic information could be stored in a variety of different manners in order to provide an external, high quality knowledge base to benefit an ML process. Such a knowledge base could be inspired by other semantic resources for sentiment analysis such as SentiWordNet (Esuli and Sebastiani (2006)), SenticNet (Cambria et al. (2016)) and WordNetAffect (Strapparava and Valitutti (2004)) which are well-developed for bigger languages such as English, but not for Luxembourgish. A different approach which was not studied in this thesis is the usage of cross-lingual techniques for sentiment analysis (e.g. Sazzed (2020) and Kandula and Min (2021)). Future work could examine whether this helps to improve the quality of sentiment detection for Luxembourgish texts.

As was stated earlier, the RTL corpus covers a wide range of topics and includes texts written by a large variety of authors published over a time span of several years (1999 to 2018). Such a corpus opens the possibility to work on several different aspects of NLP research and not just sentiment analysis. One possibility would be to carry out topic modelling on the data set in order to discover the distribution of different subjects in the corpus. As a next step, topic modelling could be combined with sentiment analysis to show how topics and their related sentiments change over time. It would be interesting, for instance, to model how the sentiment towards a politician such as the prime minister of Luxembourg, Xavier Bettel, evolved from the start of his election campaign to today. Additionally, socio-demographic factors such as age or gender could be taken into account

to implement a more fine-grained study. That way, questions such as *How did the sentiment of women towards Xavier Bettel change over time?* could be answered. A good starting point for planning a topic modelling process is, for instance, Chauhan and Shah (2021) in order to get a first overview of the research field. To proceed, studying already existing projects combining sentiment analysis with topic modelling can give further inspiration as to how to combine those two research fields (e.g. Poddar et al. (2017), Naskar et al. (2016) and Nguyen and Shirai (2015)).

Last but not least, it should be mentioned again that the textual basis of this thesis was a large corpus of user comments and news articles which was provided by RTL Luxembourg, the partner of the *STRIPS* project. Therefore, the question arises whether the work presented here could result in a possible application for RTL Luxembourg. As was described in the different ML experiments in chapter 4, the $F_1$ score achieved during evaluation greatly differs among algorithms and training and testing were carried out with a large class imbalance. A well-functioning sentiment detection system that is applied to real world problems should be capable of attributing sentiment to every potential class, i.e. *negative, neutral* and *positive* in this case, without leaning towards one of them. For this reason, I believe that future work needs to be done to particularly improve the quality of annotations and the balance of the classes in the corpus in order for the work to be applied to real-world problems. Additionally, sentiment annotations would need to be extended from sentence to full user comments level, as this better meets the desires of a user of the RTL Luxembourg platform. After having improved the quality of annotations, a sentiment detection system using fine-tuned *BERT* would be the best starting point for detecting *negative, neutral* and *positive* in Luxembourgish user comments. More precisely, there are two concrete application fields that are relevant: First, a well-functioning sentiment detection system could be used for internal purposes at RTL Luxembourg in order to filter new incoming user comments containing extreme *negativity*. That way, potential hate speech or spam could be filtered out and redirected to a designated team for further inspection (see e.g. Cinelli et al. (2021) for a recent work on hate speech detection in YouTube comments). Second, an application for external usage, i.e. for the users of the `www.rtl.lu` news platform, is also possible. If developed further, RTL could propose a tool which allows users to query for the development of sentiment over time according to different topics.

To recap, this thesis proposes a variety of ML techniques which can be used in order to detect sentiment expressions in texts written in Luxembourgish. Due to its pioneering work in the field, no ultimate solution for automatic sentiment analysis could be offered. It was shown that introducing additional linguistic features into a system does not necessarily yield better results (see chapter 4.3.4.2). In other words, adding POS information to an ML technique does not help to improve the sentiment detection in a corpus. Instead, using a more complex system such as *BERT* combined with a large training corpus for fine-tuning leads to the best results when working with textual data similar to the one studied in the thesis.

# Bibliography


Muhammad Abdul-Mageed and Mona Diab. Linguistically-motivated subjectivity and sentiment annotation and tagging of Modern Standard Arabic. *International Journal on Social Media MMM: Monitoring, Measurement, and Mining*, 2011.

Wazir Ali, Naveed Ali, Yong Dai, Jay Kumar, Saifullah Tumrani, and Zenglin Xu. Creating and Evaluating Resources for Sentiment Analysis in the Low-resource Language: Sindhi. In *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 188–194, Online, April 2021. Association for Computational Linguistics.

Ron Artstein and Massimo Poesio. Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, 34(4):555–596, 2008.

Alexandra Balahur and Marco Turchi. Multilingual Sentiment Analysis using Machine Translation? In *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis*, pages 52–60, Jeju, Korea, July 2012. Association for Computational Linguistics.

Valentin Barriere and Alexandra Balahur. Improving Sentiment Analysis over non-English Tweets using Multilingual Transformers and Automatic Translation for Data-Augmentation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 266–271, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

Valerio Basile, Michael Fell, Tommaso Fornaciari, Dirk Hovy, Silviu Paun, Barbara Plank, Massimo Poesio, and Alexandra Uma. We Need to Consider Disagreement in Evaluation. In *Proceedings of the 1st Workshop on Benchmarking: Past, Present and Future*, pages 15–21, Online, August 2021. Association for Computational Linguistics.

Roy Baumeister, Ellen Bratslavsky, Catrin Finkenauer, and Kathleen D. Vohs. Bad Is Stronger than Good. *Review of General Psychology*, 5:323–370, 12 2001.

Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python.* O'Reilly Media, 2009.

Gerd Bohner and Nina Dickel. Attitudes and attitude change. *Annual Review of Psychology*, 62:391–417, 01 2011.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

Josy Braun, Marianne Johanns-Schlechter, Josée Kauffmann-Frantz, Henri Losch, and Geneviève Magnette-Barthel. *Grammaire de la langue luxembourgeoise*. Ministère de l'Éducation nationale et de la Formation professionnelle, 2005.

Josy Braun, Marianne Johanns-Schlechter, Josée Kauffmann-Frantz, Henri Losch, and Geneviève Magnette-Barthel. *Grammaire de la langue luxembourgeoise*. Ministère de l'Éducation nationale et de la Formation professionnelle, 2020.

Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, October 2001.

Jose Camacho-Collados and Mohammad Taher Pilehvar. On the Role of Text Preprocessing in Neural Network Architectures: An Evaluation Study on Text Categorization and Sentiment Analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 40–46, Brussels, Belgium, November 2018. Association for Computational Linguistics.

Erik Cambria, Soujanya Poria, Rajiv Bajpai, and Bjoern Schuller. SenticNet 4: A Semantic Resource for Sentiment Analysis Based on Conceptual Primitives. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2666–2677, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee.

Kai-Uwe Carstensen, Susanne Jekat, and Ralf Klabunde. Computerlinguistik - Was ist das? In Kai-Uwe Carstensen, Christian Ebert, Cornelia Ebert, Susanne Jekat, Ralf Klabunde, and Hagen Langer, editors, *Computerlinguistik und Sprachtechnologie : eine Einführung*, pages 1–25. Spektrum Akademischer Verlag, Heidelberg, 3 edition, 2010.

Uttam Chauhan and Apurva Shah. Topic Modeling Using Latent Dirichlet allocation: A Survey. *ACM Computing Surveys*, 54(7):1 – 35, 2021.

Juntian Chen, Yubo Tao, and Hai Lin. Visual exploration and comparison of word embeddings. *Journal of Visual Languages and Computing*, 48:178–186, 2018.

Matteo Cinelli, Andraž Pelicon, Igor Mozetič, Walter Quattrociocchi, Petra Kralj Novak, and Fabiana Zollo. Dynamics of online hate and misinformation. *Scientific Reports*, 11, 11 2021.

Tobias Daudert, Manel Zarrouk, and Brian Davis. CoSACT: A Collaborative Tool for Fine-Grained Sentiment Annotation and Consolidation of Text. In *Proceedings of the First Workshop on Financial Technology and Natural Language Processing*, pages 34–39, Macao, China, August 2019.

Deepset. German BERT, 2019. URL `https://huggingface.co/bert-base-german-cased`. Accessed: 2021-04-09.

Anindita Desarkar and Ajanta Das. *Big-Data Analytics, Machine Learning Algorithms and Scalable/Parallel/Distributed Algorithms*, pages 159–197. 01 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

Eduard Dragut and Christiane Fellbaum. The Role of Adverbs in Sentiment Analysis. In *Proceedings of Frame Semantics in NLP: A Workshop in Honor of Chuck Fillmore (1929-2014)*, pages 38–41, Baltimore, MD, USA, June 2014. Association for Computational Linguistics.

Nathalie Entringer, Peter Gilles, Sara Martin, and Christoph Purschke. Schnëssen. Surveying language dynamics in Luxembourgish with a mobile research app. *Linguistic Vanguard. (2020).*, 2020.

Gülşen Eryiğit, Fatih Samet Çetin, Meltem Yanık, Tanel Temel, and İlyas Çiçekli. TURKSENT: A Sentiment Annotation Tool for Social Media. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 131–134, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

Andrea Esuli and Fabrizio Sebastiani. SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy, May 2006. European Language Resources Association (ELRA).

Atefeh Farzindar and Diana Inkpen. *Natural Language Processing for Social Media*. Morgan & Claypool Publishers, 2015.

John Rupert Firth. *Studies in linguistic analysis*, chapter A synopsis of linguistic theory, pages 1–32. Basil Blackwell Oxford, 1962.

Alcides Fonseca and Bruno Cabral. *Handbook of Deep Learning Applications*, volume 136 of *Smart Innovation, Systems and Technologies*, chapter Desigining a Neural Network from Scratch for Big Data Powered by Multi-node GPUs, pages 1–20. Springer, 2019.

Barbara L. Fredrickson. What Good Are Positive Emotions? *Rev Gen Psychol*, 2(3): 300–319, September 1998.

Svitlana Galeshchuk, Ju Qiu, and Julien Jourdan. Sentiment Analysis for Multilingual Corpora. In *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, pages 120–125, Florence, Italy, August 2019. Association for Computational Linguistics.

Daniela Gierschek, Peter Gilles, Christoph Purschke, Christoph Schommer, and Joshgun Sirajzade. A Temporal Warehouse for Modern Luxembourgish Text Collections. In *DH Benelux*, 2019. URL `https://orbilu.uni.lu/bitstream/10993/41840/1/DH_Benelux_2019_paper_22%20(1).pdf`. Accessed: 2020-09-28.

Peter Gilles. From status to corpus: Codification and implementation of spelling norms in Luxembourgish. *In W., Davies and E., Ziegler (Eds.), Macro and micro language planning (pp. 128-149). London: Palgrave Macmillan (2015).*, 2015.

Peter Gilles. Luxembourgish. *In P., Maitz, H. C., Boas (Ed.), A., Deumert (Ed.) and M., Louden (Ed.), Varieties of German Worldwide. Oxford: Oxford University Press.*, in press.

Google. Natural Language API. `https://cloud.google.com/natural-language/docs/analyzing-sentiment`. Accessed: 2021-03-22.

Carl-Friedrich Graumann. *Sociolinguistics an international handbook of the science of language and society = Soziolinguistik : ein internationales Handbuch zur Wissenschaft von Sprache und Gesellschaft*, volume Volume 2 = 2. Teil. of *Handbücher zur Sprach- und Kommunikationswissenschaft ; Bd. 3.2*, chapter Sozialpsychologie/Social Psychology, pages 865–870. Mouton de Gruyter, Berlin ; New York, second completely rev. & extended ed.. edition, 2005.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer New York, New York, NY, second edition, 2017.

Vasileios Hatzivassiloglou and Kathleen R. McKeown. Predicting the Semantic Orientation of Adjectives. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 174–181, Madrid, Spain, July 1997. Association for Computational Linguistics.

Vasileios Hatzivassiloglou and Janyce M. Wiebe. Effects of Adjective Orientation and Gradability on Sentence Subjectivity. In *COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics*, 2000.

Johannes Hellrich. *Word embeddings : reliability  semantic change*. Frontiers in artificial intelligence and applications. Dissertations in artificial intelligence ; Volume 347. 2019.

Geoffrey E. Hinton. Learning distributed representations of concepts. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, pages 1–12, 1986.

Mickel Hoang, Oskar Alija Bihorac, and Jacobo Rouces. Aspect-Based Sentiment Analysis using BERT. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 187–196, Turku, Finland, September–October 2019. Linköping University Electronic Press.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spaCy: Industrial-strength Natural Language Processing in Python, 2020.

Anette Huesmann. *Zwischen Dialekt und Standard : Empirische Untersuchung zur Soziolinguistik des Varietatenspektrums im Deutschen.* Reihe Germanistische Linguistik ; 199. Reprint edition, 2017.

Hayeon Jang and Hyopil Shin. Effective Use of Linguistic Features for Sentiment Analysis of Korean. In *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation*, pages 173–182, Tohoku University, Sendai, Japan, November 2010. Institute of Digital Enhancement of Cognitive Processing, Waseda University.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, 2017.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of Tricks for Efficient Text Classification. *arXiv preprint arXiv:1607.01759*, 2016.

Daniel Jurafsky and James H. Martin. Speech and Language Processing. Draft of December 30, 2020. URL `https://web.stanford.edu/~jurafsky/slp3/5.pdf`.

Daniel Jurafsky and James H. Martin. *Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Pearson Education, USA, 2nd edition, 2009.

Hemanth Kandula and Bonan Min. Improving Cross-Lingual Sentiment Analysis via Conditional Language Adversarial Nets. In *Proceedings of the Third Workshop on Computational Typology and Multilingual NLP*, pages 32–37, Online, June 2021. Association for Computational Linguistics.

Mehmed Kantardzic. *Data Mining. Concepts, Models, Methods, and Algorithms.* IEEE Press, third edition, 2020.

Borislav Kapukaranov and Preslav Nakov. Fine-Grained Sentiment Analysis for Movie Reviews in Bulgarian. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 266–274, Hissar, Bulgaria, September 2015. INCOMA Ltd. Shoumen, Bulgaria.

Kian Kenyon-Dean, Eisha Ahmed, Scott Fujimoto, Jeremy Georges-Filteau, Christopher Glasz, Barleen Kaur, Auguste Lalande, Shruti Bhanderi, Robert Belfer, Nirmal Kanagasabai, Roman Sarrazingendron, Rohit Verma, and Derek Ruths. Sentiment Analysis: It's Complicated! In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1886–1895, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

Jason Kessler. Scattertext: a Browser-Based Tool for Visualizing how Corpora Differ. In *Proceedings of ACL 2017, System Demonstrations*, pages 85–90, Vancouver, Canada, July 2017. Association for Computational Linguistics.

Svetlana Kiritchenko and Saif Mohammad. The Effect of Negators, Modals, and Degree Adverbs on Sentiment Composition. In *Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 43–52, San Diego, California, June 2016. Association for Computational Linguistics.

Manfred Klenner, Angela Fahrni, and Stefanos Petrakis. PolArt: A Robust Tool for Sentiment Analysis. In *Proceedings of the 17th Nordic Conference of Computational Linguistics (NODALIDA 2009)*, pages 235–238, Odense, Denmark, May 2009. Northern European Association for Language Technology (NEALT).

Mohammed Korayem, David Crandall, and Muhammad Abdul-Mageed. Subjectivity and Sentiment Analysis of Arabic: A Survey. volume 322, pages 128–139, 12 2012.

Klaus Krippendorff. Computing Krippendorff 's Alpha-Reliability. 2011.

Adrienne Lehrer. *Semantic fields and lexical structure*, volume 11 of *North-Holland Linguistic Series*. Amsterdam : North-Holland ; New York : American Elsevier, 1974.

Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.

Jiwei Li and Eduard H. Hovy. Reflections on Sentiment/Opinion Analysis. *ArXiv*, abs/1507.01636, 2015.

Christine Liebrecht, Florian Kunneman, and Antal van den Bosch. The perfect solution for detecting sarcasm in tweets #not. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 29–37, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

Bing Liu. *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge University Press, 2015.

Henning Lobin. *Computerlinguistik und Texttechnologie*. Fink, Paderborn, 2010.

lushan88a. google_trans_new, 2020. URL `https://github.com/lushan88a/google_trans_new`. Accessed: 2020-12-07.

John Lyons. *Introduction to theoretical linguistics*. Cambridge University Press, second edition, 1969.

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 1999.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, USA, 2008.

Stephen Marsland. *Machine Learning. An Algorithmic Perspective.* Machine Learning & Pattern Recognition Series. CRC Press, second edition, 2015.

J.R. Martin and David Rose. Working with Discourse : meaning beyond the clause, 2013.

Angel R. Martinez. Natural language processing. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(3):352–357, 2010.

David Mertz. *Cleaning Data for Effective Data Science.* Packt Publishing, 1st edition, 2021.

Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013b.

Saif Mohammad. A Practical Guide to Sentiment Annotation: Challenges and Solutions. In *Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 174–179, San Diego, California, June 2016. Association for Computational Linguistics.

Saif Mohammad. *Challenges in Sentiment Analysis*, pages 61–83. 04 2017.

Kevin Mulligan and Klaus Scherer. Toward a Working Definition of Emotion. *Emotion Review*, 4:345–357, 09 2012.

M. Munezero, C. S. Montero, E. Sutinen, and J. Pajunen. Are They Different? Affect, Feeling, Emotion, Sentiment, and Opinion Detection in Text. *IEEE Transactions on Affective Computing*, 5(2):101–111, 2014.

Kevin P. Murphy. *Machine Learning. A Probabilistic Perspective.* MIT Press, 2012.

Andreas C. Müller and Sarah Guido. *Einführung in Machine Learning mit Python.* 1st edition, 2017.

Debashis Naskar, Sidahmed Mokaddem, Miguel Rebollo, and Eva Onaindia. Sentiment Analysis in Social Networks through Topic modeling. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 46–53, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA).

Günter Neumann. Text-basiertes Informationsmanagement. In Kai-Uwe Carstensen, Christian Ebert, Cornelia Ebert, Susanne Jekat, Ralf Klabunde, and Hagen Langer, editors, *Computerlinguistik und Sprachtechnologie : eine Einführung*, pages 576–615. Spektrum Akademischer Verlag, Heidelberg, 3 edition, 2010.

Thien Hai Nguyen and Kiyoaki Shirai. Topic Modeling based Sentiment Analysis on Social Media for Stock Market Prediction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1354–1364, Beijing, China, July 2015. Association for Computational Linguistics.

Shigehiro Oishi and Jaime Kurtz. *The Positive Psychology of Positive Emotions*, pages 101–114. 01 2011.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. A Fine-grained Sentiment Dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France, May 2020. European Language Resources Association.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? Sentiment Classification using Machine Learning Techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 79–86. Association for Computational Linguistics, July 2002.

Natalie Parde and Rodney Nielsen. Detecting Sarcasm is Extremely Easy ;-). In *Proceedings of the Workshop on Computational Semantics beyond Events and Roles*, pages 21–26, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'AlchéBuc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

Telmo Pires, Eva Schlinger, and Dan Garrette. How Multilingual is Multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy, July 2019. Association for Computational Linguistics.

Lahari Poddar, Wynne Hsu, and Mong Li Lee. Author-aware Aspect Topic Sentiment Model to Retrieve Supporting Opinions from Reviews. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 472–481, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. SemEval-2014 Task 4: Aspect Based Sentiment Analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland, August 2014. Association for Computational Linguistics and Dublin City University.

Christoph Purschke. REACT – Einstellungen als evaluative Routinen in sozialen Praxen. *Sprechen über Sprache : Perspektiven und neue Methoden der Spracheinstellungsforschung*, pages 123–142, 2014.

Christoph Purschke. REACT – A constructivist theoretic framework for attitudes. *Responses to Language Varieties: Variability, processes and outcomes*, pages 37–54, 2015.

Christoph Purschke. Attitudes Toward Multilingualism in Luxembourg. A Comparative Analysis of Online News Comments and Crowdsourced Questionnaire Data. *Frontiers in Artificial Intelligence*, 3:79, 2020a.

Christoph Purschke. spellux - Automatic text normalization for Luxembourgish, 2020b. URL `https://github.com/questoph/spellux`. Accessed: 2020-07-28.

James Pustejovsky and Amber Stubbs. *Natural Language Annotation for Machine Learning*. O'Reilly, China Sebastopol, CA, 2013.

Sudharsan Ravichandiran. *Getting Started with Google BERT*. Packt Publishing, 1st edition, 2021.

Darren Reading. tex-neural-network, 2019. URL `https://github.com/dreading/tex-neural-network`. Accessed: 2021-05-23.

Yashwanth Reddy Regatte, Rama Rohit Reddy Gangula, and Radhika Mamidi. Dataset Creation and Evaluation of Aspect Based Sentiment Analysis in Telugu, a Low Resource Language. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5017–5024, Marseille, France, May 2020. European Language Resources Association.

Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA.

Sara Rosenthal, Preslav Nakov, Svetlana Kiritchenko, Saif Mohammad, Alan Ritter, and Veselin Stoyanov. SemEval-2015 task 10: Sentiment Analysis in Twitter. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 451–463, Denver, Colorado, June 2015. Association for Computational Linguistics.

Emmanuelle Roussel, Adeline Patard, and Rea Peltola. *Cross-Linguistic Perspectives on the Semantics of Grammatical Aspect*, volume 30 of *Cahiers Chronos*. Brill, 2019.

RTL Luxembourg, 2020. URL `https://www.rtl.lu/`. Accessed: 2020-09-11.

RTL Luxembourg. Enttäuscht, datt automatesch Ukënnegungen um Findel op Lëtzebuergesch ewechfalen, 2021. URL `https://www.rtl.lu/news/national/a/1782963.html`. Accessed: 2021-12-19.

Salim Sazzed. Cross-lingual sentiment classification in low-resource Bengali language. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 50–60, Online, November 2020. Association for Computational Linguistics.

Martin Seligman and Mihaly Csikszentmihalyi. Positive Psychology: An Introduction. *The American psychologist*, 55:5–14, 02 2000.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning. From Theory to Algorithms*. Cambridge University Press, 2014.

Erik Siegel and Adam Retter. *eXist: A NoSQL Document Database and Application Platform*. O'Reilly Media, 2014.

Joshgun Sirajzade. Korpusbasierte Untersuchung der Wortbildungsaffixe im Luxemburgischen. Technische Herausforderungen und linguistische Analyse am Beispiel der Produktivität. *Zeitschrift für Wortbildung = Journal of Word Formation, 2(1)*, 2018.

Joshgun Sirajzade and Christoph Schommer. The LuNa Open Toolbox for the Luxembourgish Language. *In Petra Perner (Ed.), 19th Industrial Conference, ICDM 2019 New York, USA, July 17 to July 21 2019, Poster Proceedings 2019, Advances in Data Mining, Applications and Theoretical Aspects*, 2019.

Joshgun Sirajzade, Daniela Gierschek, and Christoph Schommer. An Annotation Framework for Luxembourgish Sentiment Analysis. In *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 172–176, Marseille, France, May 2020a. European Language Resources association.

Joshgun Sirajzade, Daniela Gierschek, and Christoph Schommer. Component Analysis of Adjectives in Luxembourgish for Detecting Sentiments. In *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 159–166, Marseille, France, May 2020b. European Language Resources association.

Carlo Strapparava and Alessandro Valitutti. WordNet Affect: an Affective Extension of WordNet. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal, May 2004. European Language Resources Association (ELRA).

Maite Taboada. Sentiment Analysis: An Overview from Linguistics. *Annual Review of Linguistics*, 2(1):325–347, 2016.

Maite Taboada and Jack Grieve. Analysing Appraisal Automatically. *In Proceeding of AAAI Spring Symposium on Exploring Attitude and Affect in Text*, pages 158–161, 04 2004.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics*, 37:267–307, 06 2011.

TNS Ilres and Kantar Belgium. TNS Ilres Plurimedia 2021.II, 2021. URL `https://www.tns-ilres.com/news/tns-ilres/2021/etude-tns-ilres-plurimedia-luxembourg-2021-ii/`. Accessed: 2021-12-20.

Michal Toman, Roman Tesar, and Karel Jezek. Influence of Word Normalization on Text Classification. 01 2006.

Amazon Mechanical Turk. Amazon Mechanical Turk. Access a global, on-demand, 24x7 workforce. `https://www.mturk.com/`. Accessed: 2020-07-08.

Peter Turney. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 417–424, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

Alper Kürşat Uysal and Serkan Gunal. The impact of preprocessing on text classification. *Information Processing and Management*, 50:104 – 112, 01 2014.

Venelin Valkov. Sentiment Analysis with BERT and Transformers by Hugging Face using PyTorch and Python, 2020. URL `https://curiousily.com/posts/sentiment-analysis-with-bert-and-hugging-face-using-pytorch-and-python/`. Accessed: 2021-04-08.

Sonja Vandermeeren. *Sociolinguistics an international handbook of the science of language and society = Soziolinguistik : ein internationales Handbuch zur Wissenschaft von Sprache und Gesellschaft*, volume Volume 2 = 2. Teil. of *Handbücher zur Sprach- und Kommunikationswissenschaft ; Bd. 3.2*, chapter Research on Language Attitudes /Spracheinstellungsforschung, pages 1318–1332. Mouton de Gruyter, Berlin ; New York, 2nd completely rev. & extended ed.. edition, 2005.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings*

*of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

David Wolpert. The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8:1341–1390, 03 1996.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR*, abs/1609.08144, 2016.

Hamed Yaghoobian, Hamid R. Arabnia, and Khaled Rasheed. Sarcasm Detection: A Comparative Study, 2021.

Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 23–30. Association for Computational Linguistics, 2020.

Seid Muhie Yimam, Iryna Gurevych, Richard Eckart de Castilho, and Chris Biemann. WebAnno: A Flexible, Web-based and Visually Supported System for Distributed Annotations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 1–6, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

Mohammed J. Zaki and Wagner Meira Jr. *Data Mining and Machine Learning. Fundamental Concepts and Algorithms.* Cambridge University Press, second edition, 2020.

Zenter fir d'Lëtzebuerger Sprooch. *D'Lëtzebuerger Orthografie.* SCRIPT ZLS, Luxembourg, 2019.

Zenter fir d'Lëtzebuerger Sprooch. Lëtzebuerger Online Dictionnaire, 2020a. URL `https://lod.lu/`. Accessed: 2020-10-09.

Zenter fir d'Lëtzebuerger Sprooch. *D'Lëtzebuerger Verben.* SCRIPT ZLS, Luxembourg, 2020b.

Chen Zhang, Qiuchi Li, and Dawei Song. Aspect-based Sentiment Classification with Aspect-specific Graph Convolutional Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4568–4578, Hong Kong, China, November 2019. Association for Computational Linguistics.

Yue Zhang and Zhiyang Teng. *Natural Language Processing: A Machine Learning Perspective*. Cambridge University Press, 2021.