FH University of Applied Sciences TECHNIKUM WIEN

Master Thesis

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Engineering at the University of Applied Sciences Technikum Wien – Degree Program Software Engineering

Multilingual Sentiment Analysis of News Texts Using a Generative Large Language Model

By: Dipl.-Übers. Diane Keller, BSc
Student Number: 51867650

Supervisors: Ing. Dr.techn. Dominik Dolezal, MSc
Dipl.-Ing. Dr.techn. Gerhard Backfried

Vienna, May 6, 2025

Declaration

“As author and creator of this work to hand, I confirm with my signature knowledge of the relevant copyright regulations governed by higher education acts (see Urheberrechtsgesetz /Austrian copyright law as amended as well as the Statute on Studies Act Provisions / Examination Regulations of the UAS Technikum Wien as amended).

I hereby declare that I completed the present work independently and that any ideas, whether written by others or by myself, have been fully sourced and referenced. I am aware of any consequences I may face on the part of the degree program director if there should be evidence of missing autonomy and independence or evidence of any intent to fraudulently achieve a pass mark for this work (see Statute on Studies Act Provisions / Examination Regulations of the UAS Technikum Wien as amended).

I further declare that up to this date I have not published the work to hand nor have I presented it to another examination board in the same or similar form. I affirm that the version submitted matches the version in the upload tool.”

Vienna, May 6, 2025 Signature

SIGNATURINFORMATIONEN	**Unterzeichner**	Diane Evelyn Keller
	Datum/Zeit-UTC	2025-05-06T23:54:36+0200
	Prüfinformation	Informationen zur Prüfung der elektronischen Signatur finden Sie unter: https://www.signaturpruefung.gv.at
Hinweis	Dieses mit einer qualifizierten elektronischen Signatur versehene Dokument hat gemäß Art. 25 Abs. 2 der Verordnung (EU) Nr. 910/2014 vom 23. Juli 2014 ("eIDAS-VO") die gleiche Rechtswirkung wie ein handschriftlich unterschriebenes Dokument.	

Kurzfassung

Nachrichtentexte spielen eine zentrale Rolle in der öffentlichen Meinungsbildung, enthalten jedoch oft nur implizite Sentiments, was deren Analyse besonders anspruchsvoll macht. Bisherige Ansätze zur Sentiment-Analyse erfordern meist aufwendig trainierte, sprach- und domänenspezifische Modelle, die für Nachrichtentexte bislang kaum existieren. Große generative Sprachmodelle (LLMs) könnten hier eine Alternative bieten, da sie breit trainiert, mehrsprachig einsetzbar und für viele Aufgaben ohne Feintuning nutzbar sind. Die vorliegende Arbeit untersucht, ob LLMs wie BLOOM ohne zusätzliches domänenspezifisches Feintuning für die Sentiment-Analyse von Nachrichtentexten geeignet sind.

Die Studie folgt dem CRISP-DM-Framework und verwendet das MAD-TSC-Datenset, das professionelle Übersetzungen und zielabhängige Annotationen in acht Sprachen bietet. Als LLM wird BLOOM gewählt, da es kostenfrei zugänglich ist, Multilingualität unterstützt und generalistisches Training erhalten hat, das grundlegende Sentiment-Analyse ermöglichen könnte. Die Sentiment-Analyse erfolgt auf Satzebene. Die Abfragen an BLOOM werden mittels Prompt Engineering entwickelt. Monolinguale Ergebnisse werden anhand standardisierter Metriken evaluiert. Multilinguale Ergebnisse werden entgegen der ursprünglichen Zielsetzung nicht erhoben, da der API-Zugang zu BLOOM kurz vor Beginn des multilingualen Teils des Projekts unerwartet eingestellt wurde.

Die Untersuchung zeigt, dass BLOOM mit geeigneten Prompts Sentiment-Analysen im Englischen mit einer Performance von über 65 % erzielen kann. Domänenspezifisch optimierte BERT-Modelle erzielen jedoch weiterhin deutlich bessere Resultate. Der Einsatz von Chain-of-Thought-Prompting könnte das fehlende Feintuning teilweise ausgleichen, indem schrittweises Denken aktiviert wird. Ein anderer Ansatz könnte die Nutzung sentiment-spezifischer Pipeline-Methoden der Transformers-Bibliothek sein, da diese gezielt auf Trainingswissen des Modells zugreifen können. Solche Ansätze erfordern jedoch zusätzliche Speicher- und Rechenressourcen.

Die mehrsprachige Sentiment-Analyse mit LLMs bleibt ein interessantes Forschungsgebiet mit Potential für die praktische Anwendung. Für einen effizienten und stabilen Betrieb jedoch muss in die Bereitstellung geeigneter Hard- und Softwareressourcen investiert werden. Daneben sind potentiell hohe laufende Kosten zu tragen.

Schlagworte: Sentiment-Analyse, großes Sprachmodell, Nachrichtentexte, zielabhängig, quelloffen, BLOOM, MAD-TSC

Abstract

News texts play a central role in shaping public opinion, yet they often contain only implicit sentiments, which makes their analysis particularly challenging. Existing approaches to sentiment analysis typically rely on extensively trained, language- and domain-specific models, which are still largely unavailable for news texts. Large generative language models (LLMs) could offer an alternative, as they are broadly trained, multilingual, and usable for many tasks without fine-tuning. This study investigates whether LLMs like BLOOM are suitable for sentiment analysis of news texts without additional domain-specific fine-tuning.

The study follows the CRISP-DM framework and uses the MAD-TSC dataset, which provides professional translations and target-dependent annotations in eight languages. BLOOM was selected as the LLM because it is freely accessible, supports multilingualism, and has undergone generalist training that could enable basic sentiment analysis. The sentiment analysis is performed at the sentence level. Queries to BLOOM are developed through prompt engineering. Monolingual results are evaluated using standardized metrics. Contrary to the original plan, multilingual results could not be obtained because API access to BLOOM was unexpectedly discontinued shortly before the multilingual phase of the project began.

The study shows that BLOOM, with appropriate prompts, can achieve sentiment analysis performance above 65% in English. However, BERT models that are optimized for specific domains continue to produce significantly better results. The use of chain-of-thought prompting could partially compensate for the lack of fine-tuning by activating step-by-step reasoning. Another approach could be the use of sentiment-specific pipeline methods from the Transformers library, as these allow for targeted access to the model's training knowledge. However, such approaches require additional storage and computational resources.

Multilingual sentiment analysis with LLMs remains a promising area of research with potential for practical application. However, to ensure efficient and stable operation, investment in suitable hardware and software resources is essential – and ongoing operational costs may be significant.

Keywords: Sentiment analysis, large language model, news texts, target-dependent, open-source, BLOOM, MAD-TSC

Acknowledgements

I would like to express my deepest gratitude to my academic supervisor, Ing. Dr.techn. Dominik Dolezal, MSc., for his intensive support and valuable scientific guidance throughout the course of this research. His focused suggestions were crucial to successfully completing this project within the given timeframe.

A special thanks goes to Dipl.-Ing. Dr.techn. Gerhard Backfried, my technical supervisor, who, with great expertise, dedication, and foresight, directed the development of this project in the right direction and accompanied its implementation with meticulous attention to detail. His valuable insights significantly contributed to improving the quality of the project.

I also extend my thanks to DI Mag. Christian Schmidt for his constructive advice and support, especially during the planning phase of the project.

Last but not least, I would like to express my heartfelt thanks to my family. Without their moral and practical support, this thesis would not have been possible.

Table of Contents

1 Introduction

Sentiment analysis, often referred to as opinion mining, deals with the automatic detection and classification of emotions, attitudes, or opinions in texts [1, p. 702]. While initially focused on domains like product reviews or social media, sentiment analysis now offers potential for broader applications, such as assessing global public opinion reflected in news texts. This thesis explores whether generative large language models (LLMs) can enable multilingual sentiment analysis of news texts without domain-specific fine-tuning.

1.1 Motivation

So far, sentiment analysis has mainly been applied to areas such as product reviews and social media, where subjective expressions of opinion dominate [5, S. 1–6]. In our globalized world, however, sentiment is important not only in relation to products or services but also on a larger scale: events around the world and people's reactions to them are relevant for decisions in politics, business and finance as well as for assessments of the security situation in given regions. News texts play a special role here due to their wide reach. The opinions that news texts reveal about certain individuals, events, or topics influence public opinion. At the same time, the prevailing public opinion is reflected in news texts. Detecting the opinions and sentiments expressed in news texts can therefore uncover shifts in public opinion.

Applying sentiment analysis to news texts offers great potential for insights into public opinion and its development. In particular, the development and research of multilingual analysis systems for news texts is of particular interest. Reliable sentiment analysis in multiple languages is crucial for understanding global reactions, given the global impact of many political, business and financial events and decisions. However, multilingual sentiment analysis faces particular challenges. Developing dedicated sentiment analysis models that, in practice, deliver consistent cross-lingual quality involves a great deal of effort and expense. And, given the difficulty of collecting and processing equivalent training data in languages that are less widespread than English, this task is hardly conceivable without being able to build on existing models.

Indeed, current approaches to the development of sentiment analysis systems typically involve using a pre-trained language model like the Bidirectional Encoder Representations from Transformers (BERT) [2] or one of its variants, and subjecting it to additional, specific training with texts from the targeted domain, optimizing it for the intended task in iterative cycles [2–7]. This method, though still labor-intensive despite pre-training, yields good recognition results. However, the drawback is that such fine-tuning must be performed anew for every additional text type and domain requiring sentiment analysis.

With the recent introduction of large generative language models (LLMs), which are trained on transformer-based architectures in countless cycles with particularly large and diverse datasets, this approach may have become unnecessary. For instance, in relation to aspect-based sentiment analysis of traditional sentiment analysis items (reviews of hotels, books, clothing and laptops) in English, Mughal *et al.* conclude:

> „Leveraging LLMs like PaLM and GPT-3.5 eliminates the requirement for labeled datasets in ABSA [i. e. aspect-based sentiment analysis] tasks, streamlining the training process and mitigating the need for high-computation machines across domains.” [8, p. 60957]

For more challenging scenarios, the studies by Radford *et al.* in 2019 [9] already offer hope that large generative language models can even be used in zero-shot scenarios without further fine-tuning, i.e. in situations in which the models are confronted with new tasks in domains for which they have not been specially trained. Generative LLMs show impressive capabilities in understanding and generating natural language. Their multilingual generalization abilities in a wide variety of domains and text types open up new possibilities for sentiment analysis in different languages and domains. In consequence, this thesis aims to investigate whether LLMs can provide a multilingual solution for sentiment analysis of news texts.

1.2 State of the Art

This section defines and delineates the scope of this thesis, provides an overview of the development and current state of research in this field, and identifies the research gap that gives rise to this thesis.

1.2.1 Term and Concept

Sentiment analysis, according to Sagnika *et al.*, is:

> “the process of analyzing opinions or views expressed in documents and their overall classification, scoring or quantification. The primary purpose is to get an idea of people’s general attitude and feelings towards a certain subject.” [10, p. 154]

To this definition, it should be added that sentiment analysis is a subfield of Natural Language Processing (NLP), its goal being the automatic machine recognition and classification of sentiments.

In research and industry, there are also other terms used alongside the English term “sentiment analysis”, such as “opinion mining” [11], “stance detection” [12] or “emotion prediction” [13]. Combinations of terms such as “sentiment”, “opinion”, “stance”, “emotion”, “affect”, “subjectivity” and “review” with words such as “analysis”, “detection”, “extraction”, “mining” and “prediction” can also be found. These terms essentially refer to similar concepts, and often no

clear distinctions are made between them. Liu addresses the problem and attributes the lack of conceptual differentiation to the fact that sentiment analysis is a subject of computer science rather than linguistics. Ultimately, he decides not to differentiate between the two most commonly used terms, “sentiment analysis” and “opinion mining”:

> “[…] there has been some confusion among practitioners and even researchers about the difference between sentiment and opinion and whether the field should be called sentiment analysis or opinion mining. Since the field originated from computer science rather than linguistics, little attention has been given to studying the difference between the two words. […] I use the terms sentiment analysis and opinion mining interchangeably. […] I use the term sentiment to mean the underlying positive or negative feeling implied by an opinion.” [14, p. 2]

This thesis adopts this pragmatic approach for all the aforementioned terms and concepts, as theoretical engagement with the different terms is not the focus of the thesis and would have little to no impact on its practical work. Accordingly, this thesis will consistently use the term “sentiment analysis”, and potential differences between ‘opinion’, ‘stance’, ‘feeling’, ‘mood’, ‘subjective attitude’, etc., will not be addressed. Anything that expresses a positive, neutral, or negative attitude (a “sentiment”) toward an object, a person, an idea, or an issue is considered the subject of sentiment analysis for the purposes of this thesis.

1.2.2 Development of Sentiment Analysis

Since its beginnings in the early 2000s [14, p. 1] , sentiment analysis has evolved considerably. Early approaches were based on dictionary and rule-based methods [15–17], which used predefined word lists and linguistic rules to identify sentiments in texts. However, these approaches are often not flexible enough to capture complex linguistic phenomena [18], and in resource-poor languages, the word lists and rule sets are usually too small to produce good analysis results, as e.g. Sazzed shows for Bengali [19, p. 227]. Therefore, machine learning methods usually produce better results [20].

With the advent of machine learning, statistical methods were introduced that made it possible to train models on large datasets [21]. Decision trees, support vector machines, and later neural networks [1, 10, pp. 160-163, 22] significantly improved the accuracy of sentiment analysis. A key milestone was the introduction of word embeddings such as Word2Vec (which stands for “word to vector”) [23] and GloVe (which is coined from “global vectors”) [24], which model semantic relationships between words.

The development of transformer architectures [25], in particular BERT [2] and generative pre-trained transformers (GPTs) [26, 27], revolutionized the field of natural language processing (NLP). These models use attention mechanisms [25] to effectively leverage contextual

information, leading to significant advancements in various NLP tasks, including sentiment analysis.

1.2.3 Multilingual Sentiment Analysis

In the beginning, sentiment analysis focused primarily on English [28]. As development progressed, the research and application of sentiment analysis was extended to other languages. A key problem in this process is the availability of training data for less widely used languages. To circumvent this problem, methods have been developed in which texts are first machine-translated into English and then analyzed [29]. However, this approach is prone to errors, as translation mistakes can affect the sentiment analysis. Additionally, the quality of machine translation systems also depends on the availability of training data in the relevant language pairs.

More recent approaches use multilingual transformer models such as Multilingual BERT (mBERT) [2] and the cross-lingual language model (XLM) XLM-RoBERTa (XLM-R) [30], which can process multiple languages simultaneously [31]. These models allow for learning shared language representations, enabling sentiment analysis in different languages without specific training for each individual language.

1.2.4 Sentiment Analysis of News Texts

Compared to subjective opinions expressed on social media, news texts present a particular challenge for sentiment analysis. Since this type of text generally strives for objectivity in its presentation of events [32, 33], any sentiments that are present are harder to detect —yet still exist, considering, for example, that newspapers may be perceived as right- or left-leaning [32, 34, 35, p. 3577]. Sentiments in news texts are less likely to be expressed through explicit lexical elements that can be identified using sentiment lexicons [33]. Clear subjectivity markers close to spoken language (e.g. interjections [36, p. 627, 37], expletives [37], expressive punctuation [14, p. 295, 38, p. 17], emojis [39]), which are commonly found in social media posts and facilitate the recognition of sentiments and their polarity, are generally absent, except perhaps in direct quotes. Instead, sentiments are conveyed through more complex stylistic devices such as irony or metaphors, which often require consideration of the wider (textual) co- and (historical, social, cultural, situational, etc.) context [32, 40, 41]. They may also appear in subtler forms of text organization, such as emphasizing facts that support the author's perspective while downplaying or omitting counterarguments [33]. Recognizing such sentiments requires contextual and world knowledge [42], surpassing the capabilities of traditional sentiment analysis. Completely outside the scope of text-based sentiment analysis, finally, are methods used to convey sentiments at the level of entire news outlets, such as the selection of topics, placement of articles within a newspaper or website, reporting style, and choice of perspective on a topic [43].

Presumably due to these specific characteristics of news texts, sentiment analysis on news texts is less common and less successful than analysis on social media posts, and research on this field of application is still not very advanced. One of the earliest works – or perhaps the very first – to apply sentiment analysis to news texts was published in 2008 by Bautin *et al.* [44]. The authors used machine translation to convert news texts from various languages into English and aggregated them over time series to compare opinions on different topics across countries. Despite the poor quality of machine translations at the time, they concluded that aggregation over time compensated for this shortcoming and provided a useful insight into the respective prevailing sentiment.

Since then, there have been only a few approaches to sentiment analysis of news texts, and hardly any of them were multilingual. The vast majority of studies focus exclusively on English. Out of 31 publications on sentiment analysis in news texts between 2008 and 2024, which the author of this thesis identified, 20 focus solely on English. Many of these studies target sentiments in economic and financial news relevant to the stock market rather than general, more politically oriented news: more than a third of the studies for English deal with financial and economic news [45–52]. Sentiment analysis of financial news has also been conducted for Chinese [53].

Only five publications address multiple languages without relying on machine translation into English: in 2012, Kaya, Fidan and Toroslu [54] compared their results for Turkish with comparable results in English, Chan *et al.* [35] included four languages – Arabic, German, English, and Turkish – in their 2020 study, while, in the same year, Pelicon *et al.* [55] analyzed Slovenian and Croatian. In 2023, Mello, Cheema and Thakar [56] studied English and Portuguese Olympic-related news, and in 2024, Dufraisse *et al.* [57] published a sentiment-annotated parallel news corpus in eight languages, namely German, English, Spanish, French, Italian, Dutch, Portuguese and Romanian. Monolingual sentiment analyses for non-English languages have been conducted for Portuguese [58], Slovenian [59], Spanish [60], Hungarian [61] and Chinese [53].

Compared to the performance of sentiment analysis on social media posts, the results achieved in research on sentiment analysis of news texts (see Appendix A) are less convincing, as noted by Hamborg *et al.* in 2019:

> "The performance of state-of-the-art sentiment extraction on news texts is rather poor […]. Two reasons why sentiment analysis performs poorly on news texts […] are (1) the *lack of large-scale gold standard datasets* and (2) the *high context-dependency* of sentiment-inducing phrases. A gold standard is required to train state-of-the-art sentiment extractors using machine learning […]. The other category of sentiment extractors use manually […] or semi-automatically […] created dictionaries of positive and

> negative words to score a sentence's sentiment. However, to our knowledge no sentiment dictionary exists that is specifically designed for news texts, and generic dictionaries tend to perform poorly on such texts." [43, p. 403]

A little later in the text, they once again point out the lack of suitable datasets of news texts:

> "Not much research has focused on improving sentiment analysis when compared to the large number of publications targeting the prime use case of sentiment analysis: product reviews. Currently, no public annotated news dataset for sentiment analysis exists, which is a crucial requirement for driving forward successful, collaborative research on this topic." [43, p. 404]

The situation has slightly improved since then (see Appendix B). In addition to the four publications on sentiment-annotated news corpora [42, 58, 59, 62], five more have been published since 2019 [55–57, 61, 63].

Regarding news-specific lexicons, the situation is not as dire as Hamborg *et al.* suggest, especially when considering that general news texts primarily revolve around politics. A few political sentiment lexicons do exist. For instance, Young and Soroka created the Lexicoder Sentiment Dictionary for English political texts as early as 2012 [64]. A French version was introduced by Duval and Pétry in 2016 [65]. In 2018, Rauh published a sentiment dictionary for German political language [66, 67]. Further political sentiment dictionaries have been added since 2019. For example, Haselmayer and Jenny publicly released their German Political Sentiment Dictionary in 2020 [68, 69] , and Ring *et al.* created a Hungarian sentiment dictionary for political texts in 2024 [61].

As shown in Appendix A, lexicon-based approaches that benefit from such domain-specific lexicons are still frequently used for sentiment analysis. However, machine learning methods such as Support Vector Machines (SVMs) [60, 70], Bidirectional Long Short-Term Memory (LSTMs) [71, 72], Naïve Bayes Multinomial [NBM] classifiers [59] and various language- and domain-specific BERT versions [55, 61, 71–73], which in various studies have been among the techniques that have achieved the best classification results compared to other methods, do not rely on these lexicons. In contrast, sentiment-annotated domain-specific datasets are crucial as they serve both as training data for model development and as gold standards for evaluating sentiment analysis models.

1.2.5 Research Gap

Přibáň *et al.* [74] show that LLMs are indeed able to recognize sentiment across language boundaries without the need for specific training for each language. However, there is still a need for further research, especially regarding the application of these models to specific text

genres such as news texts. Despite all advances, there remains a research gap in the application of LLMs, especially for multilingual sentiment analysis of news texts, as evidenced by the absence of LLM applications in the overview in Appendix A. Most existing studies focus on individual languages or specific domains, such as product reviews. There is a lack of research evaluating the performance of LLMs in sentiment analysis across multiple languages and in relation to news texts.

1.3 Research Questions

In light of the identified research gap, this master thesis poses the question of whether, and to what extent, large language models are suitable for conducting multilingual sentiment analysis of news texts without specific training of an existing model with news texts in different languages, achieving a level of quality and performance comparable to previously labor-intensive fine-tuned systems. The sub-questions arising from this research question, the methods applied, and the corresponding types of expected results are outlined in Table 1:

Research question	Method	Type of expected result
RQ 1: Which publicly available multilingual sentiment-annotated news datasets can be used to research multilingual sentiment analysis of news texts with an LLM?	• Data science process: data research and collection, exploration and preprocessing	• Collection of news texts, preprocessed for sentiment analysis using an LLM
RQ 2: How can an LLM be used for automatic multilingual sentiment analysis of news texts without fine-tuning?	• Selection of an LLM to research. • Evolutionary Prototyping: iterative development and implementation of a sentiment analysis workflow using the selected LLM	• LLM selected • Working sentiment analysis workflow prototype

Research question	Method	Type of expected result
RQ 3: What are the monolingual and cross-lingual performances of the developed sentiment analysis of news texts?	• Computation of performance metrics for the sentiment analysis output per language	• Evaluation of the results per language
	• Comparison of the results between the languages	• Evaluation of the overall performance

Table 1: Research questions, research methods and expected results.

The methods applied to answer the research questions are described in Section 2. Section 3 presents the results, and in Section 4, these results are discussed and evaluated.

2 Methods

This section outlines the overall methodological approach and the specific methods employed to address the key challenges of this thesis: evaluating the suitability of LLMs for multilingual sentiment analysis of news texts without specific training on multilingual news corpora.

The overall methodological approach was guided by the Cross-Industry Standard Process for Data Mining (CRISP-DM) [75], an established framework for data-driven projects. It defines six phases – business understanding, data understanding, data preparation, modeling, evaluation, and deployment –, which provided a structured and iterative process for addressing the research questions.

While the phases generally follow a sequential order, they also allow for returning to an earlier phase at any time depending on the outcome of individual phases. Chapman illustrates this lifecycle with typical optional iterations back to earlier phases in a diagram like the one shown in Figure 1 [75, p. 13].

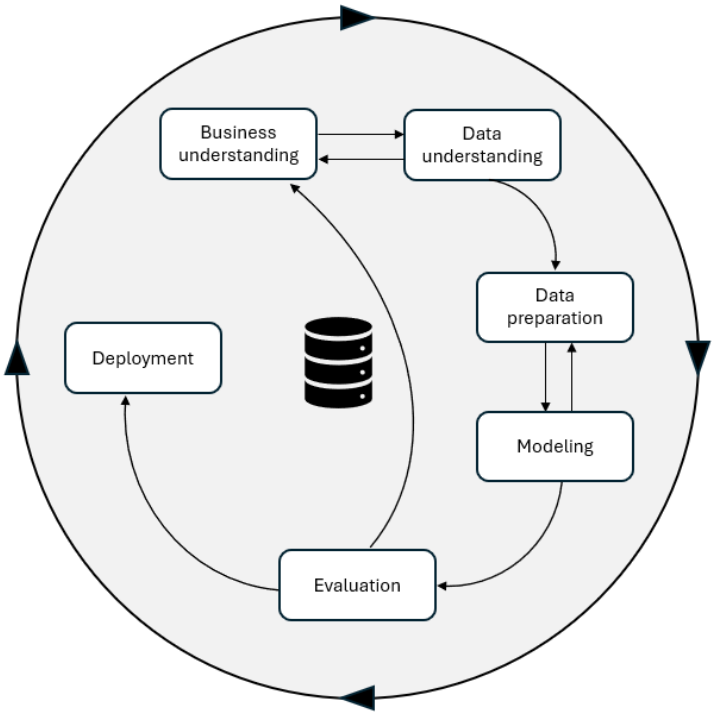


Figure 1: Phases of the CRISP-DM lifecycle (Reproduction of the diagram in Chapman [75, p. 13])

Applying this methodological approach to the master project required structuring the work in three stages, corresponding to the research questions (RQs):

- **RQ 1:** The first stage focused on identifying suitable publicly available multilingual sentiment-annotated collections of news texts as a basis for the study. This involved defining criteria for the selection of text collections and developing an approach for their collection, exploration and preprocessing. These tasks are addressed within the business understanding, data understanding, and data preparation phases of the CRISP-DM lifecycle.

- **RQ 2:** The second stage dealt with selecting an appropriate LLM and developing a workflow for automated sentiment analysis. The selected model needed to possess generative capabilities, support multiple languages, and provide an interface for automated querying. These tasks were primarily performed within the modeling phase, which involved iterative prototyping and workflow development, and were supported by periodic evaluations of intermediate results.

- **RQ 3:** The final stage involved evaluating the results, measuring, and comparing the monolingual and cross-lingual performance of the developed sentiment analysis workflow to assess its overall effectiveness. These evaluation activities align with the evaluation phase of CRISP-DM and contributed to the project's conclusion during the deployment phase.

The phases of the CRISP-DM lifecycle cannot all be directly assigned to the different research questions, as some of them contributed to various parts of the research questions. For instance, the business understanding phase played a fundamental role in all three research questions because it established the requirements for successfully answering the research questions. The data understanding phase and the data preparation phase were used to address RQ 1. The modeling phase involved developing both the programmatic data preparation and the iterative development of sentiment analysis steps and their evaluation and assessment (RQ 1 to RQ 3). The evaluation phase was required for the final assessment of the monolingual and multilingual results achieved in the project in response to RQ 3, but was also relevant for the evaluation of intermediate results when working on RQ 2. Based on this intermediate evaluation, the methods for generating queries directed at the selected LLM were optimized, improving their performance. This is where the relevance of the iterations back to earlier phases, as outlined in the CRISP-DM model, became most evident. Results obtained in the modeling phase were evaluated in the evaluation phase, and based on the evaluations and conclusions drawn, new modeling requirements were made, which led to a change in the implementation of the sentiment analysis workflow, the results of which were then evaluated again, and so on, until this approach reached its limits with the available resources and no further improvement could be expected. Finally, the deployment phase encompassed the provision of data, program code, project documentation, and this thesis, thereby concluding the practical investigation of all three research questions. The assignment of CRISP-DM phases to the research questions is shown in Table 2:

Research question	CRISP-DM phase
RQ 1: Which publicly available multilingual sentiment-annotated news datasets can be used to research multilingual sentiment analysis of news texts with an LLM?	• **Business understanding:** Definition of requirements and goals for the data search and selection of a data collection. • **Data understanding:** Collection and exploration of datasets. • **Data preparation:** Pre-processing of datasets for use in sentiment analysis. • **Modeling:** Development of program functions for the exploration and preprocessing of datasets • **Deployment:** Provision of the datasets or information where and how the original datasets can be found; provision of the developed program functions.

RQ 2: How can an LLM be used for automatic multilingual sentiment analysis of news texts without fine-tuning?	• **Business Understanding:** Definition of requirements for the LLM and its selection; definition of requirements for the workflow development. • **Modeling:** Iterative development and implementation of a workflow for sentiment analysis; development of program functions for intermediate evaluations. • **Evaluation:** Validation of the workflow's functionality and assessment of intermediate results as a basis for iterative optimization. • **Deployment:** Provision of the code and instructions for use.
RQ 3: What are the monolingual and cross-lingual performances of the developed sentiment analysis of news texts?	• **Business Understanding:** Definition of requirements for the evaluation. • **Modeling:** Development of program functions for the monolingual and multilingual evaluation of results. • **Evaluation:** Assessment and interpretation of the results. • **Deployment:** Provision of the results and their evaluation.

Table 2: Mapping of CRISP-DM phases to research questions

The following sections describe how the six phases of CRISP-DM were applied to answer the research questions, and which specific methods were used to execute the corresponding tasks.

2.1 Business Understanding

The first phase of the CRISP-DM process, "business understanding", involves the following tasks as outlined by Chapman [75, pp. 16-19]:

- Identification of business objectives and success factors
- Assessment of the initial situation: gathering project requirements, finding and selecting necessary resources, identifying potential risks and previsible costs
- Definition of goals and success criteria for the project
- Creation of a project plan

These tasks were adapted to the specific requirements of the master project, as described in the following subsections.

2.1.1 Identification of Business Objectives and Success Factors

The business and research interests of the company of the technical supervisor of this project (hereafter referred to as the "client" for simplicity), as well as the success factors determining the effective implementation of these interests, were examined to define and narrow down the subject and objectives of the master thesis and the corresponding project in the light of the current state of research and to plan the project. The possibilities and limitations of a master

thesis (expected level of research sophistication, time frame, personnel limitation to a single person) and the technical and financial resources available to the author to carry out the project also had to be taken into account.

2.1.2 Gathering Project Requirements

The project requirements primarily comprised the requirements for the data collection and the LLM to be used for the planned study. Selection criteria were developed for both resources based on the developed project objectives and the given constraints. Additionally, requirements for modeling, implementing the sentiment analysis workflow and evaluating the results were defined.

2.1.2.1 Selection Criteria for the Data Collection

A multilingual sentiment-annotated news text corpus was required to enable the author to conduct the intended multilingual sentiment analysis and to evaluate the results and performance of the analyses across different languages. The data needed to meet several requirements: Thematically, the data had to align with the domain predetermined by the topic of this master's thesis, i.e., it had to consist of news texts. Given the multilingual approach of the thesis and the intended statistical evaluation (see RQ 3), the data had to cover multiple languages in comparable sample sizes and they had to be suitable to serve as a basis for the formulation of sentiment queries to an LLM. The annotations needed to be of reliable quality to serve as a “gold standard” for measuring the accuracy of the LLM's sentiment predictions. Accordingly, the following criteria for selecting a publicly available data collection were defined:

- **News Texts:** In line with the objectives of this study, the data collection had to consist solely of general journalistic news texts to allow for sentiment analysis of this text type. Social media posts and other informal text types were not considered, as their linguistic style and structure differ significantly from news texts.

- **Multilingualism:** The data collection had to include news texts in at least three different languages to test and compare the performance of the large language model across different languages. In case no multilingual corpora could be found, the possibility of assembling various monolingual datasets and comparing them with each other was considered.

- **Sentiment Annotation:** The texts had to be at least partially annotated with sentiment labels to assess the accuracy of sentiment analysis. The type of sentiment annotation was not strictly defined to avoid unnecessarily limiting the data collections considered. The labels could indicate the polarity of the sentiments (positive, neutral, negative), their intensity, or the type of sentiment, provided the annotation type was the same across multiple languages.

- **Consistent Annotation Levels:** To ensure comparability between results across languages, the levels of analysis (e.g. document, paragraph, sentence, heading, target) had to be consistent across languages.

- **Quality of Annotation:** To ensure that the annotated samples could be considered a so-called “gold standard” against which analysis results could be verified, the correctness of the labels had to be ensured according to generally accepted best practices. Therefore, the samples could not be machine-annotated and had to be assessed by at least three people per sample to minimize annotation errors and increase reliability.

- **Availability of Text Samples:** The texts of the samples had to be included in the corpus or be freely available or accessible free of charge to allow for the verification of the existing annotations and to be able to carry out the intended analyses of the samples.

- **Sufficient Number of Samples:** To achieve statistically significant results, the collections had to contain a sufficient number of samples per language. As a minimum, several hundred samples per language were required in order to be able to carry out several tests with the LLM under investigation, each with 100 samples. Larger datasets with around 5,000 samples per language were preferred with a view to the usability of the data collections in later work, in which machine learning methods could be applied to the selected datasets in order to check whether these methods can improve the performance of the sentiment analysis achieved in this work, the higher number of samples taking into account that machine learning methods need sufficient numbers of test, training and validation data.

- **Comparability of Datasets:** To obtain comparable data for different languages, the number of samples across languages had to be as balanced as possible, and the nature of the data had to be consistent (e.g., approximately equal average sample length, same subject areas if the datasets were thematically restricted, headlines vs. continuous text).

- **Text Preprocessing:** To serve as input for the intended sentiment analysis with a large generative language model, the samples needed to be cleaned of any non-textual elements, such as line breaks or HTML tags, or at least be easily cleanable. Further preprocessing, such as tokenization, lemmatization, removal of numbers and stop words, conversion to word embeddings, etc., was an exclusion criterion if the original text was not available, as the language model would be provided with natural language text, not individual words or vectors.

- **Language Proficiency:** To work effectively with the samples, for example, to know which characters belong to the alphabet of the respective language and which characters might be superfluous or junk data, and to understand the content and tone of the samples and assess the correctness of annotations, the languages used had to match, as far as possible,

the language skills of the author of this thesis. The author's native language is German. She has very good knowledge of English, French, and Spanish, and understands some Portuguese, Italian, and Dutch.

- **Clear Target Definition:** For target-dependent or aspect-based sentiment analysis, the target or aspect of the object to which the sentiment refers had to be clearly and textually identifiable so that it could be included in the prompt instruction the large language model being studied would receive to know what the sentiment analysis should focus on.

2.1.2.2 Selection Criteria for the LLM

The criteria for selecting the LLM to be used in the master project were initially based on the topic of the master thesis and the intended use in this study. These criteria were:

- **Generative capabilities:** The model had to be able to process input in natural language and generate appropriate responses.

- **Language support:** The languages of the selected data collection had to be supported by the model.

- **Performance:** The model had to be trained with a sufficient number of training parameters to ensure it was capable of handling the complexity and nuances of the natural languages it was trained on and, therefore, able to handle the challenge of detecting sentiments in texts. Consequently, it should perform well in einschlägigen Benchmarks testing the actual perfomance of LLM's.

- **Interface:** The model had to offer an interface that supports automated queries.

- **Accessibility:** The model had to be usable for testing purposes with technical and financial resources available to the author of this thesis.

- **Security:** The model should comply with the security requirements of the client.

- **License:** The model should be usable for the client's intended purposes.

2.1.2.3 Data Selection Process

Having defined the client's special business objectives and success factors, the criteria for the selection of a data collection were refined. Based on the three most important of the criteria – multilingualism, the presence of sentiment annotations, and the news text domain – the search for available data collections was then conducted in sentiment analysis literature through

academic libraries, Semantic Scholar[1] and Google Scholar[2], as well as online platforms like ScienceDirect[3], IEEE Xplore[4] and Springer Link[5]. Additionally, repositories such as GitHub[6], Kaggle[7] and Hugging Face[8] were explored. Any corpora that appeared to be potential candidates were then closely reviewed to assess their suitability for the intended use, allowing for a final comparative decision to be made among these data collections.

2.1.2.4 LLM Selection Process

The above criteria for the selection of an LLM were refined and supplemented based on the client's business objectives, success factors and constraints and the selected data collection These criteria were then prioritized according to the client's objectives.

During the search for LLMs to consider, it quickly became evident that the market for LLMs was highly dynamic, with new language models, versions, or variants constantly emerging, while others were losing relevance. To ensure the search remained efficient in this evolving landscape, efforts focused on finding the most up-to-date overviews of the best generative language models. Two overviews were identified [76, 77], one of which specifically focused on generative language models [76]. Since both overviews were quite similar regarding the generative models mentioned, the more comprehensive, detailed, and targeted overview [76] was chosen as the basis for selection. The models were selected from this list.

The selection process was conducted iteratively. After applying the most critical criteria, derived from the prioritization of all criteria, only two LLMs remained. These two models were then compared in greater detail to identify the one best suited for the purposes of this study.

2.1.2.5 Requirements for the Modeling

The master project required a variety of modeling tasks, including programmatic data preparation, the integration of a generative LLM, the creation and application of prompts as well as the evaluation and visualization of the results. To implement these tasks efficiently, the following technical and methodological requiremenets for the modeling were defined to ensure flexibility, scalability, and resource efficiency:

---

[1] https://www.semanticscholar.org/

[2] https://scholar.google.com/

[3] https://www.sciencedirect.com/

[4] https://ieeexplore.ieee.org

[5] https://link.springer.com/

[6] https://github.com/

[7] https://www.kaggle.com/

[8] https://huggingface.co/

- **Modularity and extensibility:** The program architecture should allow individual components to be designed independently and easily extended with additional components and functions. It should support the integration and uniform handling of different components of the same type (e.g., various datasets, different LLMs), enabling resource substitution, comparison of similar resources, or the merging of multiple resources into a single one.

- **Support for data processing**: It should be possible to integrate libraries and data structures that allow efficient handling of large datasets and provide typical methods for data analysis, cleaning and preprocessing tasks. Various input and output formats (e.g., CSV, JSON) should be supported to ensure extensibility.

- **Support for the integration of generative LLMs:** There should be libraries that facilitate the integration of generative LLMs.

- **NLP support:** Libraries capable of handling common NLP tasks, such as sentence or word tokenization, should be available.

- **Support for statistical measurement, evaluation and visualization:** Libraries should be available for applying standard metrics, conducting statistical analyses and visualizing results.

These modeling requirements formed the basis for selecting the programming language, libraries, tools, and design patterns used throughout the project, as described in Section 3.1.4.

2.1.2.6 Requirements for the Implementation of the Sentiment Analysis Workflow

During the business understanding phase, specific requirements for the planned sentiment analysis workflow were defined. Initial requirements were formulated based on the topic of the thesis, the defined research questions, and the expected types of results for each research question:

- **Functionality**: The program had to enable multilingual sentiment analysis by interfacing with a generative large language model (LLM).

- **Extensibility**: It must support the integration of additional languages, datasets and LLM's without requiring extensive manual configuration.

- **Usability**: The program should provide an interface for automated querying and efficient handling of large datasets.

- **Constraints**: The program must operate within the limits of available computational and financial resources, avoiding dependency on expensive infrastructure.

- **Data management:** Input data should be prepared to meet the LLM's requirements, and the results should be provided in a standardized format.

- **Evaluation:** The workflow should enable both quantitative and qualitative evaluation of sentiment analysis results, including the calculation of relevant metrics.

These general requirements needed to be refined and expanded throughout the project. In particular, the business and project objectives, success factors and limitations of the project and the possibilities and requirements of the chosen dataset and LLM had to be considered in the requirement definitions. Specific requirements for the workflow could only be defined after identifying these goals, factors, constraints and the selection of the dataset and LLM, as described in Section 3.1.4.4.

Additionally, the requirements for the workflow were continuously adapted based on new insights gained during development, reflecting the iterative nature of the CRISP-DM lifecycle and the prototyping approach planned for the development of the sentiment analysis workflow.

2.1.2.7 Requirements for the evaluation of the results

The requirements for evaluating the results were based on the insights to be gained and the corresponding standard methods and metrics to be used, as well as any specifically developed evaluation methods for special tasks. Since sentiment analysis is a classic classification task, the use of metrics such as accuracy, recall, precision, and F1-score was essential to assess the correctness and reliability of the sentiment analyses and to compare results across different prompts, developed prototypes and languages. Identified challenges included dealing with multi-class classification as opposed to simple binary classification and addressing imbalanced class distributions. Furthermore, creating an automatic ranking of the results was considered a challenge, especially if no single metric was deemed decisive, but rather the significance of all relevant metrics needed to be considered, requiring a programmatic evaluation approach.

To gain insights into optimizing prompt engineering, additional methods were considered necessary. Understanding which components of a prompt contributed to good or poor performance required correlating the use of these components with the prompts' results. A particular challenge was to identify problematic or beneficial elements in prompts composed of various subcomponents to use these insights for the targeted optimization of the prompts.

2.2 Data Understanding

According to Chapman, the data understanding phase of the CRISP-DM process includes the following tasks [75, pp. 20-22]:

- Collection of the raw data
- Description of the data
- Examination of the data
- Checking the data quality

In this project, RQ 1 (see Section 1.3) comprised the data selection – which was part of the business understanding phase – the data understanding and the data preparation phase. The data understanding phase was understood as the collection of raw data and divided into the following steps:

- **Download of the selected data collection from the source**
  Depending on the provider of the source (e.g. GitHub or Hugging Face), the data collection may be made available in different ways and require different manual or programmatic methods to obtain the data collection.

- **Decompression of the data, if necessary**
  Data collections are often provided in compressed formats and need to be decompressed for processing after download.

- **Identification of relevant files**
  The authors of data collections typically include additional files containing usage instructions or supplementary data, which may not be necessary. The files required for the planned use therefore must be identified.

- **Storage of the original data to be used in the project directory**
  The identified original data that is intended to be used is saved in the project so that it can be accessed at any time without having to carry out the above steps again.

These steps include examining and describing the data and checking its completeness at file structure level.

2.3 Data Preparation

The data preparation phase of the CRISP-DM process includes the following tasks [75, 23–26]:

- Selection of data from the raw data
- Cleaning the data to achieve the desired quality
- Construction of derived attributes, transformation of data, creation of new records
- Merging data from multiple datasets
- Formatting data for the intended use

To accomplish these tasks, associated with the preprocessing of the data collection targeted by RQ 1 (see Section 1.3), the following steps were defined and carried out if they were applicable for the data collection that was finally chosen for the project:

- **Analysis of the data structure**
  Analyzing how the data is split across directories and files, as well as the internal structure of files, to determine the necessity of subsequent steps.

- **Merging training, test, and validation subsets into one dataset per language, if applicable**
  As this project is not a typical data science project involving model development, testing, and validation, the division of data into training, test and validation subsets is unnecessary. Any existing splits can therefore be merged into one single dataset.

- **Conversion of data into a format suitable for processing in the project**
  Depending on the programming language used and the libraries available, specific data formats may be particularly suited. For Python, for example, it seems a good idea to convert the data into a pandas DataFrame format to leverage its efficient processing capabilities.

- **Removal of redundant data, if applicable**
  Unnecessary rows and features should be removed to improve clarity and efficiency.

- **Analysis of relevant data for completeness, consistency, and correctness**
  Dataset features should be complete and consistent (e.g., no empty fields or mismatched data types) to avoid issues during aggregation or analysis.

- **Addition of calculated data for analysis purposes**
  Calculated metrics like sentence, word, and character counts can help identify dataset errors, statistical outliers and inconsistencies.

- **Cleaning of relevant data, if necessary**
  Incomplete, inconsistent, or incorrect data should be corrected or removed.

- **Conversion of sentiment annotations, if necessary**

To align sentiment labels with the format used for sentiment analysis, either the predictions or the labels can be adapted. Labels can be in string format (e.g., “negative”, “positive”, “neutral”) for human readability or numeric format for statistical operations.

- **Storage of preprocessed data in the selected format within the project directory**
  Preprocessed data is stored in a format suitable for subsequent use, depending on the chosen data format and serialization options. For instance, Python dictionaries are typically stored as JSON files, while pandas DataFrames are saved as CSV or PKL files for larger datasets.

- **Extraction and storage of a balanced data subset**
  To facilitate efficient and intuitive evaluation of sentiment analysis results against predefined labels, all sentiment classes should be represented equally in the dataset. For instance, having equal proportions of positive, negative, and neutral samples allows for simple accuracy-based comparison and ranking. Balancing the dataset is a key preprocessing method used in this project.

2.4 Modeling

The modeling phase of the CRISP-DDM process includes the following tasks [75, pp. 27-29]:

- Selection of modelling technique
- Test design
- Model building
- Model assessment

The modeling phase of the master project followed an evolutionary prototyping approach, aiming to develop a functional sentiment analysis workflow for the selected data collection and LLM through iterative development and refinement. The selection of modeling techniques, test strategies, model creation, and model evaluation followed an iterative cycle in which successive solutions were developed, tested, and optimized. Based on the general and specific requirements outlined in Sections 2.1.2.5 and 3.1.4, the architecture of the program, the programming language, tools, libraries, and key design principles were defined.

To determine the steps that the workflow to be implemented for the retrieval of sentiment classifications from the selected LLM should contain, manual experiments were first carried out with the LLM. Based on the findings, the requirements were defined for an initial workflow, which was subsequently implemented. During the execution of the resulting code, the LLM’s responses were checked for validity and evaluated using metrics defined in the evaluation requirements. These metrics were used to rank the queries, aiming to identify patterns that could indicate which query components may have a positive or negative influence on the

results. As a result, new requirements for the workflow were formulated, and the described cycle of implementation, execution, validation, evaluation, and analysis of results for optimization potential began anew. This process is illustrated in Figure 2:

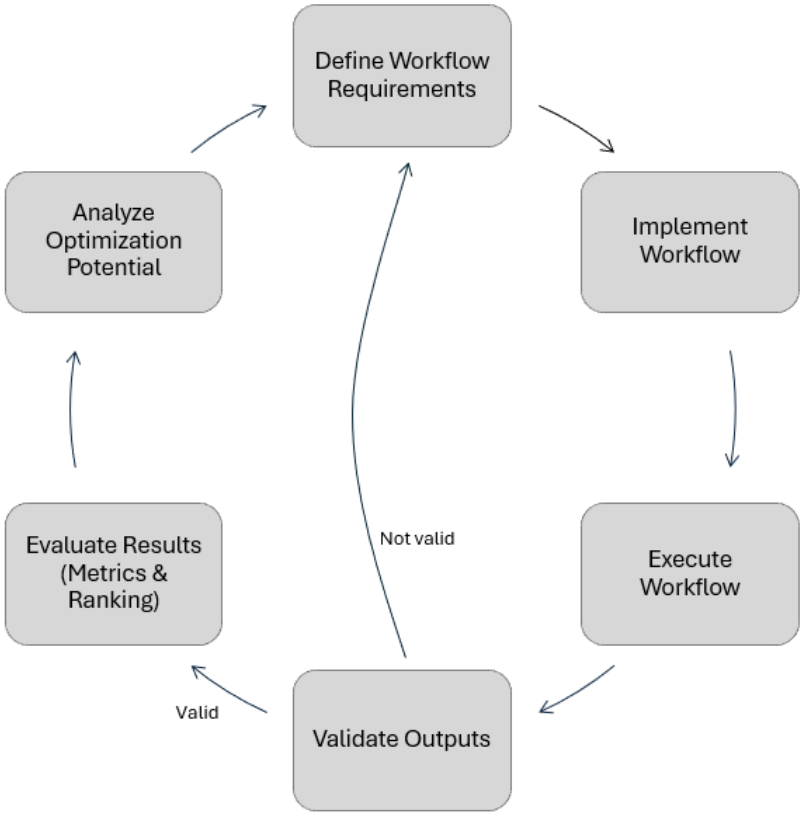


Figure 2: Iterative workflow modeling

The implementation of each cycle, along with its validation and evaluation results, optimization analyses, and the resulting new workflow requirements, are detailed in Section 3.4.

2.5 Evaluation

The evaluation phase of the CRISP-DM process includes the following tasks [75, pp. 30-31]:

- Evaluation of results
- Review of the process
- Determination of next steps

As described in the preceding section, the evolutionary prototyping undertaken in the master project followed a cyclical approach, with each development cycle including an evaluation of the developed prototype. The requirements and challenges associated with these evaluation phases were outlined in Section 2.1.2.7. The specific methods used and their implementation

are presented in Section 3.4.5.3, along with the approaches to developing and evaluating the respective prompt-engineering strategies.

2.6 Deployment

The deployment phase of the CRISP-DDM process includes the following tasks [75, pp. 32-34]:

- Planning of deployment
- Planning of monitoring and maintenance
- Creation of final report
- Project review

In the master project, the deployment phase comprised the concrete generation of project documentation and the provision of datasets, code, documentation and the master thesis. The methods, implementation and results of the project were described in the master thesis. The code documentation was designed to be as user-friendly as possible, both integrated directly in the code itself and also compiled in separate documents so that it can be consulted independently of the code. In addition, various graphics such as class and activity diagrams and other overview diagrams were created and saved in the project itself to visualize critical aspects of the implementation.

Since the specific methods of code documentation depended on the chosen programming language, the integrated development environment (IDE) and the available tools and libraries, the corresponding details are described in Section 3.6.

The created files and documents were ultimately uploaded to a publicly accessible repository managed by the author of this thesis. This repository allows reviewers of the thesis, as well as other interested individuals, to access all materials, provided they adhere to the licenses to which these materials must be subject due to the resources processed in them. Write permissions for the repository remain exclusively with the author to ensure that no materials are modified by third parties. Details about the repository and the licensing are also provided in Section 3.6.

As the project is completed with the present thesis and the deployment of the materials, and the relationship with the client is therefore also terminated, no further monitoring or maintenance concept has been developed for the code and its use. The program is made available without any warranty and is used at the user's own risk. To avoid unwanted side effects, it should only be used by individuals who understand the code, thoroughly review it beforehand, adapt it to their specific needs if necessary and maintain it thereafter.

3 Results

This chapter presents the results of the research conducted within the framework of this thesis. The findings are organized according to the major phases of the CRISP-DM process: business understanding, data preparation, modeling, and evaluation. Each section discusses the outcomes of the corresponding research activities, emphasizing key decisions, challenges encountered, and their resolution. The evaluation of the multilingual sentiment analysis workflow, including the influence of different prompt engineering strategies, is explored in detail. The results obtained from testing the selected LLM on various sentiment-annotated datasets in multiple languages are critically analyzed and compared.

3.1 Business understanding

The first phase of the CRISP-DM process focused on defining the goals and success criteria of the project based on the client's needs and the broader context of multilingual sentiment analysis research. This section outlines how business objectives were translated into specific technical and methodological requirements. It describes the identification of relevant datasets, selection criteria for the LLM, and the constraints imposed by available resources. Additionally, key considerations such as multilingual coverage, sentiment annotation standards, and evaluation metrics guided the project's direction. These aspects provided a foundation for aligning the research with both academic and practical goals, ensuring that the developed system met real-world requirements while addressing the identified research gap.

3.1.1 Business Objectives and Success Factors

The client is a commercial company whose business involves collecting global news in various languages and analyzing it, particularly for sentiment, to provide actionable insights to its customers The client is interested in using fast and reliable technologies for these tasks to deliver high-quality services. To protect its business model and sensitive data, the client prioritizes ensuring that the data it uses and extracts remain under its control and do not fall into external hands. For this reason, it is important to the client that any new technology can be fully integrated into its IT infrastructure and utilized without requiring data to be sent to external providers. The client possesses both the necessary technical equipment and a high level of technical expertise for this type of implementation.

Given the rapid advancements in generative large language models (LLMs) as discussed in the introduction, it seems plausible that these technologies could be used for multilingual sentiment analysis. Before adopting these technologies for business purposes, though, the client wants to ensure that they function as desired in terms of speed and reliability. This study aims to help determine whether LLMs could possibly be employed for multilingual sentiment analysis without requiring additional fine-tuning.

3.1.2 Data Selection

Having established that the client is more interested in general political news than in financial and economic news, financial news was excluded from consideration. The focus shifted to identifying datasets containing general or political news.

Using the methods outlined in Section 2.1.2.3, only five publicly available multilingual, sentiment-annotated news corpora were found. These are described in the following subsections and assessed for their suitability as a basis for the intended study.

3.1.2.1 GDELT (Global Database of Events, Language, and Tone)

The open GDELT project [78], sponsored by Google Jigsaw, was founded in 1979 and now collects global news from various sources such as journalistic news outlets, blogs and social media and analyzes them in near real time, especially with regard to events, actors, locations and sentiments. The GDELT database is updated every 15 minutes [79]. It can be queried, analyzed, visualized and exported via Google BigQuery [45] or cloud-based services [80]. The raw data is also available for download in CSV format. GDELT captures news in over 100 languages from almost every country in the world and from almost every region in those countries [79]. For this purpose, 98.4% of non-English news content is first machine-translated into English. Fifteen languages (Arabic, Basque, Chinese, German, French, Galician, Hindi, Indonesian, Catalan, Korean, Pashto, Portuguese, Russian, Spanish, Urdu) are processed in their original form. With a total of 24 sentiment dictionaries like SentiWordNet [16] and SentiWords [81], over 2,300 sentiment categories are captured.

The GDELT database is divided into several tables [82]: The “event” table contains all events identified in the news texts, along with information about the actors involved and details on the number of sources mentioning the event within the first 15 minutes of its initial mention, as well as the average sentiment of the event during that time, rated on a scale from -100 (extremely negative) to +100 (extremely positive).

The “mentions” table contains all news articles that mention the event, including details about the respective news text such as its source, the language in which it was published, any translation method used if applicable, and the average sentiment score related to the event in the text [82].

The “knowledge graph” links additional information about the event [78]. For sentiment analysis, the sentiment dictionaries are scanned to record how many of the words in each news text correspond to various sentiment dimensions in each of the sentiment dictionaries that contain matches, and, if possible, what average value was calculated.

The GDELT database claims to be and most probably actually is the largest sentiment analysis collection of news in the world [78] and is a valuable tool for conducting comprehensive sentiment analyses covering various regions and languages on global events. However, the dataset is only partially suitable for the investigation in this study. First, the events for which sentiments are analyzed are automatically identified, meaning that their identification may not always be accurate. GDELT itself addresses this uncertainty by providing a confidence score between 10% and 100% for the attribution of a mentioned event to a given generally identified event. This confidence score could be used to eliminate likely incorrect attributions from the investigation and focus only on events that can be confidently assigned.

Second, the sentiment analyses are conducted automatically. Their correctness is not guaranteed by anything and cannot be easily assessed. Nevertheless, the scale from -100 to +100 could be used to compare the assessment of sentiment in individual texts on given events. However, since the analysis results of GDELT cannot be regarded as a gold standard, only differences in the sentiment ratings between the analysis systems examined (GDELT and an LLM to be examined) could be worked out here. Which system leads to the better results would have to be determined by comparison with a human assessment of the news texts. Apart from the effort involved in manual notation by several people for a sufficiently large collection of news texts, it would also have to be determined how usual sentiment assessments can be mapped to such a fine-grained scale of -100 to +100 or, conversely, how the scale can be reduced to a scale of human assessment, or how the question to the annotators would have to be formulated and evaluated in order to obtain a result on the specified scale.

Third, the original news texts are not stored in the database, but only a reference to the respective source [82]. Among the types of sources used, the “WEB” source type appears to be the most likely to guarantee public and uncomplicated access to the source. For this source type, a qualified URL is specified that can be accessed. Provided that the referenced sources are freely accessible and have not been deleted or moved to another address in the meantime, the original text can be extracted from the referenced website and prepared for the sentiment analysis. This requires the news text to be reprocessed (web scraping), which involves considerable effort, as the HTML structure of different sources varies, requiring additional handling to separate the actual news content from other elements like comments, links, or advertisements. The challenges of web scraping are described, for example, by Mello, Cheema and Thakkar [56, p. 135]. One potential approach could be to provide the large language model with just the source URL, allowing the model to determine the relevant content from the webpage without manual preprocessing. However, this approach assumes the model can read specified webpages, which is not always the case. Leaving this task to the model also poses the risk of losing control over which input the model processes, leading to unexpected results.

Finally, identifying the specific event to which the sentiment analysis of a news article should refer presents a challenge. In the GDELT “event” table, the event is identified by a global EventID and is characterized by the actors involved and various hierarchical action codes that describe the relationship between the actors (e.g., “Appeal” > “Appeal to yield” > “Appeal for easing of administrative sanctions”) [82]. It remains unclear how these abstract labels can be used to define the event in such a way that the large language model can be prompted to analyze the sentiment in the news text related to the event.

Since the sentiment annotations of the GDELT corpus cannot be used as a gold standard, the original texts are difficult to obtain and no suitable labels of the targets to which the sentiments should refer can be extracted from the GDELT tables, the GDELT collection was excluded as unsuitable for the intended study. The suitability analysis of the GDELT collection and the datasets presented below is summarized in Table 5 in Section 3.1.2.5.

3.1.2.2 MMS (Massively Multilingual Corpus of Sentiment Datasets)

The Massively Multilingual Corpus of Sentiment Datasets (MMS) is a collection of 79 datasets in 27 languages, selected, unified, and prepared by Augustyniak *et al.* in 2023 [83] from a total of 350 sentiment datasets according to defined quality criteria. The selection criteria were as follows [83, p. 4]:

- A minimum of three sentiment classification levels (exclusion of binary sentiment classification)
- Exclusion of machine sentiment classification
- Exclusion of sentiment classification based on emojis
- Detailed documentation of the annotation process

The datasets were processed as follows [83, p. 4]:

- Duplicate entries were merged, and inconsistencies were corrected.
- Multilingual collections were split into monolingual collections.
- Classifications with more than three levels (especially 5-level Likert [84] scales) were reduced to three levels.
- Classifications of emotional categories (joy, happiness, fear, sadness, disgust, anger) were mapped to sentiment categories (positive, negative, neutral).
- The datasets were assigned to four different domains (social media, review, news, and others).
- Self-confidence scores were calculated for all samples.

Only five of the 79 datasets fall into the “news” category (see overview in Appendix C): en_financial_phrasebank_sentences_75agree, en_per_sent, en_vader_nyt, sl_sentinews and

hr_sentiment_news_document [85]. They cover only three languages (English, Slovenian, and Croatian), are imbalanced in the number of samples (13,971 English vs. 10,417 Slovenian and 2,025 Croatian samples), and differ in their annotation levels (sentence vs. document level). However, three datasets are annotated at the document level: en_per_sent (5,333 samples), sl_sentinews (10,417 samples) and hr_sentiment_news_document (2,025 samples). They cover three languages, appear to have been carefully hand-annotated by multiple individuals according to their annotation documentation, and could therefore serve as a basis for the intended investigation in this thesis. However, the number of samples remains imbalanced, and the average text lengths in the three languages differ significantly.

For example, when comparing the English, Slovenian, and Croatian versions of the European General Data Protection Regulation (GDPR) [86–88], it becomes apparent that the same text typically has different word and character counts depending on the language (see Table 3). This can be explained primarily by structural differences between the languages. Morphologically rich languages like Croatian and Slovenian [89] express much through morphological mechanisms, requiring fewer words and characters to convey the same meaning as less morphologically complex languages. Due to their close relationship [90], the Croatian and Slovenian versions are nearly the same length, while the English version contains 8–10% more characters and even 15–17% more words.

Language	Character Count	Word Count	Ratio Between the Languages			
			Character Count		Word Count	
EN	349,468	54,866	EN/HR:	1.10	EN/HR:	1.17
			EN/SL:	1.08	EN/SL:	1.15
HR	316,725	46,802	HR/EN:	0.91	HR/EN:	0.85
			HR/SL:	0.98	HR/SL:	0.98
SL	322,335	47,624	SL/EN:	0.92	SL/EN:	0.87
			SL/HR:	1.02	SL/HR:	1.02

Table 3: Character and word counts of the EN, HR, and SL versions of the GDPR [86–88]. The HTML texts were copied and pasted into MS Word documents, and the counts were obtained using MS Word's counting function.

In the three MMS datasets considered - en_per_sent, hr_sentiment_news_document, and sl_sentinews - the text lengths of the Croatian samples differ significantly from those of the other two languages. While the English and Slovenian samples show approximately the same ratio in length as their respective language versions of the GDPR, and can therefore be considered comparable, the Croatian texts are only about half as long as those in the other two languages (see Table 4). Since it can be assumed that this difference in text length would affect the accuracy of sentiment analysis using an LLM, and the number of Croatian samples

is only about half that of the English samples and one-fifth that of the Slovenian samples, the Croatian dataset would need to be excluded from the investigation or included only with reservations.

Language	Average Character Count	Average Word Count	Ratio Between the Languages	
			Character Count	Word Count
EN	2130	357	EN/HR: 2.08 EN/SL: 1.06	EN/HR: 2.18 EN/SL: 1.14
HR	1022	164	HR/EN: 0.48 HR/SL: 0.51	HR/EN: 0.46 HR/SL: 0.53
SL	2017	312	SL/EN: 0.95 SL/HR: 1.97	SL/EN: 0.87 SL/HR: 1.90

Table 4: Comparable MMS news datasets: Average character and word counts (rounded).

The three datasets would also pose further challenges if they were to serve as the basis for the intended investigation in this thesis. At least the English and Croatian datasets would require special preprocessing, as they contain at least line break characters ("\n"). These and any other unwanted special characters would need to be found and removed. More critically, however, the author of this thesis does not speak two of the three languages, Croatian and Slovenian, even minimally, and therefore cannot assess the correctness of the annotations or the need for further text preprocessing in these datasets.

In addition to the news datasets compiled and prepared by Augustyniak *et al.* for the MMS corpus, the original datasets were also examined for their suitability for the current project. These include three English datasets: Financial_phrasebank [91], PerSenT [92], und NYT [15], as well as the Slovenian SentiNews 1.0 dataset [59] and the Sentiment Annotated Dataset of Croatian News [55]. The Slovenian and Croatian datasets were annotated according to the same well-documented principles. They have labels at the sentence, paragraph, and document levels and could therefore be compared with the two English datasets, *NYT* and *Financial_phrasebank*, which are annotated at the sentence level.

However, Financial_phrasebank focuses on financial news, making it highly specialized, raising the question of whether the more general Slovenian and Croatian datasets are comparable. This question was ultimately answered in the negative, as sentiments in Financial_phrasebank were only correctly annotated when they related to economic or financial topics, while other sentiments were labeled "neutral" regardless of their actual tone. Therefore, Financial_phrasebank was excluded as a candidate for this study in both its original and MMS versions.

Regarding the NYT dataset, it is originally annotated with an intensity scale ranging from -4 to 4, rather than with a three-point polarity. A possible approach would be to compare the original sentence-level annotated Slovenian and Croatian datasets with the NYT dataset prepared for the MMS corpus, en_vader_nyt, utilizing the fact that Augustyniak *et al.* converted the original intensity scale into a three-point polarity scale. However, the original Slovenian dataset is annotated on a five-point polarity scale and would need to be mapped to three points to be compared with the other two datasets.

The three sentence-level annotated datasets are even more imbalanced, with 168,899 Slovenian, 25,074 Croatian, and 5,190 English sentences. However, 5,190 samples per language represent a good dataset size for conducting not only the intended LLM-based sentiment analysis but also further experiments using machine learning methods (fine-tuning). For sentence-level analysis, 5,190 selected Croatian and Slovenian sentences could be chosen from the datasets to be compared with an equal number of English sentences.

The only significant remaining problem is the language issue. As with the document-level annotated datasets, the author lacks knowledge of Slovenian and Croatian. For these reasons, the plan to use these datasets for analysis was ultimately abandoned when a better alternative was found, as will be described in Section 3.1.2.4.

3.1.2.3 Olympic News Datasets

In 2023, Mello, Cheema und Thakkar [56] introduced a bilingual corpus of news headlines in English and Portuguese, focusing on the impacts (referred to as “legacy”) of the London 2012 and Rio de Janeiro 2016 Olympic Games. The corpus consists of two different datasets (see overviews in Appendices E and F). In Dataset 1, the sentiments in 352 English and 365 Portuguese headlines were manually classified by a “domain expert” [56, p. 138] using a three-point polarity scale (positive, neutral, negative). In Dataset 2, the sentiments in 807 English and 464 Portuguese headlines and news texts were analyzed and annotated using a combination of three machine sentiment classifiers (Vader [15] and two variants of BERT [2] trained on different datasets). The Portuguese texts were first machine-translated into English for analysis in English. However, both datasets contain only full-text headlines, while the corresponding news texts are merely linked and not included themselves. Some URLs lead to paid offerings from the respective news providers.

For the intended investigation in this thesis, only Dataset 1 could be used, as it is the only one annotated by humans. However, only one person was involved in the annotation, which does not meet the usual best practices for sentiment annotation, so the dataset cannot truly be considered a gold standard for comparison with the results of LLM-based sentiment analysis, as intended in this thesis. While the number of samples per language in Dataset 1 is balanced, it is relatively small and would only be suitable for the potential future use of machine learning

methods when combined with other datasets from different sources. Furthermore, the dataset covers only two languages, meaning that additional languages with datasets from other sources would be needed for a truly multilingual analysis. But the Olympic News dataset cannot, for example, be meaningfully combined with the previously described MMS datasets, as the latter are annotated at the document or sentence level rather than at the headline level. For these reasons, the Olympic News datasets were ultimately excluded from further consideration.

3.1.2.4 MAD-TSC (Multilingual Aligned News Dataset for Target-dependent Sentiment Classification)

In 2023, Dufraisse *et al.* [57] published a parallel corpus of news texts in eight languages for target-dependent sentiment analyses, called the Multilingual Aligned News Dataset for Target-dependent Sentiment Classification (MAD-TSC). The collection contains a total of 4,714 unique sentences from 286 news articles from 30 countries, translated by professional translators from their original languages into seven other languages, resulting in coverage of eight languages (German, English, Spanish, French, Italian, Dutch, Portuguese, Romanian). The texts were aligned at the sentence level across languages, so each sentence in one language corresponds to a sentence in the other languages. A total of 5,110 named entities were identified in the 4,714 sentences, which serve as targets for sentiment analysis. A separate sample was created for each named entity in a sentence, resulting in 5,110 samples in each language. All 5,110 samples are labeled with sentiment tags on a three-point scale, based on the judgment of three people per sample. The labels for a given sample are consistent across all languages.

The multilingual parallel corpus from Dufraisse *et al.* is ideal for the intended multilingual sentiment analysis in this thesis and meets all the predefined selection criteria. The news texts or their sentences are included directly in the corpus and can be used in a well-prepared state. Many of the corpus languages are familiar to the author of this thesis, and the less familiar or unfamiliar languages pose no problem since the samples require no further text preprocessing, and their content is reliably known through the corresponding samples in other languages. The translations were performed by professional translators, ensuring maximum reliability, and the labels were assigned according to best practices. The corpus provides an even number of samples per language, in sufficient quantity for all intended purposes, and is particularly well-suited for measuring the cross-lingual performance of a multilingual sentiment analysis system due to its comparability across all languages.

The data were originally created by Presseurop, a multilingual news outlet funded by the European Commission from 2009 to 2013 [93], and later taken over by Voxeurop, a European cooperative society [94]. At the time of writing this thesis, the content produced by Presseurop and Voxeurop is governed by the CC BY-NC-SA 4.0 license [95, 96]. However, Dufraisse *et*

al. claim the data to be under the CC BY-NC-ND license [57, p. 8288, 97], which they also apply to their edition of the corpus. As such, the corpus may be freely used for non-commercial purposes and shared in the original form provided by Dufraisse *et al*.

Given that the corpus is well-suited for the objectives of this study and the legal conditions are satisfied, the challenge of finding a multilingual, sentiment-annotated news text corpus has now been successfully solved.

3.1.2.5 Result

The results of the evaluation of the multilingual corpora described in the preceding Sections are summarized in Table 5. Of the sentiment-annotated news text collections considered, only the MAD-TSC corpus fulfills all the selection criteria described in Section 2.1.2.1. Consequently, it has been chosen as the dataset for the intended study. A particularly advantageous feature of this corpus is that it is a parallel corpus, where all samples were professionally translated from one language into all other included languages. This ensures a high level of comparability in the results of the planned sentiment analysis, presumably free from biases introduced by translation errors. This is especially true for this dataset because the samples were presented to annotators in three languages. If annotators arrived at differing sentiment assessments based on the versions in different languages, likely due to translation inaccuracies, such samples were excluded. Given these quality factors in both the translation and annotation of the samples, it can be assumed in the planned study that differences in the accuracy of the sentiment analysis across languages will be attributable solely to variations in the language-specific capabilities of the LLM under investigation.

	GDELT	**MMS (Document Level)**	**MMS + MMS Originals (Sentence Level)**	**Olympic News Dataset 1**	**MAD-TSC**
News Texts	✓	✓	✓	✓	✓
Multilingualism	✓	✓	✓	(✓)	✓
Sentiment Annotation	✓	✓	✓	✓	✓
Consistent Annotation Levels	✓	✓	✓	✓	✓
Annotation Quality	-	✓	✓	(✓)	✓
Availability of Text Samples	(✓)	✓	✓	✓	✓
Sufficient Number of Samples	✓	✓	✓	(✓)	✓
Comparability of Datasets	✓	(✓)	✓	✓	✓
Text Preprocessing	-	-	-	✓	✓
Language Proficiency	✓	-	-	(✓)	✓
Clear Target Definition	-	✓	✓	✓	✓

Table 5: Suitability analysis of datasets based on selection criteria

3.1.3 LLM Selection

For the selection of an LLM for the master project, the relatively up-to-date overview provided by Alexander Thamm GmbH [76] at the end of 2023 was taken as a starting point. This overview was chosen because all the models it mentioned already met the first selection criterion: the requirement that the model should have generative capabilities: The overview analyzed 14 generative language models, listed alphabetically as follows: Bloom, Claude, Command R+, Dolly 2.0, Falcon, GPT-3.5, GPT-4, Guanaco-65B, LaMDA, LLAMA, Luminous, Orca, PaLM and Vicuna. These LLMs and their characteristics, including bibliographic references and URLs, are compiled in the overview table in Appendix G.

Given the client's interest in ensuring that the selected model could be fully integrated into the company's IT infrastructure and utilized without requiring data to be sent to external providers, the initially defined selection criteria "security" and "licence" were prioritized and further specified: First, the LLM should be open source. Second, it should be usable for commercial purposes. The following sections explain these criteria in more detail and apply them to the list of language models.

3.1.3.1 Open Source vs. Closed Source

LLMs can be categorized according to whether their developers pursue an open-source or a closed-source policy [77, 98]. In the case of open-source language models, the model is available for download and can be installed and used in the user's system. Using the model in this way, users only incur costs for operating their own infrastructure and integrating the LLM into it. The user's input data remains within the user's control, allowing for secure operation with sensitive or confidential data. Users can freely modify the original model according to their individual needs within the license terms, i.e. change the code or fine-tune the model with their own data and parameters to optimize performance. Disadvantages of this type of LLM operation include the high demand for memory and storage resources and computing power, and the need for technical expertise to properly operate and optimize the LLM for the user's needs. The user is also responsible for the maintenance and ensuring the availability of the LLM and must quickly remedy any outages in a manner that suits their needs, which may require high technical competence.

Closed-source language models, on the other hand, remain under the ownership and control of an external manufacturer or operator, who hosts the LLM [77, 98]. This arrangement has the advantage for the users that the provisioning, maintenance, and operation of the LLM are the responsibility of the operator, who, under the right legal agreement, is also liable for the system's availability. The users do not need to possess the corresponding technical resources or know-how. However, they must pay for the provision and use of the model and are dependent on the quality of the maintenance of the services and support from the manufacturer

or operator. Modifications to the model to suit individual needs are only possible within the technical mechanisms provided by the manufacturer or operator, and users are protected from potential price changes or changes to the operator's offerings only within the terms of the closed agreements.

Another critical disadvantage is that this type of use of an LLM can be problematic with sensitive data, especially if the LLM is operated in a country or even on a continent other than that of the end user, where data protection rules and safeguards differ from those of the user. For instance, when transferring sensitive data from a European country to a U.S.-based LLM, there is no guarantee that U.S. authorities will not force access to the data under certain circumstances [99].

Given the client's needs, an open-source approach was deemed appropriate when evaluating open-source versus closed-source LLMs. The client has the necessary resources and expertise, along with a strong interest in sentiment analysis systems that can be integrated into its own infrastructure – without the need to transmit sensitive data to third parties.

However, beyond purely open-source LLMs, other LLM business models could also meet the client's requirements for data security and model customizability while potentially alleviating the company from the technical burden of operating an LLM within its own systems. For instance, an LLM provider might offer a license allowing their model to run on the client's own servers, with all client-used and generated data remaining on-premise. Depending on the licensing model, the installation, customization, and maintenance of the LLM could either be managed by the client – potentially with access to the source code for paying customers to enable customizations – or handled entirely by the provider, who would charge for the associated services.

Whether such business models are currently offered was not investigated in the context of this master project, as this type of LLM deployment would not have been accessible free of charge to the author. The targeted sentiment analysis in this thesis, however, was intended to be carried out without incurring any costs.

Accordingly, only open-source LLMs were considered for this project. The following commercial LLMs were therefore excluded from the list: Claude, Command R+, GPT-3.5, GPT-4, as well as Google's LaMDA and PaLM, which are not, or at least not yet, publicly accessible in Europe. Luminous was also excluded – although its source code is reportedly available to paying customers [67] – because the LLM cannot be used free of charge for the research purposes of this thesis.

The differences between open-source and closed-source policies are recapitulated in Table 6.

Open Source	Closed Source
• No dependence on external service provider (no risk of price increases, service switch-off, model removal, …)	• Dependence on external service provider and their conditions • Switching between providers might not be possible
• Data stays on-premise => sensitive data can be processed	• Data is transferred to external site, may even be transferred to another country or continent (legal and security implications)
• No subscription or data volume costs • Operation costs of the own infrastructure	• Usually, costs according to the data volume processed (up- and down-scaling possible according to the provider‘s conditions)
• Expertise needed for running and maintaining LLM and interfaces • Possible longer-lasting failures if problems occur that are not easily solved	• User needs less technical expertise to use the LLM • Technical support from the owner and/or service provider • User depends on the owner/service provider maintaining the LLM and providing updates • Possible guarantee of the service provider as to the accessibility and availability of the service
• Source code is known and can be adapted to fine-tune the LLM • Community of users helps evaluate and optimize the model and find solutions for problems • Community of users develops libraries enhancing the capabilities of the LLM	• User depends on the owner providing options to customize and fine-tune the LLM

Table 6: Open-source vs. closed-source models [77, 98]

3.1.3.2 Usability for Commercial Purposes

Since the client is a commercial company and wants to use the LLM for business purposes, those open-source LLMs were excluded that may only be used for non-commercial or research purposes, i.e. LLaMA [100] and its further developments Guanaco-65B [101], Orca [102] and Vicuna [103]. Having applied this criterion, three LLMs were left: Bloom, Dolly 2.0, and Falcon.

3.1.3.3 Language Support

Next, the language support criterion was applied. While BLOOM and the Falcon models support multiple languages, Dolly 2.0 was primarily trained on English data. As a result, it did not meet the requirements for the intended multilingual purposes of the master project and was excluded from further consideration.

With Dolly 2.0 excluded, the decision narrowed to BLOOM and the Falcon models. While the Falcon models were trained in only 10 languages, BLOOM's training included 46 natural languages. Nevertheless, BLOOM and Falcon only partially cover the languages in the MAD-TSC news corpus. The Falcon models were apparently well-trained only in English, German, Spanish, and French [104–106]. For the other trained languages (Italian, Portuguese, Polish, Dutch, Romanian, Czech, and Swedish), the developers indicate that the models have only limited language capabilities. In turn, the training corpus used by BLOOM lacks the languages German, Italian, Dutch and Romanian, among others, so that, of the MAD-TSC languages, only English, French, Spanish and Portuguese are covered [107, pp. 7-8]. Since BLOOM as well as the Falcon models cover only four out of eight required languages, a decision could not be made between them based on this criterion.

3.1.3.4 Performance

A comparison of the training parameters between BLOOM and the Falcon models revealed that the largest Falcon model, trained with 180 billion parameters, is approximately comparable to BLOOM, which was trained with 176 billion parameters. Since it was estimated that the smaller Falcon models would perform less effectively in terms of the complexity and nuances required for sentiment analysis compared to the largest Falcon model, they were excluded from further consideration. The decision thus came down to Falcon 180 B and BLOOM.

Since the number of training parameters provides only a rough indication of a model’s potential performance, benchmarks were consulted to evaluate the actual measured performance of the two models in typical text generation tasks. A significant benchmark for language models is the Massive Multitask Language Understanding (MMLU) Benchmark [108], which assesses a language model’s world knowledge. As of early November 2024, the MMLU ranking [109] places Falcon-180B 35th out of 115 language models with an average score of 70.6% [110, p. 34], while Bloom ranks 88th with a score of 39.1% [111, p. 28]. The HellaSwag benchmark [112, 113], which measures the ability of language models to complete sentences, ranks Falcon-180B 21st out of 89 models with an accuracy of 85.9 [110, p. 34] and BLOOM 54th with an accuracy of 73.2 [111, p. 30] (See Table 8 in Section 3.1.3.7). Considering these rankings, Falcon-180B appears to be the more promising model compared to BLOOM.

3.1.3.5 Licenses

After the balance seemed to tip in favor of the Falcon 180B model, the usability of the models still had to be examined on a legal, technical and financial level. First, a closer look was taken at the licensing terms of both models. Although both BLOOM and Falcon-180B are open-source LLMs and can be used for commercial purposes, their use is subject to certain license conditions that are intended to ensure the ethical use of the models. The Falcon model is governed by an “Acceptable Use Policy” (Falcon 180B TII License 1.0) [114], which allows its use only in compliance with all local, national and international laws. It explicitly prohibits using the model or its derivatives to exploit or harm minors or living beings, to generate false information that causes harm to others, or to engage in defamation, disparagement or harassment.

The license terms for BLOOM are more detailed and comprehensive. The BigScience Responsible AI License (RAIL) v1.0 [115] excludes the same abuses as the Falcon License, but also the distribution of personal data that can be used to harm a person, the distribution of content without expressly indicating that it is machine-generated, and the use of the model to impersonate someone else. Furthermore, it is prohibited to use the model or products derived from it for fully automated decision-making if this adversely affects the rights of a person or creates or modifies legally binding obligations. All uses that are intended to discriminate against a person or group or lead to such discrimination are also prohibited. Additionally, the model and products derived from it may not be used for medical purposes or in the administration of justice, criminal prosecution or in immigration or asylum proceedings. Products derived from the model must be accompanied by the BigScience RAIL license and the products must be subject to the same prohibitions as BLOOM itself in an enforceable form. No decision criteria can be derived from the licenses of both models. The purposes pursued in this work fall within the permitted use of both models.

3.1.3.6 Technical and Financial Aspects

Finally, the technical and financial aspects were examined, as they represented the prerequisites for the practical usability of the models within the scope of the master project. These aspects included the criteria originally referred to as “Interface” and “Accessibility.” Specifically, the evaluation examined how and under what conditions a user could access and interact with the models.

Open large language models can be used in various ways. As transformer-based models, they can be downloaded in full as shards and directly used with the Transformers library. However, hosting and running these models on one’s own servers requires significant resources in terms of storage, memory, and graphics processing units (GPUs), which far exceed the capabilities of a typical notebook or desktop computer. In experiments conducted by the author of this thesis with the system available to her (see Table 7), the storage issue was resolved using an

external 1 TB hard drive. However, the available random-access memory (RAM) was not sufficient to work with the downloaded model. Attempts to work on the Intel GPU also failed due to the difficulty of correctly integrating the Intel® Extension for PyTorch provided by Intel [116], which was intended to replace the GPU functionalities usually provided by the Compute United Device Architecture (CUDA) [117] on Nvidia graphics cards, into the existing installation of Python and its libraries.

Category	Details
Operating system	Microsoft Windows 10 Pro, 64 Bit
CPU	11th Gen Intel®Core™ i7-1165G7, 2.80 GHz, 2803 MHz, 4 Cores, 8 logical processors
RAM	32 GB
GPU	Intel® Iris® Xe Graphics
Programming language	Python 3.12.2
IDE	PyCharm 2023.3.4 (Professional Edition)
Test suite	Pytest 8.0.0
Docstring style	Numpy
Documentation tool	Sphinx 7.2.6
Repository	GitHub (https://github.com/DianeKeller/SentimentAnalysis.git)
Interactive development environment	Python 3.12.3 Jupyter Notebook 7.2.1 Anaconda 2.6.3

Table 7: Technical setup

As an alternative to a local LLM installation, cloud services such as Microsoft Azure[9], Amazon Web Services[10] or Google Colab[11] in conjunction with Google Drive[12] offer to provide the

[9] https://azure.microsoft.com/

[10] https://aws.amazon.com/

[11] https://colab.google/

[12] https://drive.google.com/

required resources externally. However, the author’s attempts to run the LLMs using the free offerings from these services, designed for beginners or students, were unsuccessful because the required technology (GPUs) and capacities (storage and memory) require paid subscriptions. Specifically, using GPUs can incur unpredictable costs, as they are not covered under a flat rate but are billed based on usage.

A technically and financially less challenging option for working with the two language models is to use an API provided by Hugging Face for the respective model. With this so-called inference approach, the model does not have to be installed and operated on the user's system or in the cloud but runs on the Hugging Face platform and can be queried via the API. Both the Falcon models and BLOOM can be used in this way. However, as shown in Figure 3, Hugging Face sets a size limit of 10 GB to the free use of models, which is already exceeded by the smallest of the Falcon models considered, Falcon-7B. As a result, a paid inference endpoint [118] must be created to use the Falcon models.

Surprisingly, the 340 GB BLOOM can be freely queried - within certain rate limits [119] – via a serverless inference API [120] using a Hugging Face User Access Token [121] as shown in Figure 3. The language model appears to be given preferential treatment by Hugging Face, likely due to Hugging Face’s significant involvement in its development and to the explicit goal of its development, which was to provide the general public with a powerful open-source large language model that, at the time of its release, should approach the performance of the proprietary large generative language models on the market [107].

Considering the financial barriers to using Falcon-180B and, conversely, the free availability of BLOOM via the serverless inference method, BLOOM emerges as the most suitable large language model for this thesis’s aim to assess the capability of large language models to perform sentiment analysis without specialized fine-tuning.

```
[1]: import os
     import requests

     access_token = os.getenv('HUGGING_FACE_AUTH_TOKEN')

     BLOOM = "https://api-inference.huggingface.co/models/bigscience/bloom"
     Falcon7 = "https://api-inference.huggingface.co/models/tiiuae/falcon-7B"
     Falcon40 = "https://api-inference.huggingface.co/models/tiiuae/falcon-40B"
     Falcon180 = "https://api-inference.huggingface.co/models/tiiuae/falcon-180B"

     headers = {"Authorization": f"Bearer {access_token}"}

     def query(api, payload):
         response = requests.post(api, headers=headers, json=payload)
         return response.json()

[2]: data = query(BLOOM,{"inputs": "Can you please let us know more details about your"})
     data

[2]: [{'generated_text': 'Can you please let us know more details about your issue? We will be happy to help you.'}]

[3]: data = query(Falcon7,{"inputs": "Can you please let us know more details about your"})
     data

[3]: {'error': 'The model tiiuae/falcon-7b is too large to be loaded automatically (14GB > 10GB). Please use Spaces (https://
     huggingface.co/spaces) or Inference Endpoints (https://huggingface.co/inference-endpoints).'}

[5]: data = query(Falcon180,{"inputs": "Can you please let us know more details about your"})
     data

[5]: {'error': 'The model tiiuae/falcon-180B is too large to be loaded automatically (359GB > 10GB). Please use Spaces (http
     s://huggingface.co/spaces) or Inference Endpoints (https://huggingface.co/inference-endpoints).'}
```

Figure 3: BLOOM vs. Falcon: The use of Falcon via the Hugging Face Inference API fails due to the 10GB limit.

3.1.3.7 Result

Among the 14 large language models collected in late 2023, closed-source models as well as models exclusively available for research purposes were excluded. The open-source model Dolly 2.0 was also excluded from the shortlist due to its lack of support for languages other than English. Since only one model can be examined in the context of this thesis, the two remaining open-source models, Falcon-180B and BLOOM, were compared to reach a decision (see Table 8).

It was found that the license terms of both models follow a similar ethical approach, allowing the use of either model within the context of this master thesis. The models are also nearly equivalent in size and in the number of languages relevant to this study; however, the Falcon model performs significantly better than BLOOM in two benchmarks, making it appear more promising.

	Falcon-180B	BLOOM
Parameters	180 billion	> 176 billion
Natural Languages	10	146
MAD-TSC Languages	DE, EN, ES, FR (IT, NL, PT, RO)	EN, ES, FR, PT
MMLU (%)	70.6 [110, p. 34]	39.1 [111, p. 28]
MMLU Rank (out of 115) [109]	35	88
HellaSwag (Accuracy)	85.9 [66, p. 34]	73.2 [111, p. 30]
HellaSwag Rank (out of 89) [113]	21	54
Usage by Inference	User-defined Inference Endpoint	Serverless
Cost	Based on usage	Free within rate limits
License	Falcon 180B TII License 1.0 [114]	BigScience RAIL License v1.0 [115]

Table 8: Falcon-180B vs. BLOOM

In the end, however, BLOOM was chosen, as technical and financial considerations proved decisive: only BLOOM can be used for the purposes of this study largely free of charge and with manageable technical effort through a Serverless Inference API. As BLOOM is used in this thesis to perform a target-dependent sentiment analysis on samples of the MAD-TSC data corpus, this thesis and the program implemented for this analysis are also subject to the BLOOM license, which is attached in Appendix F of this thesis.

3.1.4 Modeling Choices

Based on the findings, requirements and constraints previously described, specific modeling decisions were made for the intended sentiment analysis system. These decisions concerned the programming language, tools, and libraries used, key design principles to follow, and consequently, the program's architecture.

3.1.4.1 Programming Language and Libraries

Data-driven projects can be implemented using various programming languages, such as Python, R, SQL, Java, Scala, Julia, and MATLAB, each with different strengths and weaknesses suitable for specific types of data-driven projects. For this master project, Python was chosen because its features and libraries offered broad support for the project's needs. The modeling requirements listed in Section 2.1.2.5 are compared below with the features of Python:

- **Modularity and extensibility:** In addition to procedural programming, Python also allows object-oriented programming and thus facilitates the creation of a modular architecture with individual, independent components (e.g., data preparation, integration of BLOOM, evaluation), which can be created, adapted or extended flexibly and prepare the possibility of integrating additional modules or frameworks, such as other LLMs or new data sets, without having to make fundamental changes to the overall system. These characteristics support the iterative optimization of the system as envisaged in the evolutionary prototyping approach of this project.

- **Support for data processing**: Python offers libraries like pandas and numpy, optimized for efficiently processing large and complex datasets. Pandas enables efficient management of tables and DataFrames, suitable for organizing and analyzing structured data. Numpy supports high-performance numerical computations and is specially optimized for computationally intensive tasks, but is also well suited for managing data types in pandas DataFrames. Both libraries complement each other and facilitate the implementation of the data preparation steps described in Section 2.3, such as cleaning, transforming and normalizing data sets.

- **Support for the integration of generative LLMs:** The Hugging Face transformers library enables Python-based integration of generative, transformer-based language models like BLOOM. This library allows downloading and processing model shards and provides pipelines for specific tasks, such as sentiment analysis. Even though the LLM selection process in Section 3.1.3 revealed that, in this project, the transformers library could not be used to address BLOOM directly or via pipelines due to a lack of sufficient resources in the author's system, it is conceivable that the finished sentiment analysis program will be used in the future in environments that allow direct work with an LLM or the use of pipelines with the transformers library. Therefore, despite its ultimate non-use in the project, the transformers library was another reason to choose Python as the programming language.

- **NLP support:** Python libraries such as spaCy and NLTK offer extensive NLP capabilities like tokenization, lemmatization, and part-of-speech tagging, which are often required for data preparation and understanding linguistic peculiarities in different languages of a data

collection. Although the project's use of natural language input sentences made lemmatization and part-of-speech tagging unnecessary, tokenization methods still were needed for analysis purposes.

- **Support for statistical measurement, evaluation and visualization:** Python has libraries such as scikit-learn, matplotlib and seaborn that cover all aspects of evaluation and visualization. Scikit-learn enables the calculation of metrics such as precision, recall and F1 score, whose importance in the project for evaluating the modeling results was highlighted in Section 2.1.2.7. Matplotlib and seaborn support the graphical representation of evaluation results, especially for cross-lingual comparison of results. These tools ensure that the results can be analyzed both quantitatively and qualitatively.

Thus, Python's extensive library ecosystem met all modeling requirements for this project. Additionally, various Python-based tools also facilitated implementation and modeling tasks. The tools used are described in the next section.

3.1.4.2 Tools

Choosing Python as the programming language enabled the use of various Python-based tools. The following tools proved to be useful for implementation and modeling with Python:

3.1.4.2.1 IDE

As integrated development environment (IDE), Pycharm was used. Designed specifically for Python, PyCharm offers the usual functions of an IDE such as code completion, code navigation support, debugging, version control integration, refactoring support, a built-in package manager, as well as automatic virtual environment creation and built-in pandas DataFrame visualization. PyCharm's functionality can be expanded with plugins, including diagramming tools and static code checkers for monitoring code quality.

3.1.4.2.2 Diagramming Tools

PyCharm's built-in diagram tool automatically generates class diagrams from existing code, visualizing inheritance structures. Additionally, the PlantUML plugin was installed, enabling easy text-based creation of class diagrams, sequence diagrams, and other types of diagrams directly within the IDE. This plugin was used, for instance, to create the dependency diagram in Figure 19 and the activity diagrams in Appendix I.3.

3.1.4.2.3 Static Code Checkers

To monitor and improve code quality, several static code checker plugins were installed in PyCharm to help detect and eliminate deviations from programming guidelines:

- **Pylint**: Checks Python code for style guide compliance, syntax errors, and potential bugs.
- **Mypy:** Conducts static type checking, offering type safety not natively enforced by Python.
- **SonarLint:** Complements Pylint by identifying vulnerabilities, code smells, and potential security risks.

With the help of these code checkers, many inaccuracies and formatting errors in the code could be eliminated.

3.1.4.2.4 Code Documentation Tools

Code documentation typically uses docstrings. Python supports three main docstring styles: Google style, reStructured Text, and NumPy style. The differences between the three styles are illustrated in Listing 1. All three docstring styles provide structured documentation rules, ensuring consistent docstring formatting, concisely aiding users in understanding the purpose, input, and output of modules, classes, methods, and functions. In the master project, the NumPy style was adopted for its clear structure and readability, despite its longer text format.

Numpy Style

```
def are_equal(data: T, other: T) -> bool:
    """
    Generic function to check if two objects are equal.

    This singledispatch mechanism provides custom comparison functionality
    for data types that do not allow for a simple equality check with the
    '==' operator.

    Parameters
    ----------
    data : T
        The data to compare to other data

    other : T
        Other data to compare with the data

    Returns
    -------
    bool
        'True' if the data and the other data are equal. 'False' otherwise.

    Raises
    ------
    NotImplementedError
        If the data type of the provided parameters is not covered by one of
        the registered functions of this singledispatch mechanism.

    Notes
    -----
    This function defines the singledispatch mechanism that delegates the
    equality checks to the registered specialized function that covers the
    the respective type of the input values. If there is no special function
    for the data type of the input values, this function serves as a
    "catch-all" function, returning a NotImplementedError.
    """
```

Google Style

```
def are_equal(data: T, other: T) \
        -> bool:
    """
    Generic function to check if two objects are equal.

    This singledispatch mechanism provides custom comparison functionality
    for data types that do not allow for a simple equality check with the
    '==' operator.

    Args:
        data (T): The data to compare to other data.
        other (T): Other data to compare with the data.

    Returns:
        bool: True if the data and the other data are equal. False otherwise.

    Raises:
        NotImplementedError: If the data type of the provided parameters is
        not covered by one of the registered functions of this singledispatch
        mechanism.

    Notes:
        This function defines the singledispatch mechanism that delegates the
        equality checks to the registered specialized function that covers the
        respective type of the input values. If there is no special function
        for the data type of the input values, this function serves as a
        "catch-all" function, returning a NotImplementedError.
    """
```

ReStructuredText Style

```
def are_equal(data: T, other: T) -> bool:
    """
    Generic function to check if two objects are equal.

    This singledispatch mechanism provides custom comparison functionality
    for data types that do not allow for a simple equality check with the
    '==' operator.

    :param data: The data to compare to other data.
    :type data: T
    :param other: Other data to compare with the data.
    :type other: T
    :return: True if the data and the other data are equal. False otherwise.
    :rtype: bool
    :raises NotImplementedError: If the data type of the provided parameters
        is not covered by one of the registered functions of this
        singledispatch mechanism.

    .. note::
        This function defines the singledispatch mechanism that delegates the
        equality checks to the registered specialized function that covers the
        respective type of the input values. If there is no special function
        for the data type of the input values, this function serves as a
        "catch-all" function, returning a NotImplementedError.
    """
```

Listing 1: Numpy vs. Google vs. reStructured text docstring style

To further enhance project documentation, Sphinx was employed as an automated documentation generator. This tool extracts docstrings from the source code and generates professional-quality documentation in formats such as HTML, PDF, or Markdown. Its extension system, including the Napoleon extension, ensured seamless support for the NumPy docstring style.

3.1.4.2.5 Sandbox

As an experimental environment, the tools Anaconda Navigator and Jupyter Notebooks were used, e.g. for exploratory data analysis and experimental coding phases, including testing queries to BLOOM. Jupyter Notebooks provide an interactive workspace where code, results, and visualizations can be combined within the same environment. This provided immediate feedback in coding experiments, enhancing rapid development and debugging. Another advantage was that, unlike when executing code in PyCharm or a console, graphical outputs remained visible without requiring separate storage, as long as the respective notebook cells were not re-executed.

3.1.4.2.6 Conclusion

The combination of PyCharm, PlantUML, Pylint, Mypy, SonarLint, Sphinx, Anaconda Navigator and Jupyter Notebooks provided comprehensive support for the development of the sentiment analysis system. The tools significantly improved the efficiency of the implementation, the quality of the code and the traceability of the results. Together with the Python libraries, they formed the basis for the successful completion of the project.

The following sections describe the design principles and architecture based on these technological decisions.

3.1.4.3 Key design principles

Based on the project objectives, the methodological approach and the technical requirements defined during the business understanding phase, a set of general design principles was derived for the modeling phase. These principles were intended to enable the iterative development of a workflow for multilingual sentiment analysis under limited resources, while ensuring the flexibility, traceability, and robustness of the program. The key design principles are outlined below. Their concrete implementation is described in the modeling sections under 3.4.

3.1.4.3.1 Separation of Concerns

To enable flexible development and allow individual components to be improved independently, the system was to be designed in a modular fashion. Functional components (e.g., various datasets or language models) should be interchangeable, combinable or

expandable without requiring modifications to unrelated parts of the system. This separation of concerns [122–124] required low coupling and adherence to the Single Responsibility Principle [125].

3.1.4.3.2 Iterative Refinement

As the modeling approach followed the CRISP-DM process and employed evolutionary prototyping, the system had to support repeated cycles of implementation, testing, and refinement. Consequently, design structures were preferred that allowed continuous optimization—especially in the areas of prompt design and evaluation.

3.1.4.3.3 Configurability

To facilitate adaptation to changing parameters, the system was to be designed in a way that allowed central settings (e.g., language options, analysis modes, dataset selections) to be defined and adjusted by the user. This was intended to ensure both systematic experimentation and reproducibility of results.

3.1.4.3.4 Maintainability

Given the project's scope and the limitation to a single developer, particular emphasis was placed on a system architecture that would be easy to understand and maintain. This included consistent naming conventions, a clear module structure to ensure future extensibility and traceability and a thorough documentation of the programm.

3.1.4.3.5 Robustness

Since the workflow involved long-running processes and potential external disruptions (e.g., network latency or service interruptions), the system had to be designed with a focus on stability and fault tolerance. Appropriate mechanisms such as intermediate saving, checkpoints, and automatic backups were intended to prevent data loss and enable the resumption of processes.

3.1.4.4 Architecture

In line with the need for flexibility, scalability and resource efficiency outlined in Section 2.1.2.5, the program's architecture was designed with modularity and extensibility as core principles (s. Figure 4). The "sentiment_analysis" component is primarily divided into three subcomponents: "prompt_engineering", "retrieval", and "evaluation", all orchestrated by the ServerlessBloomWorkflow. The workflow sets the relevant global configurations, provides access to the MAD-TSC data and – depending on the task specified in the program's main method – invokes the corresponding sentiment analysis component. In addition to the sentiment_analysis component, other components such as "data_sources" and "authentication" are used to access the original data from the MAD-TSC corpus. Various other

components can be called from all parts of the program to manage data (“data_structures”), handle serialization (“serialization”), perform natural language processing tasks (“nlp”), carry out statistical analyses (“stats”), and utilize a range of utility functions (“utils”, “decorators”).

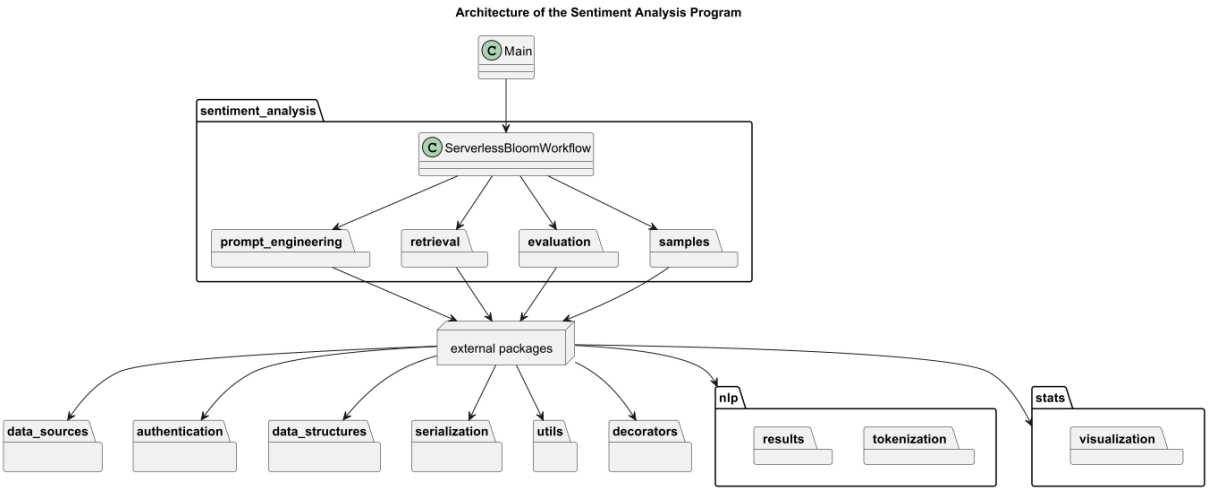


Figure 4: Architecture of the Sentiment Analysis Program

3.2 Data Provisioning and Understanding

The MAD-TSC corpus [57] was manually downloaded as a ZIP file from the authors’ GitHub repository[13] and unzipped. It contains three folders, two of which contain machine translations of the original data, which the authors created on a trial basis using two different machine translation systems to investigate the suitability of modern machine translation for sentiment analysis in foreign languages. This data was not required for the present study and was not considered further.

The third folder, labeled “original”, contains the original data sought for this study. These are divided into subfolders, one for each language represented in the corpus. Each language folder contains three JSONL files, representing the test, training, and validation subsets.These language folders were manually stored in the project under data/jsonl/mad_tsc (see Figure 5).

[13] https://github.com/EvanDufraisse/MAD_TSC

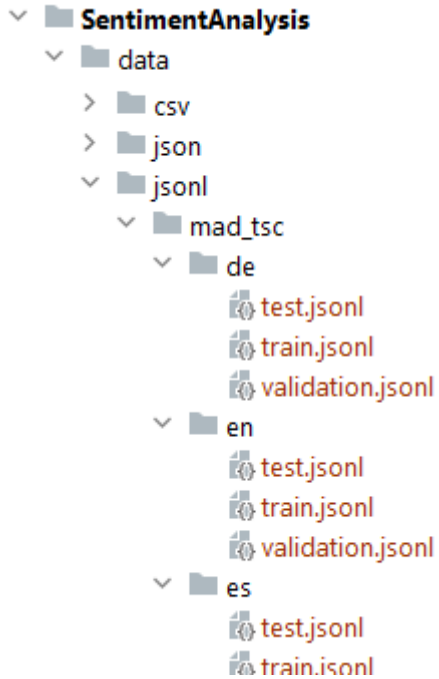


Figure 5: Directory structure of the original MAD-TSC files

A review of the files and their sizes suggested that all relevant files were present in the downloaded folders and provided no indication that any of the data files might be incomplete. Each “test” file contained 1,000 samples, each “train” file contained 3,810 samples and each “validation” file contained 300 samples.

3.3 Data Preparation

Each data file contained three data fields per row:

- The **“primary_gid”** field contained a unique identifier for each sample. Corresponding samples across different languages shared the same primary_gid, allowing them to be identified as different language versions of the same sample.

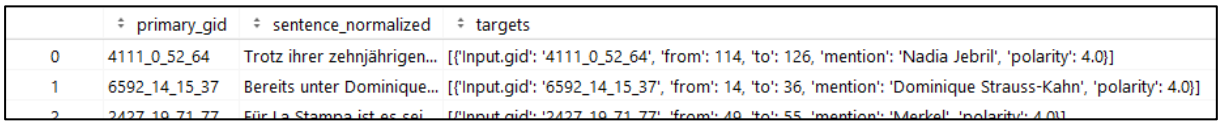
- The **“sentence_normalized”** field contained the text of the sample. Typically, a sample consisted of a single sentence. However, due to the nature of translations, this was not always the case. Depending on the structures of the various languages, a single sentence in the source language (the language being translated from) could be rendered as multiple sentences in the target language (the language being translated into). Conversely, multiple sentences in the source language could sometimes be combined into a single sentence in the target language. As a result, a sample in any language included in the corpus consisted of at least one sentence.

- The „**targets**” field was a more complex data structure resembling a dictionary object in Python. It contained a single key, “targets”, which was associated with another dictionary as its value. The inner dictionary specified sentiment annotations for the sample, including the following features:

    - **“Input.gid”:** identifier corresponding to the primary_gid
    - **“mention”:** the person’s name identified in the sentence
    - **“from”:** position of the first character of the mention in the sentence
    - **“to”:** position of themention’s last character in the sentence
    - **“polarity”:** numerical sentiment label (either 2.0 for “negative”, 4.0 for “neutral” or 6.0 for “positive”).

3.3.1 Loading and preprocessing

The contents of the files were first loaded in pandas DataFrames. The resulting data structure is shown in Figure 6:



	primary_gid	sentence_normalized	targets
0	4111_0_52_64	Trotz ihrer zehnjährigen...	[{'Input.gid': '4111_0_52_64', 'from': 114, 'to': 126, 'mention': 'Nadia Jebril', 'polarity': 4.0}]
1	6592_14_15_37	Bereits unter Dominique...	[{'Input.gid': '6592_14_15_37', 'from': 14, 'to': 36, 'mention': 'Dominique Strauss-Kahn', 'polarity': 4.0}]
2	2427_19_71_77	Für La Stampa ist es sei...	[{'Input.gid': '2427_19_71_77', 'from': 49, 'to': 55, 'mention': 'Merkel', 'polarity': 4.0}]

Figure 6: Original MAD-TSC data structure, loaded into a pandas DataFrame

As the project did not require a division into test, training and validation sets, the test, train and validation subsets were merged into one pandas DataFrame per language with 5110 samples each, detailing the original file in an additional column. The resulting DataFrames were preprocessed as follows:

Details from the original “targets” column were extracted into separate columns, with targets (called “mention”) and polarity ratings placed in dedicated columns. Additionally, various computed columns were added for analysis purposes:

- The “**n_duplicates**” column contains the number of text duplicates created by the authors of the MAD-TSC dataset as a result of multiple targets being identified in a sentence.
- The “**length**” column indicates the character count of each text sample.
- The “**n_sentences**” column contains the number of sentences, counted after sentence segmentation with the sentence tokenizer.
- The “**n_words**” column contains the word count of each sample, counted after applying the word tokenizer.

The preprocessing steps per language are illustrated in Figure 7.

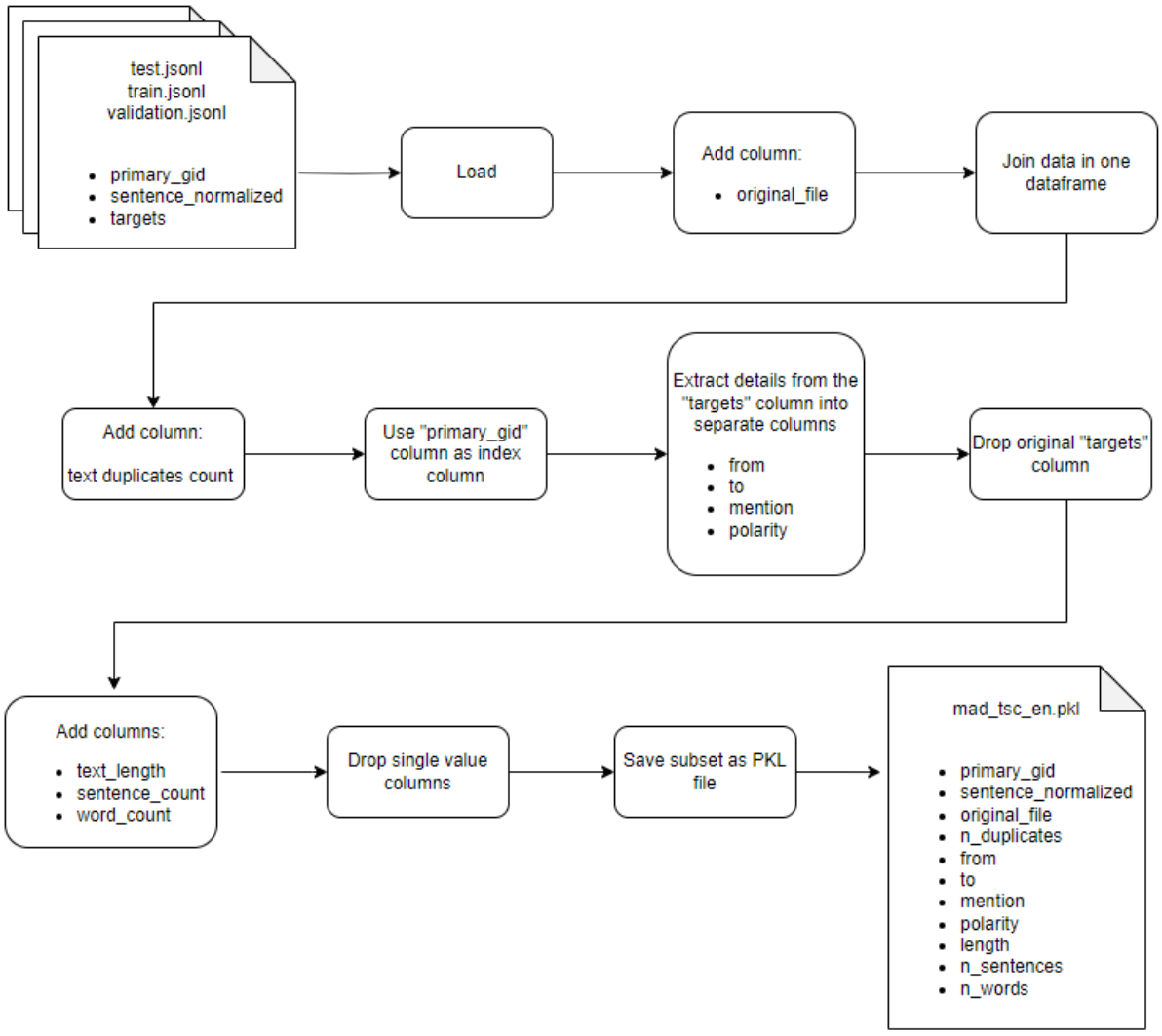


Figure 7: Data preprocessing using the example of the English data

The prepocessed language datasets were saved as PKL files in the project under data/pkl An example of the preprocessed data is shown in Figure 8:

	sentence_normalized	original_file	n_duplicates	from	to	mention	polarity	length	n_sentences	n_words
4111_0_52...	Trotz ihrer zehnjährigen...	train	1	114	126	Nadia Jebril	4.00000	197	1	25
6592_14_1...	Bereits unter Dominique...	train	1	14	36	Dominique...	4.00000	116	2	14
2427_19_7...	Für La Stampa ist es sei...	train	1	49	55	Merkel	4.00000	314	2	51
3987_25_5...	Sie dachten, befürchtete...	train	1	62	72	Berlusconi	2.00000	154	1	20
1836_6_4_...	Marcel Gauchet fragt si...	train	1	0	14	Marcel Ga...	4.00000	80	1	12
714_0_153...	Der offene Brief, in welc...	train	2	164	178	David Goo...	2.00000	344	1	45
1915_3_31...	Gibraltars Regierungsch...	train	1	26	39	Peter Caru...	4.00000	128	1	15
1789_1_28...	Während der maltesisc...	train	1	38	48	Tonio Borg	6.00000	336	2	45
842_24_33...	Genau betrachtet wird k...	train	1	248	255	Tsipras	4.00000	317	5	51
1113_0_80...	Am 6. Mai 2012 hat Fra...	train	1	19	36	François H...	2.00000	410	6	61
1091_0_72...	Der Druck von Seiten d...	train	1	86	99	David Cam...	4.00000	250	1	34

Figure 8: Preprocessed German data saved in data/pkl/mad_tsc_de.pkl.

The combined language datasets in this project are referred to as “subsets” and collectively, they are called the “suite”. The MadTscWorkflow class in the “data_sources” package provides users with access to functions of the MAD-TSC data suite and its various language subsets. The class’s main function is to load the suite and the subsets using the “execute” method. When the subsets are loaded for the first time, they are created from the original files with the preprocessing steps described above and saved in the src/data/pkl folder. For subsequent calls, the preprocessed subsets are loaded directly from the PKL files.

3.3.2 Data Analysis

The preprocessed data were then checked for consistency and completeness to determine whether further preprocessing was needed. The most important findings are discussed in the following subsections. The detailed corpus statistics is provided in Appendix H.

3.3.2.1 Data Gaps

First, the data was checked for gaps. The heatmap of missing values shows no data gaps for any of the features (see Figure 9).

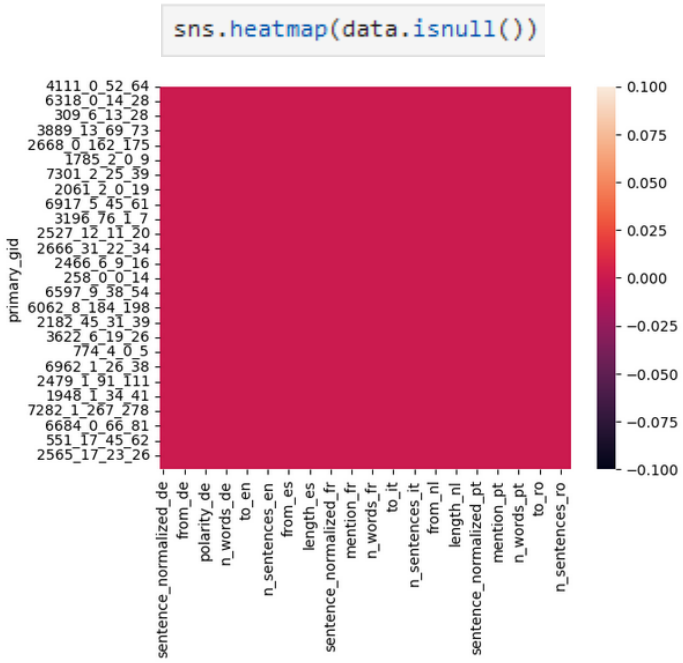


Figure 9: Heatmap of missing values

Thus, no data needed to be excluded or added due to incompleteness.

3.3.2.2 Text Lengths

Next, text lengths were analyzed for anomalies (see Figure 10).

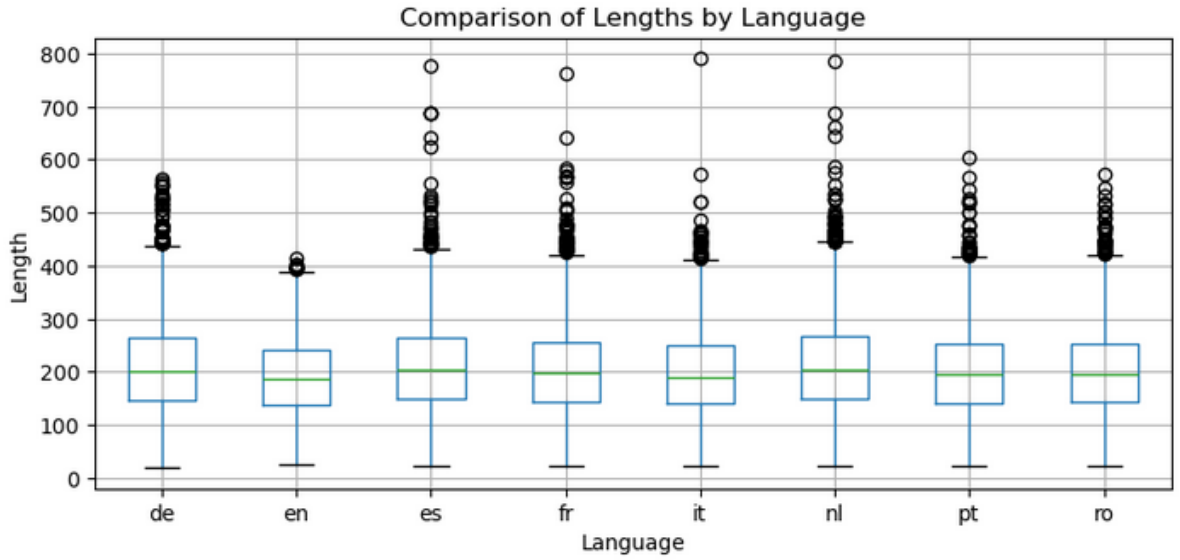


Figure 10: Comparison of sample text lengths across languages

The text lengths of samples across languages show very similar medians and quartiles. Outliers are particularly noticeable in all languages except English, which, according to the authors, is often due to the requirements of the translation process. Concepts that are simple and easily understood in the source language may require longer explanations in another language. Therefore, the significant difference in text-length outliers indicates not only that English tends to be more concise in comparison to other languages, but also that most of the samples were likely translated from English. As the authors of the data collection found, these differences in text lengths do not significantly impact the expression of sentiment [57, p. 8288]. Therefore, there was no need for action here either.

3.3.2.3 Sentence Lengths

The number of sentences identified using the tokenizer was not always one, even though target-dependent sentiment annotation had been primarily performed at the sentence level (see Figure 11). This discrepancy was partly due to the tokenizer misinterpreting abbreviations or numbers with periods as sentence endings, and partly due to translation peculiarities where sentences may have been split or merged. The maximum number of sentences was only six, though, the median sentence count consistently being one, as expected.

	n_elements	min_sentences	max_sentences	mean_sentences	median_sentences	std_dev_sentences
mad_tsc_de	5110	1	6	1.48728	1.0	0.709893
mad_tsc_en	5110	1	3	1.102935	1.0	0.324463
mad_tsc_es	5110	1	5	1.17593	1.0	0.438605
mad_tsc_fr	5110	1	5	1.192955	1.0	0.451557
mad_tsc_it	5110	1	4	1.176321	1.0	0.425304
mad_tsc_nl	5110	1	5	1.306262	1.0	0.559916
mad_tsc_pt	5110	1	5	1.195303	1.0	0.454009
mad_tsc_ro	5110	1	4	1.179061	1.0	0.428748

Figure 11: Minimum, maximum and average sentence counts and standard deviations by language

The sentence counts having revealed no indication of necessary action, the analysis of word counts did not appear to be relevant either and was omitted.

3.3.2.4 Sentiment Labels

The review of the sentiment labels in the different languages showed that the labels of given samples were the same across all language versions (see Figure 12).

```
'Polarity' columns are identical.
'Polarity' columns are identical.
'Polarity' columns are identical.
'Polarity' columns are identical.
'Polarity' columns are identical.
'Polarity' columns are identical.
'Polarity' columns are identical.
```

Figure 12: Comparison of sentiment labels across seven languages with the German dataset

The labels were thus consistent across all languages.

3.3.2.5 Targets

The comparison of the “mention” columns in the datasets showed that target names could vary by language. This could be seen in particular in language-specific accents, which could be omitted in some languages whereas they were precisely added in others (e.g., “Miloš Zeman” vs. “Milos Zeman”), as well as in the use of first and last names, which could vary depending on the political culture and linguistic norms of a given language (“José Manuel Barroso” vs. “Barroso” vs. “José Manuel Durão Barroso,” etc.). As the diagrams in Figure 13 reveal, the frequency distributions of mentions across languages exhibited differences while maintaining recognizable similarities. However, it should be noted that the frequency of a specific individual could not be directly inferred from these diagrams, as naming conventions seemed to vary within the same language (e.g., “Angela Merkel” vs. “Merkel”).

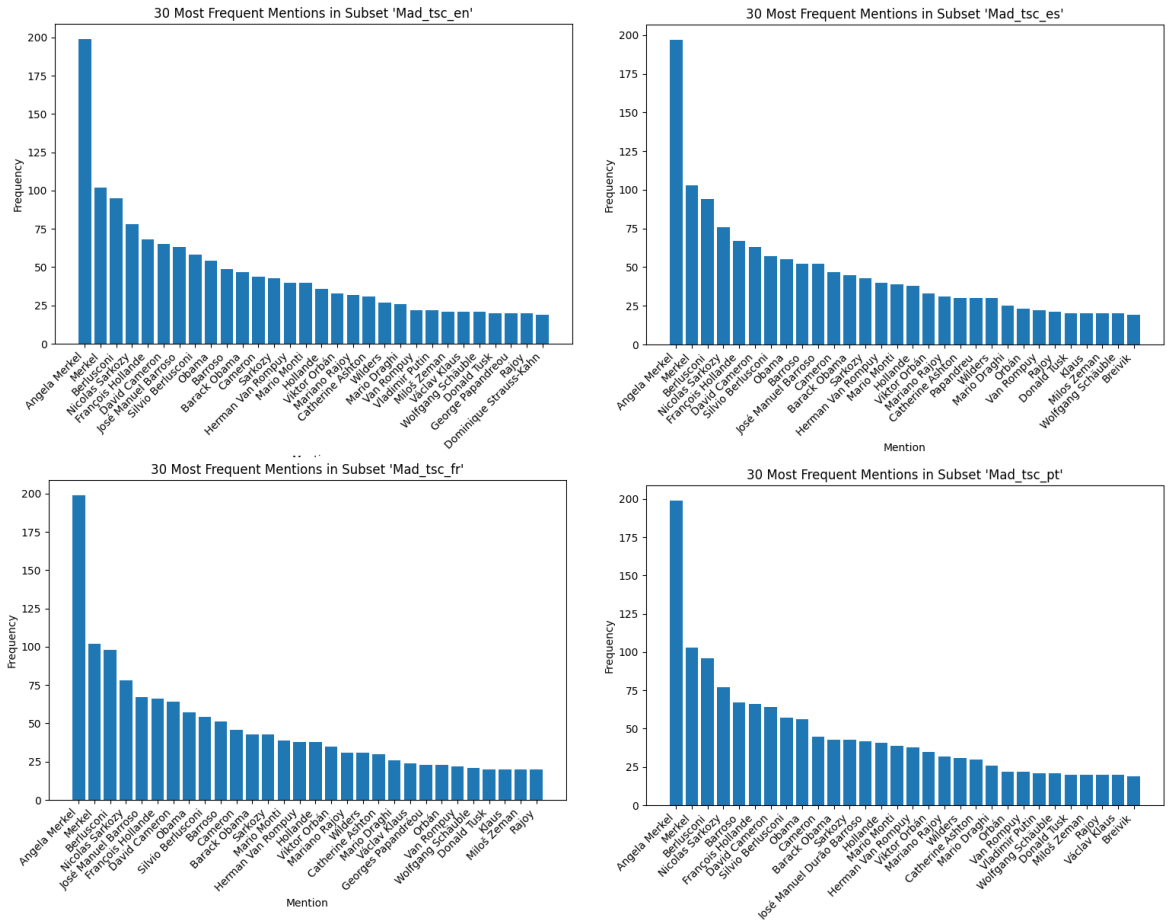


Figure 13: Frequency distributions of MAD-TSC mentions in the languages supported by BLOOM (EN, ES, FR, PT)

3.3.2.6 Class Imbalance

The distribution of sentiment classes in the MAD-TSC corpus was not entirely balanced, as the table in Figure 13 illustrates. Approximately 36% of the samples were annotated as “negative”, 39% as “neutral” and only about 25% as “positive”.

min_polarity	2
max_polarity	6
mean_polarity	3.773386
median_polarity	4.0
std_dev_polarity	1.541085
unique_polarity	[2.0, 4.0, 6.0]
n_positive	1260
n_negative	1839
n_neutral	2011
%_positive	24.657534
%_negative	35.988258
%_neutral	39.354207

Figure 14: Statistics of sentiment classes

The distribution of sentiment classes within the corpus was also uneven. This imbalance was also reflected within subsets of the corpus. If the corpus were divided into 52 batches of 100 samples each, the distributions shown in Figure 15 would result.

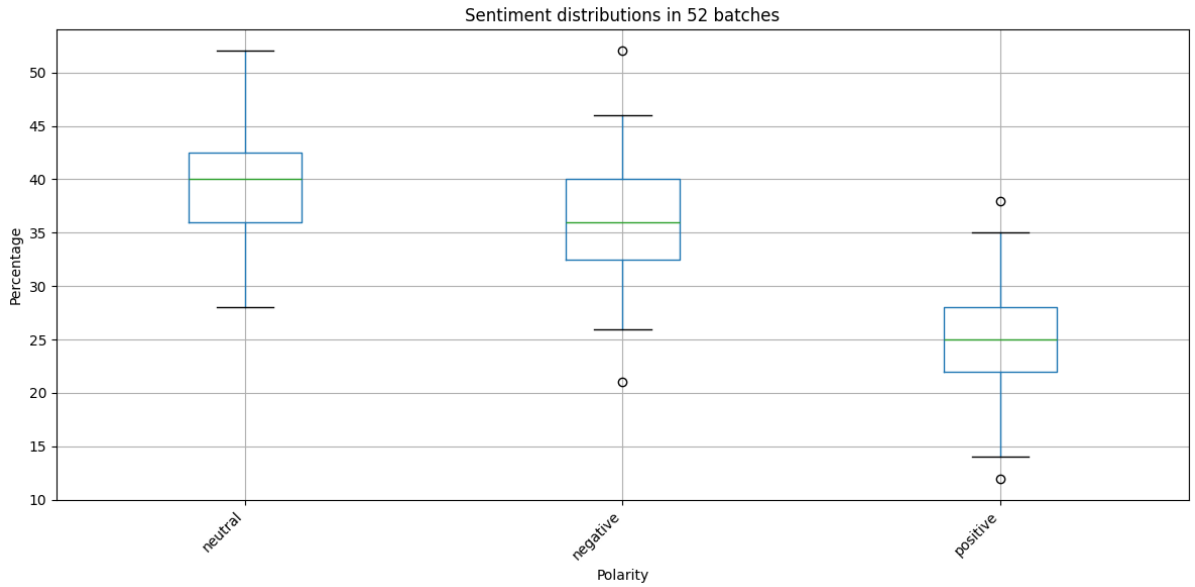


Figure 15: Distribution of sentiment classes across 52 batches of 100 samples each

Depending on the batch, the proportions of the three sentiment classes could vary significantly from the overall average – especially when the whiskers and outliers came into play. Since the sentiment analysis performed in this project could only rely on a small subset of the full dataset,

the risk arose that the data selected for prompt engineering might reflect a random, unrepresentative sentiment distribution. This could lead to biased engineering outcomes, causing the resulting prompts to perform worse across the full dataset than they otherwise might have. Moreover, an imbalanced distribution complicates the interpretation of metric results. Therefore, a balanced set of positive, negative, and neutral samples was selected for sentiment retrieval. Further details on the creation of this balanced dataset for all languages supported by BLOOM and the MAD-TSC corpus can be found in Section 3.4.3.2.

3.3.2.7 Conclusion

No evidence was found to suggest that the MAD-TSC datasets had deficiencies that would limit their usability for the intended purposes. The data exhibited the highest possible degree of completeness and consistency, as suggested by the authors' careful curation of the datasets. Therefore, additional preprocessing was only required to adapt the data to technical needs of the intended implementation, like the creation of balanced multilingual subsets and the conversion of the sentiment labels to more easily understandable textual labels instead of the original numerical format.

3.4 Modeling and Evaluation

Since the project was based on the CRISP-DM process and followed an evolutionary prototyping approach, evaluation was closely linked to the iterative modeling process—that is, it was an integral part of model development. After each implementation of one of the prototypes, an evaluation was conducted using statistical methods and standard metrics. Therefore, modeling and evaluation are discussed together in this section.

3.4.1 Project Structure

To conduct the intended target-dependent sentiment analysis with BLOOM and use the MAD-TSC data collection as the gold standard, an object-oriented "SentimentAnalysis" program was developed in Python.

The basic structure of the program is shown in Figure 16.

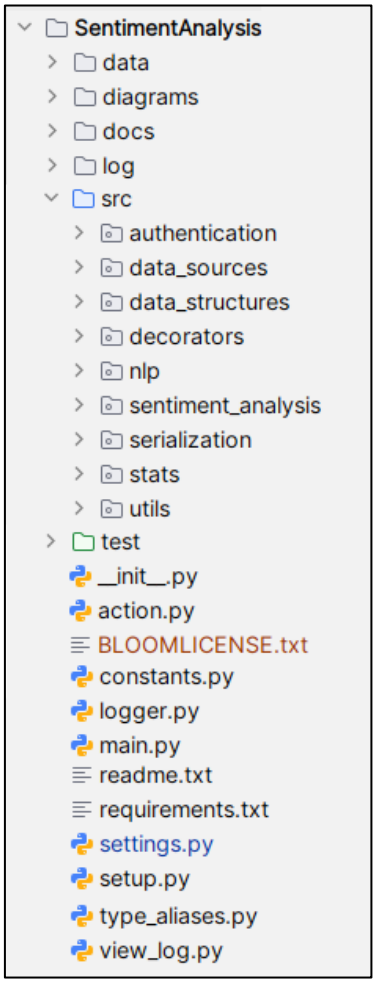


Figure 16: The SentimentAnalysis program, implemented in Python

At the top level of the program are files that are required to run the program (BLOOMLICENSE.txt, readme.txt, setup.py, main.py, and view_log.py) or that contain global settings for the program (constants.py, settings.py, requirements.txt, type_aliases.py, action.py, logger.py). The "**data**" directory contains the data saved by the program in various file formats, "**docs**" contains the program documentation generated with Sphinx, "**log**" holds the log files, "**src**" contains the main program code, and "**test**" includes various unit tests.

The actual program in the "**src**" directory is divided into several packages. The "**authentication**" package manages access to the Hugging Face platform for the use of BLOOM. The "**data_sources**" package contains classes and methods for managing and utilizing source data, specifically for handling and utilizing the MAD-TSC datasets. The "**data_structures**" package provides various data structures to work efficiently with the datasets. The "**decorators**" package offers various decorators to offload method and function code from recurring data checks, error routines, and timing methods. The "**nlp**" package includes NLP functions, particularly word and sentence tokenizers, needed for dataset

analysis. The “**sentiment_analysis**” package contains the code for performing all tasks associated with sentiment analysis. The “**serialization**” package provides serialization functions and management methods for various data and file formats. The “**stats**” package includes code for statistical analyses and visualizations. Finally, the “**utils**” package provides various helper functions.

The program was designed not only to load and use a single dataset and interact with a single LLM but also to allow for the straightforward addition of multiple datasets and LLMs for comparison. The program's structure was therefore created to enable relatively easy expansion, even though only one dataset and one LLM were analyzed in the context of this thesis.

The object-oriented SentimentAnalysis program makes extensive use of various programming techniques such as the strategy pattern (e.g. for the serialization of various data formats, see Appendix I), mixins, decorators, factory classes and singledispatch mechanisms. Their understanding is essential for modifying or extending the code and this knowledge is assumed for anyone intenting to modify the code. The task of this thesis is not to describe the code, but rather to explain the workflow that the program follows in order to perform the sentiment analysis with BLOOM and evaluate its results. Detailed explanations of the code can be found in the program documentation, the docstrings within the code, and the “readme.txt” file.

3.4.2 Accessing BLOOM

As already shown in Section 3.1.3.6, BLOOM can be queried via a serverless inference API [120] using a Hugging Face User Access Token [121], as illustrated in Figure 3. To enable this functionality in the SentimentAnalysis program, the “HuggingFaceStrategy” class was created. This class provides a query method, which allows queries to be sent to the BLOOM API and returns its responses.

The Hugging Face Access Token was stored in the system environment variable HUGGING_FACE_AUTH_TOKEN on the Windows system running the SentimentAnalysis program. This approach ensured secure access to the token without exposing it to the program user. The HuggingFaceStrategy class reads this environment variable into a private __access_token variable:

```
1  import os
2
3  self.__access_token = os.getenv('HUGGING_FACE_AUTH_TOKEN')
```

Listing 2: Reading the user access token for use in BLOOM queries

3.4.3 BLOOM Workflow

The “ServerlessBloomWorkflow” class manages the prompt engineering, validation and optimization process, as well as the execution of sentiment analysis and the evaluation of its results.

3.4.3.1 Sentiment analysis configuration

As part of the BLOOM workflow, the intended sentiment analysis is configured. For this purpose, the SentimentAnalysisConfig class provides a singleton instance that manages the configuration settings (see
Listing 3). As a singleton, the class allows querying required parameters from anywhere in the program without needing to create new instances or pass around an existing instance between classes.

The following options are managed in the SentimentAnalysisConfig singleton:

- **Batch Size:** The default setting for the number of samples to be processed in a batch is 100. For balanced data the size is set to 99 in the BLOOM workflow.

- **Number of batches to process:** The default setting is 1 and was not changed during the course of the project due to the time required for sentiment queries.

- **Chunk Size:** The default is 15 prompts, which allows 15 prompt variants to be processed with one batch of 99 or 100 samples before the rate limit forces a pause.

- **Prototype version:** This corresponds to the prompt-engineering strategy number. The default is '00', representing an invalid value.

- **Data offset:** Determines how many samples to skip during analysis. The default is 0 but can be set to any value between 1 and 5109. As prompt engineering used the first sample for pre-validation of the prompts, subsequent sentiment analysis skipped the first sample.

- **“From_sample” and “to_sample”:** Instead of the data offset, the start and end number of samples to be processed can be specified. The default settings are 0 and 9999999, imposing no restrictions.

- **“Balanced” setting:** A Boolean value indicating whether the samples should be balanced. The default is True.

```
class SentimentAnalysisConfig:
    """ SentimentAnalysisConfig class.... """

    _instance: SentimentAnalysisConfig | None = None
    _settings: Dict[str, Any] = {}
    _initialized: bool = False

    log = LoggingMixin().log

    def __new__(cls, *args: Any, **kwargs: Any) \
            -> SentimentAnalysisConfig:
        """ Creates and initializes a singleton instance of this class.... """

        if cls._instance is None:
            cls._instance = super(SentimentAnalysisConfig, cls).__new__(cls)

        cls._initialize(*args, **kwargs)
        return cls._instance

    @classmethod
    def _initialize(
            cls,
            api: str = '',
            llm: T | None = None,
            from_sample: int = 0,
            to_sample: int = 9999999,
            batch_size: int = 100,
            data_offset: int = 0,
            n_batches: int = 1,
            chunk_size: int = 15,
            prompt_group: List[int] = None,
            version: str = '00',
            balance: int = 33,
            balanced: bool = False,
            n_best_prompts: int = 5,
            target_n_prompts: int = 150,
            with_validation: bool = True,
    ) -> None:
        """ Initializes the configuration settings with the provided values. """

        if not cls._initialized:
            cls.set('api', api)
            cls.set('llm', llm)
            cls.set('from_sample', from_sample)
            cls.set('to_sample', to_sample)
            cls.set('batch_size', batch_size)
            cls.set('data_offset', data_offset)
            cls.set('n_batches', n_batches)
            cls.set('chunk_size', chunk_size)
            cls.set('prompt_group', prompt_group)
            cls.set('version', version)
            cls.set('balance', balance)
            cls.set('balanced', balanced)
            cls.set('n_best_prompts', n_best_prompts)
            cls.set('target_n_prompts', target_n_prompts)
            cls.set('with_validation', with_validation)

            cls._initialized = True

    @classmethod
    def get(cls, key: str) \
            -> Any:
        """ Retrieves the value associated with a given key from the settings...."""

        return cls._settings.get(key)

    @classmethod
    def set(cls, key: str, val: Any) \
            -> None:
        """ Sets the value for a given key in the settings.....  """

        cls._settings[key] = val
```

Listing 3: Singleton implementation of the SentimentAnalysisConfig class

- **„Balance" value:** Specifies how many samples to use for each sentiment category if "balanced" is set to True.

- **Number of best prompts.** Indicates how many top prompts should be output. The same value is used for the number of worst prompts.

- **Target number of prompts:** The number of promps to generate during prompt engineering. The default is 150, which turned out to be a feasible number for performing prompt engineering with only one language.

- **"With Validation" setting:** A Boolean value indicating whether result validation should be performed during sentiment analysis. The default is True.

In addition to these optional values, information about the **LLM** used and the **API** for accessing the LLM is also stored in the singleton. These values are set automatically when a specific LLM is invoked.

3.4.3.2 Balancing Sentiment Classes Across Languages

In the sentiment analysis configuration within the BLOOM workflow, the balanced value was set to True, the number of samples per sentiment class was set to 33, and the batch size was set to 99 (see Listing 4). This configuration caused the SamplesManager, which is called when sentiment analysis starts to provide samples for the analysis, to create balanced datasets using the BalancedSamplesProvider. These datasets contained 99 samples for each relevant language subset in the data suite, with each sentiment class appearing 33 times.

```
class ServerlessBloomWorkflow(LoggingMixin):

    def __init__(self):
        """ Initializes the workflow with default configuration and model setup…. """

        self.config = SentimentAnalysisConfig(
            batch_size=99,
            n_batches=1,
            chunk_size=15,
            version='01',
            data_offset=1,
            balanced=True,
            balance=33,
            n_best_prompts=5,
            target_n_prompts=150,
            with_validation=True
        )

        # Initialize the ServerlessBloom class
        self.llm = ServerlessBloom()
```

Listing 4: Initialization of the serverless BLOOM workflow

The class diagram in Figure 17 illustrates the relationships between the SamplesManager and the sample provider classes. The SamplesManager delegates the sample generation task to either the UnbalancedSamplesProvider or the BalancedSamplesProvider, depending on the settings in the SentimentAnalysisConfig class. Both providers implement the SamplesProvider base class.

Figure 17: Samples manager and samples providers

When the SamplesManager calls the BalancedSamplesProvider, it first checks whether balanced data is already stored on disk. If such data exists, it loads and returns it to the SamplesManager. If not, the BalancedSamplesProvider retrieves unbalanced samples from the SamplesManager, which in turn gets them from the UnbalancedSamplesProvider. The BalancedSamplesProvider then creates a balanced reference dataset using data from one language and uses this reference to extract corresponding samples from other languages. The balanced samples are saved to disk and returned to the SamplesManager. A detailed representation of this workflow can be found in the activity diagram in Appendix I.2.

3.4.3.3 Sample Structure

When the samples were loaded, the _get_sentiment_df method of the UnbalancedSamplesProvider ensured that the ten features, along with the index column containing sample IDs from the preprocessed data (see Figure 8), were reduced to three: sentence_normalized, mention, and polarity. Additionally, the original numeric polarity values

were converted into textual labels: “negative”, “positive” and “neutral”. Figure 18 shows the structure of the samples.

	sentence_normalized	mention	polarity
1830_1_64...	The answer, Finkelstein ...	Lennon	neutral
3500_0_14...	"Europe is more worrie...	Abdel Mus...	positive
5322_37_4...	Indeed, the Northern Le...	Silvio Berlu...	neutral
551_24_4_...	But Fredrik Reinfeldt an...	Fredrik Rei...	neutral

Figure 18: Sample structure

3.4.4 Sentiment Retrieval

The sentiment retrieval with BLOOM faced several challenges. Program interruptions and crashes caused by usage rate limits, unexpected connection failures or poor result quality had to be addressed. The following sections outline these challenges and corresponding solutions.

3.4.4.1 Sentiment Retrieval Architecture

For sentiment retrieval, datasets in four languages supported by BLOOM were available, each containing 5,110 samples. Starting with prompt-engineering Strategy 2, 150 prompts were created per language. This resulted in a potential total of 150 x 5,110 x 4 = 3,066 million queries per strategy. However, Hugging Face's rate limit required a one-hour pause after approximately 1,500 queries. To keep the time required for prompt engineering within reasonable limits, the primary decision was to conduct prompt engineering using only one language. Furthermore, the number of samples was reduced to a single batch of 100 samples, which limited the time required to (150 x 100 / 1500) - 1 = 9 hours per strategy run, without taking into account the program runtime, unforeseen server-side program aborts, interruptions by the user and breaks for the analysis of problems and results.

Given that crashes and interruptions are likely during a nine-hour runtime, a robust architecture had to be developed for sentiment retrieval to minimize data loss and allow the program to resume rom the point where it stopped. Therefore, a nested structure was chosen:

- **Top Level:** Language of the samples.
- **Second Level:** Batches of 100 samples (or 99 when balanced).
- **Third Level:** Up to 150 prompts divided into “chunks” of 15 prompts each.
- **Fourth Level:** Each prompt added a query column to the batch dataset, storing specific queries created from samples and prompts.

- **Final Level:** Individual query processing.

For each level, a corresponding “processor” was implemented to manage its specific tasks. This nested structure is represented in the class diagram shown in Figure 19.

Figure 19: Sentiment retrieval architecture

The logic of the different components is illustrated in more detail in the activity diagrams in Appendix I.3.

3.4.4.2 Batch-Processing and Checkpoints

To minimize data loss from program crashes or user interruptions, queries were processed in batches of 99 balanced samples, with results serialized to disk at the end of each batch. Checkpoints were created per language, batch and query variant set (chunk), allowing the program to restart at a specific checkpoint if needed. Thus, in the event of a program abort, only a maximum of 98 queries had to be repeated. The program could also be manually restarted at any checkpoint, such as in cases where result files were mistakenly overwritten.

Given the time-consuming nature of the API queries, the number of batches per language was limited to one (containing 99 samples). For optimizing query variants, only one language – English – was used.

3.4.4.3 Program Interruptions

Different types of program interruptions were handled based on their specific causes, which fell into three categories: reaching rate limits, server-side interruptions or interruptions caused by the user.

3.4.4.3.1 Rate Limits

Initial automated experiments revealed that with the free Hugging Face API tier, only about 300 queries per hour were allowed before a rate limit imposed a one-hour wait. To handle this, the program was designed to enter a sleep state of slightly over 60 minutes when the limit was reached, resuming processing where it left off after the wait time had passed. Upgrading to a paid Hugging Face account increased the rate limit to approximately 1,500 queries per hour, though the one-hour wait time still applied.

3.4.4.3.2 Other Server-Side Interruptions

Any other errors, particularly connection drops, were addressed with shorter sleep times to allow system recovery. However, until the end of the tests, many errors persisted that could not be successfully intercepted because they occurred too infrequently to understand what caused them and how to intercept them, or because a repeated waiting period could not resolve them. As a result, the program required periodic manual restarts.

3.4.4.3.3 Interruptions by the User

User-side interruptions could occur for various reasons, including technical issues like power outages or computer crashes, or issues related to program flow and implementation. Some interruptions required a full program restart due to necessary implementation changes

discovered during testing. Other interruptions were anticipated and could be handled programmatically. For example, some prompts produced poor-quality results during retrieval, leading to two distinct cases:

- **Occasional invalid queries:** Some queries produced no valid responses for certain samples, meaning no extractable sentiment could be obtained from the LLM's answer. These cases were collected throughout the processing of a query column and displayed at the end (see Figure 20). The user could choose to save the results in the dataset's response column or discard them without terminating the program. If the user provided no input within one minute, the program defaulted to saving the data and resumed execution.

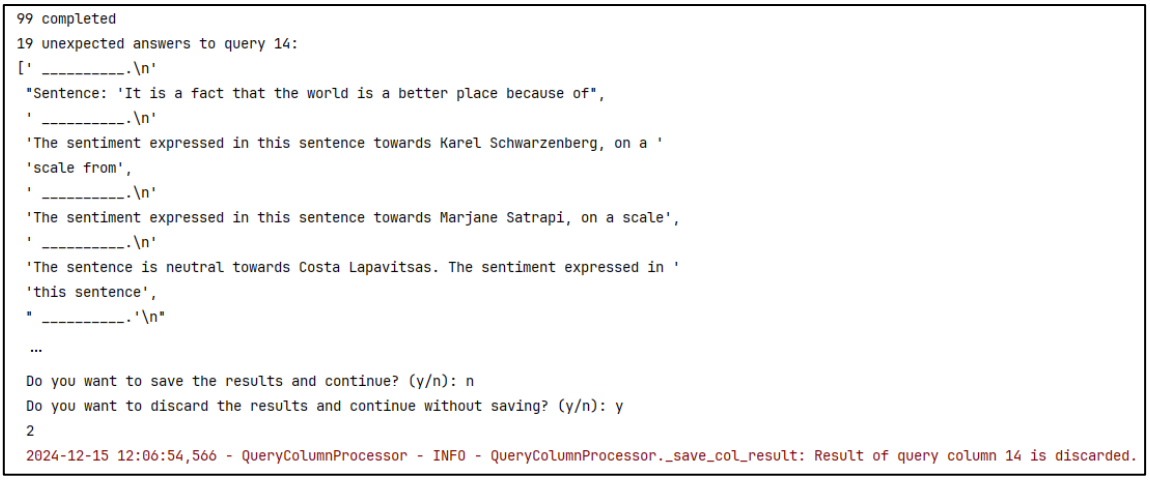


Figure 20: User dialog due to invalid responses for individual samples

- **Uniform query results:** If queries in the same query column consistently returned the same sentiment, indicating that the individual samples played no meaningful role in determining sentiment labels, further processing of the entire column became unnecessary. To detect such cases and terminate processing early, the program monitored sentiment diversity. If results remained identical after processing half of the samples, the program stopped processing the column as shown in Figure 21, and no response column was saved.

Figure 21: Query processing terminated after half of the samples

Similarly, if only two sentiment classes were present at the end of processing, the program skipped saving the results.

3.4.4.4 Processing the responses

Since BLOOM's responses are designed to continue a given input, they always included the original text of the sample. The following part, with which the text was continued, was typically longer than desired, even when BLOOM was explicitly instructed to provide a one-word response. Moreover, while the responses were not detailed enough to include an in-depth justification for the sentiment prediction – a potentially interesting area of study – they often exceeded the length limit imposed by BLOOM and therefore were truncated mid-sentence, as can also be seen in Figure 20. Continuation queries could have resolved this, but the focus was on obtaining concise one-word responses.

To address this, the prompts and the patterns for the composition of the queries were crafted as of the second prompt-engineering strategy to ensure whenever possible that BLOOM reliably included the desired answer (“positive,” “negative,” or “neutral”) in the first word of its response. The first word was then automatically extracted and the rest of the response discarded.

```
def _process_response(self, response: Dict[str, str]) \
        -> str:
    """ Extracts the predicted sentiment from the response provided by the LLM.... """

    answer = self._remove_input_data_from_response(response)

    predicted_sentiment = self._extract_sentiment_from_answer(answer)

    if not predicted_sentiment:
        self.failed_answers.append(answer)

    return predicted_sentiment


def _extract_sentiment_from_answer(self, answer: str) \
        -> str:
    """ Extracts the sentiment from the answer. """

    self.set_word_tokenizer('NoPunctuation')
    words = self.word_tokenizer.tokenize(answer)

    if is_none_or_empty(words):
        return ''

    predicted_sentiment = words[0]

    if predicted_sentiment not in ['negative', 'positive', 'neutral']:
        return ''

    return predicted_sentiment
```

Listing 5: Processing a response

3.4.5 Prompt Engineering

BLOOM was trained in four of the eight MAD-TSC languages: English, French, Spanish, and Portuguese (in descending order of training dataset size). Table 9 shows the size of the training datasets per language in bytes:

Language	Size in Bytes
EN	484,953,009,124
FR	208,242,620,434
ES	175,098,365,045
PT	79,277,543,375

Table 9: BLOOM: Training data sizes for EN, FR, ES and PT in bytes [77, 98]

Among these four languages, the English training dataset was more than twice the size of the second largest (French) and six times larger than the smallest (Portuguese). Given that model quality generally improves with larger training datasets, it was expected that BLOOM's sentiment analysis accuracy would be highest in English. Therefore, prompt engineering was conducted using English samples.

3.4.5.1 Placeholders

Based on the author's linguistic expertise, BLOOM's Hugging Face repository guidelines and preliminary manual experiments with the API, it was determined that a targeted sentiment query to BLOOM must consist of multiple components. These components include variable text elements and fixed elements such as the sample text and the mention of the sentiment target. The latter two are represented here using placeholders enclosed in square brackets: `[SENTENCE_NORMALISED]` and `[MENTION]`, referred to as sample and target placeholders, respectively.

The sample and target placeholders created various positions within the queries where variable text could be inserted. These positions are marked with placeholders enclosed in angle brackets, such as `<before_sentence>` or `<before_mention>`. The interaction between different position, sample, and target placeholders was determined by query patterns, which varied depending on the prompt-engineering strategy used. For example, the following pattern was used in prompt-engineering Strategy 1:

```
<before_sentence>[SENTENCE_NORMALISED]<before_mention>
[MENTION]<scale><question>
```

Figure 22: PromptEngineeringStrategy1 – Query pattern

Four position placeholders were defined. For each, specific values had to be assigned that could replace the placeholders in a concrete query. Typically, several values were considered suitable for each position placeholder. These values were organized into value lists associated with the position categories. The possible values for the position placeholders were defined as “prompt ingredients”, as shown in Listing 6 for prompt-engineering Strategy 1.

```
@property
def prompt_ingredients(self) -> PromptIngredientsType:
    return {
        'before_sentence':
            [
                'Sentence: '
            ],
        'before_mention':
            [
                ' The sentiment expressed in this sentence towards ',
            ],
        'scales':
            [
                ', on a scale from negative to neutral to positive, ',
                ', on a scale from positive to neutral to negative, '
            ],
        'questions':
            [
                'is',
                'is definitely',
                'most probably is',
                'most certainly is',
                'is rather',
                'tends to be',
                'is quite',
            ]
    }
```

Listing 6: PromptEngineeringStrategy1 - Possible values for position placeholders

3.4.5.2 Prompts and Queries

At this point in the Project, a terminological distinction was made between “prompt” and “query”. A “prompt” was defined to be the specific selection of individual values from the alternatives available for the defined positions according to the query pattern. A “query”, on the other hand, designed the concretely formulated request that was composed according to the defined pattern from the components or “parts” defined in the prompt and the respective concrete sample text and the sentiment target. This query could directly be sent to the Bloom API as part of the payload.

For example, applying the pattern `<before_sentence> [SENTENCE_NORMALISED] <before_mention> [MENTION] <scale><question>` to the prompt parts in

Listing 7 and the first sample from Figure 18 resulted in the query shown in Listing 8:

```
"1": {
        "before_sentence": "Sentence: ",
        "before_mention": "The sentiment expressed in this sentence towards ",
        "scale": ", on a scale from negative to neutral to positive, ",
        "question": "is"
    },
```

Listing 7: Prompt 1

```
Sentence: 'The answer, Finkelstein maintains, is that in the decades since Lennon's death society has changed radically.' The sentiment expressed in this sentence towards Lennon, on a scale from negative to neutral to positive, is
```

Listing 8: Query 1

From this point on, the terms “Prompt” and “Query” will be used according to these definitions.

3.4.5.3 Prompt Engineering Strategies

The development of suitable prompts followed the process outlined in Section 2.4 through several iterative cycles, leading to the creation of multiple prompt engineering strategies. To ensure these strategies could be revisited at a later stage of development, the prompt-engineering code from one cycle was not overwritten by the code of the next cycle. Instead, the strategy design pattern was used, allowing various strategies to remain available simultaneously and be interchangeable at runtime.

3.4.5.3.1 General Approach

A total of four different prompt strategies were developed. From the prompts generated by each strategy, queries were created for 99 balanced samples and sent to BLOOM. The returned responses were analyzed, and conclusions were drawn regarding optimization opportunities, which were then implemented in the subsequent strategy whenever possible.

To evaluate the results, the number of valid responses per prompt generated by each strategy was first analyzed. Subsequently, the prompts were evaluated using standard classification metrics: accuracy, precision, F1-score, and recall. Statistical evaluations were conducted, including minima, maxima, means, and standard deviations. These metrics were also used to rank the prompts according to their performance, as shown in Figure 23. The five best and five worst prompts were visualized in diagrams, with threshold markers for fair, good, very good,

and excellent results to facilitate recognition of individual prompt performance (see Figure 28 and Figure 29 in Section **Fehler! Verweisquelle konnte nicht gefunden werden.** below).

	acc	macro_f1	macro_recall	macro_precision
en_6	0.44444	0.29646	0.33333	0.36905
en_7	0.44444	0.37815	0.44444	0.61121
en_8	0.44444	0.37807	0.44444	0.55253
en_12	0.44444	0.26564	0.33333	0.54171
en_5	0.43434	0.28223	0.32576	0.43659
en_14	0.40404	0.25067	0.30303	0.21818
en_11	0.38384	0.20481	0.28788	0.42816
en_2	0.30303	0.24697	0.22727	0.47112
en_1	0.28283	0.21303	0.21212	0.21847

	acc	macro_f1	macro_recall	macro_precision
en_6	2.50000	3.00000	3.50000	7.00000
en_7	2.50000	1.00000	1.00000	1.00000
en_8	2.50000	2.00000	2.00000	2.00000
en_12	2.50000	5.00000	3.50000	3.00000
en_5	5.00000	4.00000	5.00000	5.00000
en_14	6.00000	6.00000	6.00000	9.00000
en_11	7.00000	9.00000	7.00000	6.00000
en_2	8.00000	7.00000	8.00000	4.00000
en_1	9.00000	8.00000	9.00000	8.00000

	rank
en_6	4.00000
en_7	1.37500
en_8	2.12500
en_12	3.50000
en_5	4.75000
en_14	6.75000
en_11	7.25000
en_2	6.75000
en_1	8.50000

Figure 23: Ranking of prompts by macro metrics

Each metric is used to rank the prompts based on their respective scores. These rankings are then averaged across the different metrics, resulting in an overall rank for each prompt.

The rankings were then used to infer which prompt ingredients contributed to better or worse results. To this end, correlations between the ingredients and the rankings were calculated. The resulting correlation matrix was visualized using heatmaps that displayed the relationships between different ingredient alternatives and rankings. Positive correlation values indicated a direct (unfavorable) relationship with the ranking scores, while negative values indicated an inverse (favorable) relationship. The higher the absolute value, the stronger the correlation. It is important to note that higher ranking numbers reflect worse performance—a rank of 1 (out of 9) is optimal, while a rank of 9 is the lowest. Therefore, positive correlations (marked in red) are undesirable, whereas negative correlations (blue) are beneficial. (An example is shown in Figure 24.) If an ingredient showed a strong positive correlation, it was considered for removal in the next prompt engineering strategy.

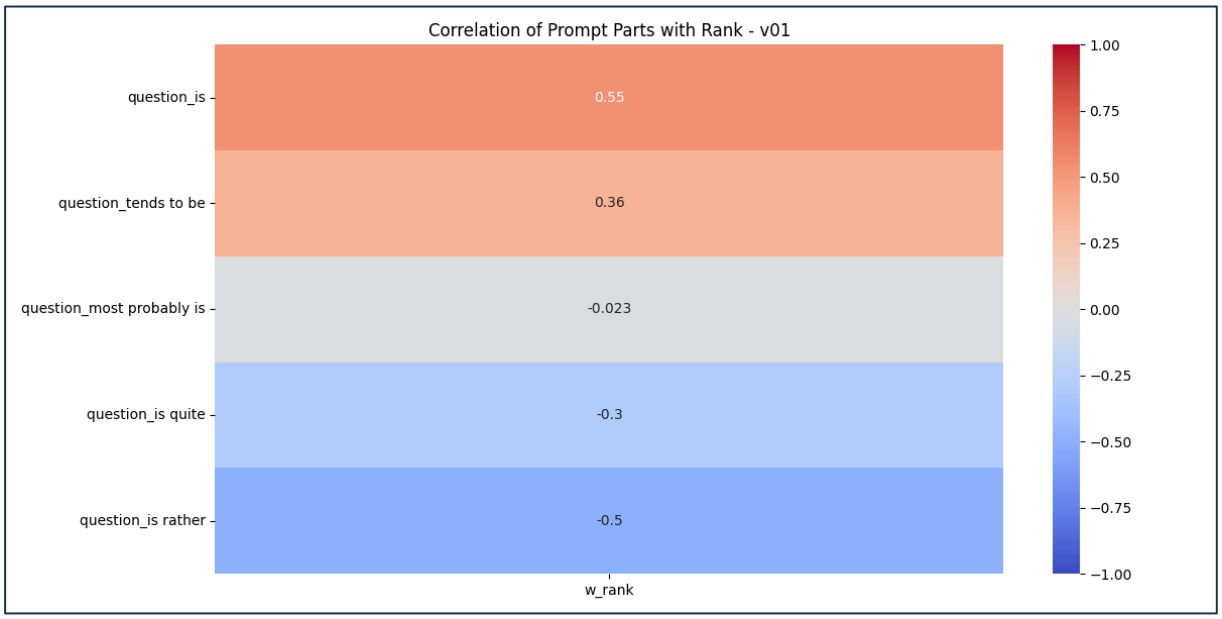


Figure 24: PromptEngineeringStrategy1 - Correlation of “questions” alternatives with the ranking

The heatmap suggests that using a simple “is” to form the question tends to perform worse than using combinations like “is quite” or “is rather”.

3.4.5.3.2 Prompt-Engenieering Strategy 1

Initially, in prompt-engineering Strategy 1, the prompt ingredients shown in Listing 6 were used based on preliminary manual experiments with the API. These ingredients were systematically combined to generate 14 prompts. The query pattern, placeholders, ingredients, and how the resulting queries were formed were described in Sections 3.4.5.1 und 3.4.5.2 to define concepts. Strategy 1 was primarily used to explore the potential for automating the querying process and extracting responses, which was then applied in all sentiment-retrieval runs. The processing of LLM responses and handling of valid versus less valid results were covered in Sections 3.4.4.3.3 and 3.4.4.4.

3.4.5.3.3 Prompt-Engineering Strategies 2 to 4

The comparison of ranked variants from prompt-engineering Strategy 1 revealed patterns that guided a more systematic generation of query variants in strategies 2 to 4. A new query pattern was introduced in Strategy 2 to give clearer and more targeted instructions to the LLM. The original four position categories were expanded to six:

```
<before_sentence>[SENTENCE_NORMALISED]<before_mention>
[MENTION]<scale><question><answer_before_mention>
[MENTION]<answer_start>
```

Figure 25: PromptEngineeringStrategy2 – Query pattern

Prompt ingredients were more finely defined within the PromptIngredientsMixin class, which was integrated into the PromptEngineeringStrategy base class. Instead of being tied to fixed positions in the query pattern, ingredients were now determined based on recurring string patterns. For example, the “targets” category (see Listing 9) was used in different parts of the prompt. The number of categories grew from 4 to 16, with each category offering between 1 (“preposition”) to 45 (“thoughts”) alternatives.

Due to the enormous number of potential combinations (ranging from hundreds of thousands to millions) the exhaustive approach used in Strategy 1 was no longer feasible. To keep things manageable, Strategy 2 and subsequent strategies limited the number of prompt variants to 150. To maximize variation, ingredients were randomly selected from different categories, in hopes that some combinations would perform exceptionally well and inform future optimizations. Only the first randomly chosen ingredient category of each prompt received prioritized treatment to introduce some systematic structure into the selection process – up to five randomly selected ingredients from that category were included, provided it had at least five elements.

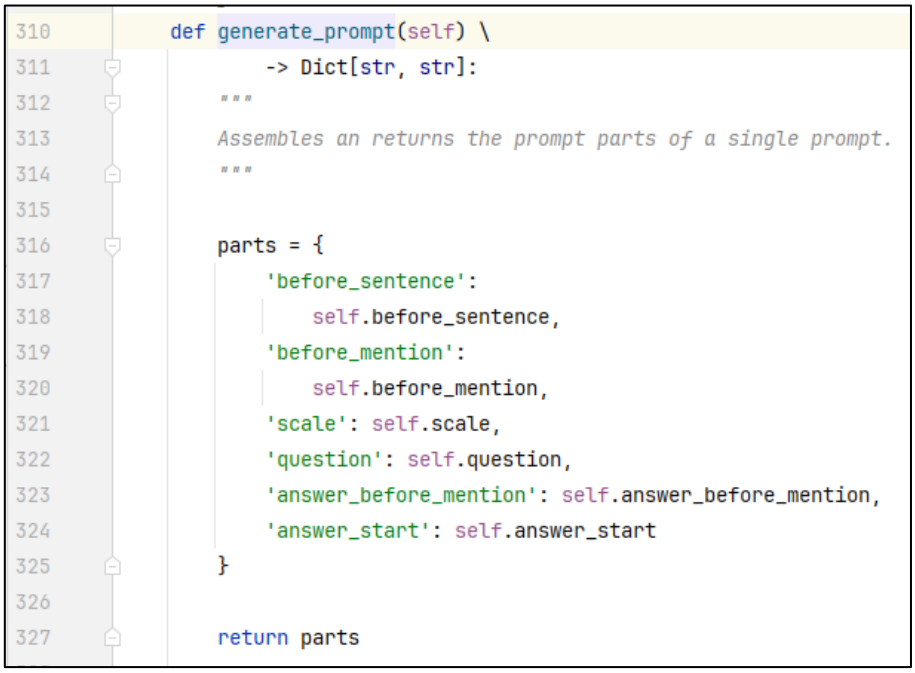
The selected ingredients were then checked to see if the same combination had been previously used. If so, the selection was discarded, and the next combination was generated. If not, the PromptGenerator was called to create the prompt from the selected ingredients. Instead of using the prompt ingredients directly, the PromptGenerator assembled various positional “prompt parts” in different ways from the finely defined prompt ingredients. Listing 10 shows the generation of the “before_sentence” prompt part, while Listing 11 illustrates the assembly of the six prompt parts of a complete prompt.

```
404             'targets':
405                 [
406                     'person',
407                     'individual',
408                     'target',
409                     'target individual',
410                     'target person',
411                 ],
```

Listing 9: PromptEngineeringStrategy2 – PromptIngredientsMixin: All ingredients / “targets” category

```
119     before_sentence = (
120             _dict['politeness'] +
121             _dict['task'] +
122             _dict['what'] +
123             _dict['preposition'] +
124             _dict['given'] +
125             _dict['where'] +
126             _dict['toward'] +
127             _dict['given'] +
128             _dict['target'] + ". \n" +
129             _dict['sentence_label'] +
130             (
131                 _dict['where'].rstrip().capitalize()
132                 if _dict['sentence_label'] == ""
133                 else _dict['where'].rstrip()
134             ) + ': '
135     )
```

Listing 10: PromptEngineeringStrategy2 – PromptGenerator: Generating the “before_sentence” prompt part from various prompt ingredients



```
310     def generate_prompt(self) \
311             -> Dict[str, str]:
312         """
313         Assembles an returns the prompt parts of a single prompt.
314         """
315
316         parts = {
317             'before_sentence':
318                 self.before_sentence,
319             'before_mention':
320                 self.before_mention,
321             'scale': self.scale,
322             'question': self.question,
323             'answer_before_mention': self.answer_before_mention,
324             'answer_start': self.answer_start
325         }
326
327         return parts
```

Listing 11: PromptEngineeringStrategy2 – PromptGenerator – Prompt generation from prompt parts

Since the automatic generation of random prompts did not guarantee their validity, each generated prompt was validated using BLOOM before being saved. In stategies 2 and 3, the first unbalanced English sample was used to generate a sentiment analysis query from the prompt. This query was then sent to BLOOM, and the response was checked to determine whether its processed result matched one of the three target sentiment classe – “negative”, “positive” or “neutral”. If it did, the prompt was considered valid and added to the collection of prompts.

As the proportion of invalid prompts increased to 74.7% in Strategy 3 (see Table 10), the number of samples used in pre-validation was increased to three in Strategy 4. Additionally, each of the three samples had to represent a different sentiment, and BLOOM was required to return the correct sentiment for all three. Invalid prompts were also stored in a separate file for later analysis.

This approach increased the proportion of valid prompts to 94.7, but at the cost of significantly longer pre-validation time – rising from just a few minutes to over six hours. A large part of this increase was likely due to the time required to store the invalid prompts. In total, Strategy 4 excluded and stored 6287 prompts as invalid.

Prompt strategy	Total prompts	Valid prompts	Invalid prompts	Valid prompts with invalid responses
1	14	9 (64.3%)	5 (35.7%)	7 (max: 37, min: 2)
2	150	59 (39.3%)	91 (60.7%)	6 (max: 46, min: 1)
3	150	38 (25.3%)	112 (74.7%)	6 (max: 6, min: 1)
4	150	142 (94.7%)	8 (5.3%)	7 (max:14, min: 1)

Table 10: Valid vs. invalid prompts

In all strategies, the results were examined for patterns that could explain why certain prompts performed better than others. A potential indicator was identified when consecutive prompts—due to the systematic element in an otherwise random prompt generation process—shared specific prompt parts and produced either particularly strong or particularly weak results, as was the case with prompts 17 through 21 in Strategy 2 (see Figure 26). However, such cases were observed only at the lower end of the performance spectrum across all strategy runs.

The shared components found in these prompts were then analyzed more closely to determine whether their ingredients appeared exclusively within that group of prompts (and possibly also in those excluded as invalid), or whether they were also part of other valid prompts. In the former case, they were excluded from subsequent strategies. In the latter case, the relevant ingredients were subjected to correlation analysis to determine whether they correlated strongly enough with poor performance to justify their exclusion from future prompt generation.

index	negative	neutral	positive	total
correct	33	33	33	99
en_11	72	1	26	99
en_22	75	11	13	99
en_23	77	18	4	99
en_24	81	10	8	99
en_25	78	13	8	99
en_26	79	13	7	99
en_27	24	6	69	99
en_38	15	1	83	99
en_42	63	7	29	99
en_44	65	16	18	99
en_45	71	2	26	99
en_46	71	4	24	99
en_55	16	1	82	99
en_57	42	39	18	99
en_58	43	40	16	99
en_64	61	29	9	99
en_65	54	16	29	99
en_68	12	68	19	99
en_69	79	16	4	99

index	negative	neutral	positive	total
en_71	70	25	4	99
en_72	59	36	4	99
en_73	79	13	7	99
en_78	46	4	49	99
en_79	59	1	39	99
en_85	49	1	49	99
en_86	87	1	11	99
en_87	81	5	13	99
en_98	4	91	4	99
en_99	47	40	12	99
en_100	82	7	10	99
en_101	74	21	4	99
en_102	70	22	7	99
en_103	70	20	9	99
en_104	50	48	1	99
en_106	59	38	2	99
en_107	74	24	1	99
en_108	55	42	2	99
en_110	84	1	14	99
en_111	63	7	29	99

index	negative	neutral	positive	total
en_119	7	32	60	99
en_120	11	33	55	99
en_121	10	25	64	99
en_122	9	36	54	99
en_123	10	15	74	99
en_126	70	28	1	99
en_127	53	45	1	99
en_131	35	29	35	99
en_136	93	1	5	99
en_140	46	49	4	99
en_141	46	48	5	99
en_142	40	55	4	99
en_143	47	42	10	99
en_144	55	38	6	99
en_76	48	11	39	98
en_20	45	4	28	77
en_21	42	5	24	71
en_18	35	9	26	70
en_19	34	8	24	66
en_17	25	11	17	53

Figure 26: PromptEngineeringStrategy2 – Sentiment class counts with highlights for the best prompts (red border) and the worst prompts (blue border)

In this way, ingredients such as “If I grasp it correctly” and “the specified” were identified in Strategy 2 as candidates for exclusion from the ingredient lists. Across all strategies, the correlation matrices of ingredients were also examined for additional instances of unusually high correlations with poor prompt rankings. However, most of the strongest correlations ranged between 20% and 40%, which was deemed too low to justify clear-cut decisions for exclusion.

However, these analyses only assessed ingredients present in ranked prompts, meaning that ingredients in prompts excluded as invalid during sentiment analysis were not considered. Yet the possibility that such ingredients were responsible for generating invalid queries remained an important factor in determining whether they should be removed from future prompt engineering strategies. To address this, an additional approach was implemented.

Valid and invalid prompts were compared to determine whether specific parts and ingredients consistently appeared only in valid prompts, only in invalid ones, or in both without any discernible pattern (see Appendix L). Simple frequency analyses were used to evaluate individual ingredients, while composite ingredients underwent a more detailed, fine-grained analysis. This involved identifying the underlying base ingredients and examining whether these might constitute “bad elements” – that is, whether they could be responsible for the poor performance of the prompts in which they appeared.

Ingredients and basic ingredients that were used at least five times and appeared exclusively in invalid prompts were ultimately excluded from the ingredient lists in the subsequent strategy. For instance, in Strategy 2, this process led to the identification and subsequent exclusion of various ingredients from the “Scales” and “Thoughts” categories.

3.4.5.3.4 Conclusion

The statistics and diagrams for the four prompt-engineering strategies described above (see Figure 27 - Figure 29) show that Strategy 2 achieved a significant improvement over Strategy 1—not only in average values but also in the best and worst prompts, where both an increase and a leveling of the metrics were observed. Strategy 3, by contrast, performed slightly worse, suggesting that simply eliminating underperforming prompt ingredients is not sufficient to improve overall performance. Strategy 4, on the other hand, delivered clearly better and, above all, more balanced results than Strategy 3, both in its best and worst prompts. Das Erfolgsrezept lag hier offensichtlich an der ausgefeilteren und umfangreicheren und damit zeitaufwendigeren Prävalidierung.

Among the four strategies, Strategy 4 performed the best overall. Unlike the others, it achieved the highest average scores and consistently strong results across all four evaluation metrics (values between 0.6 and 0.7) for its five best prompts. Even the five worst prompts in Strategy 4

did not fall below a score of 0.29. However, the single best-performing prompt was found in Strategy 2, with all metrics—except precision—slightly outperforming the best prompt in Strategy 4.

PromptEngineeringStrategy1

	acc	macro_f1	macro_recall	macro_precision
count	9.00000	9.00000	9.00000	9.00000
mean	0.39843	0.27956	0.32351	0.42745
std	0.06371	0.06307	0.08125	0.13934
min	0.28283	0.20481	0.21212	0.21818
25%	0.38384	0.24697	0.28788	0.36905
50%	0.43434	0.26564	0.32576	0.43659
75%	0.44444	0.29646	0.33333	0.54171
max	0.44444	0.37815	0.44444	0.61121

PromptEngineeringStrategy2

	acc	macro_f1	macro_recall	macro_precision
count	59.0	59.0	59.0	59.0
mean	0.45249	0.3896	0.44252	0.52713
std	0.06812	0.07645	0.08017	0.09596
min	0.28283	0.24478	0.21212	0.32976
25%	0.40909	0.34111	0.40404	0.46814
50%	0.44444	0.37689	0.44444	0.54838
75%	0.4899	0.43256	0.48485	0.59465
max	0.65657	0.65433	0.65657	0.70935

PromptEngineeringStrategy3

	acc	macro_f1	macro_recall	macro_precision
count	38.0	38.0	38.0	38.0
mean	0.488	0.4008	0.469	0.4931
std	0.0563	0.0783	0.0726	0.1199
min	0.3434	0.2037	0.3106	0.2628
25%	0.4646	0.3518	0.4268	0.3838
50%	0.4899	0.4125	0.4848	0.5154
75%	0.5253	0.4596	0.5253	0.5651
max	0.596	0.5347	0.596	0.7089

PromptEngineeringStrategy4

	acc	macro_f1	macro_recall	macro_precis…
count	142.0	142.0	142.0	142.0
mean	0.52234	0.48191	0.51629	0.5509
std	0.05845	0.07279	0.0669	0.0739
min	0.37374	0.29913	0.34091	0.33288
25%	0.47727	0.43178	0.47475	0.51827
50%	0.52525	0.48981	0.52525	0.55347
75%	0.56566	0.53538	0.56566	0.59509
max	0.64646	0.64549	0.64646	0.74603

Figure 27: Comparison of the four prompt-engineering strategies – Statistical overviews

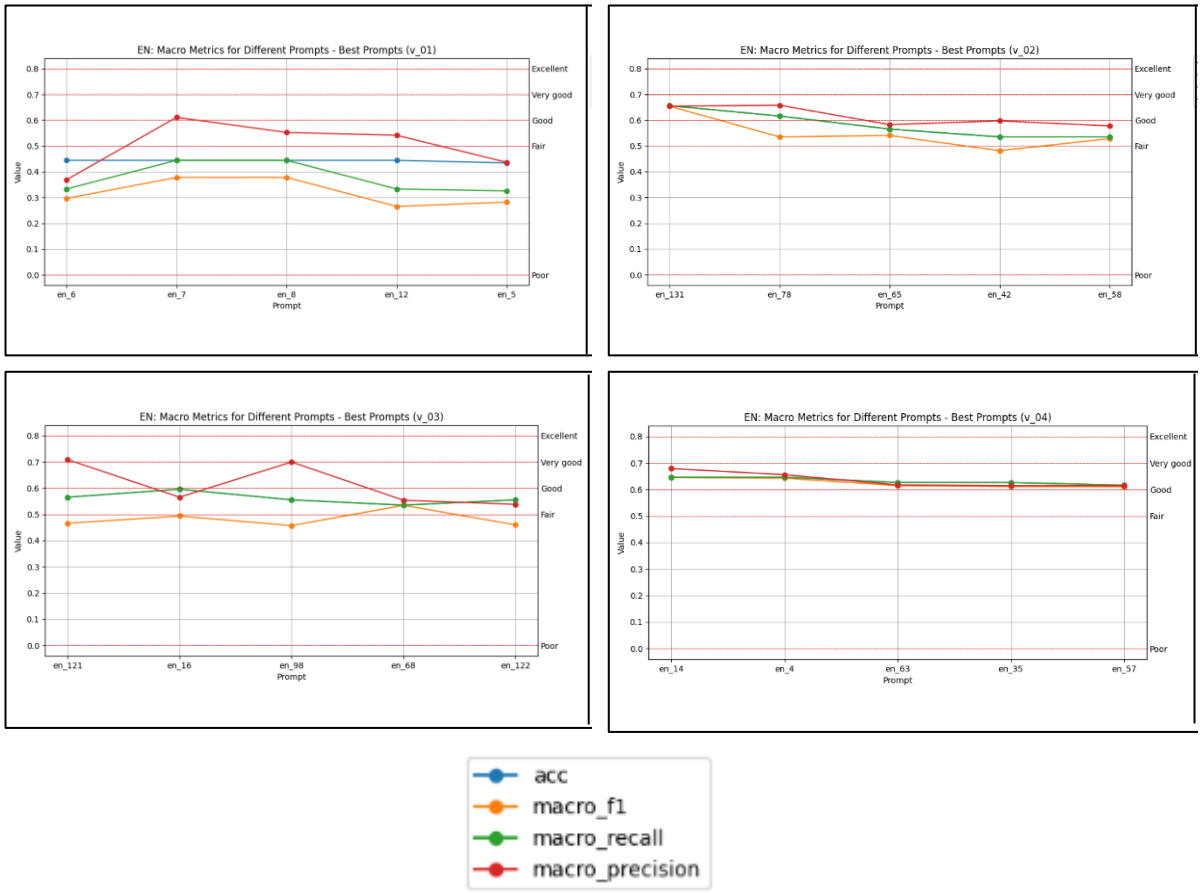


Figure 28: Comparison of the four prompt-engineering strategies – Best prompts

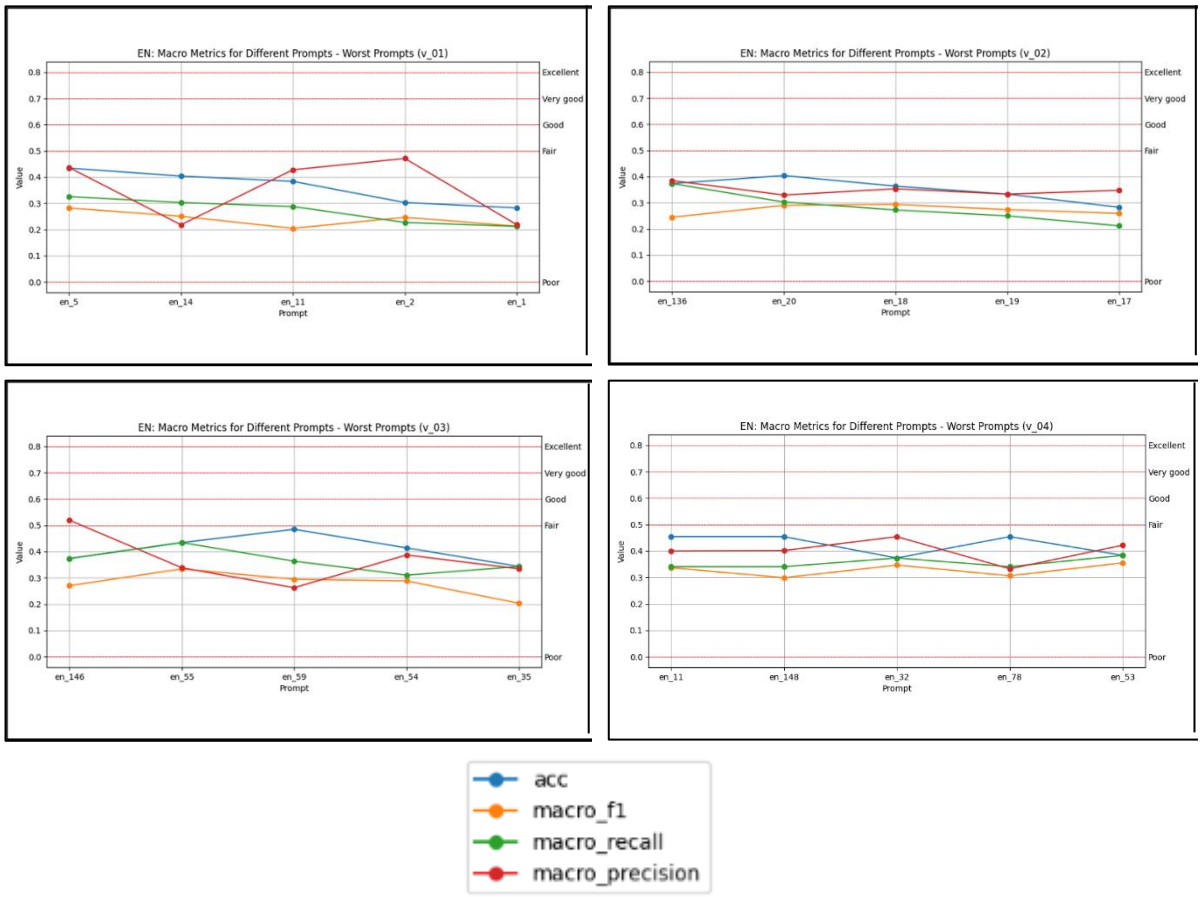


Figure 29: Comparison of the four prompt-engineering strategies – Worst prompts

Further improvements to the prompts using the previously applied strategies (e.g., eliminating unfavorable ingredients or adding new ones) no longer appeared viable, as no clear criteria for doing so could be established. Strengthening the pre-validation process also seemed unproductive, as this would have required correct results across more than just three random samples—drastically reducing the number of prompts considered valid and significantly increasing the number of prompts to be tested. This would, in turn, extend the pre-validation time, which had already exceeded six hours in Strategy 4. Additionally, the risk of overfitting prompts to the specific samples in the validation batch would have increased.

At this point, changing the query pattern appeared to be the more promising path forward. Both the order and the number and type of position categories could have been modified to develop additional strategies. However, such a change would have exceeded the project's time frame and must be reserved for future research.

Although Strategy 4 produced the best overall results among the prompt-engineering strategies, prompt #131 from Strategy 2 achieved the highest individual performance, with an

accuracy and recall of 0.65657, an F1-score of 0.65433, and a precision of 0.65452. Therefore, this prompt was selected for use in the subsequent multilingual sentiment analysis.

3.4.6 Multilingual Sentiment Analysis

For the multilingual sentiment analysis, the best-performing prompt from Strategy 2 – prompt #131 – was selected, since the prompts from Strategies 3 and 4 had yielded poorer results. However, during the implementation of the workflow, it was discovered that Hugging Face had unexpectedly discontinued support for BLOOM via the Serverless Inference API. As a result, it was no longer technically possible to perform or evaluate a multilingual sentiment analysis. The overall evaluation presented in the following section must therefore be limited to the assessment of monolingual results only.

3.5 Overall Evaluation

Due to the unexpected discontinuation of the BLOOM Serverless Inference API on Hugging Face, the originally planned cross-lingual overall evaluation could not be carried out. Nevertheless, the monolingual modeling and evaluation steps provided meaningful insights into the effectiveness of the developed single-language (English) sentiment analysis workflow for BLOOM.

Querying a generative language model like BLOOM via a natural language input interface proved to be fundamentally feasible over the course of the project, provided that stable API access was available. Although generative models tend to produce responses that exceed the expected output format for sentiment analysis, targeted prompt design and a subsequent extraction of sentiment answers made it possible to obtain largely reliable results.

The technical implementation of the automated querying process required robust batch processing, intermediate result storage, and the integration of checkpoints. These measures proved essential for ensuring that processing could be resumed in the event of interruptions—such as API timeouts or user-related issues.

The statistical evaluation of BLOOM’s responses to a balanced English-language subset of the MAD-TSC corpus yielded an average performance of over 65% across all standard metrics, compared to the gold standard annotations, following targeted prompt optimization.

A detailed discussion of the significance of these results, their limitations, and the resulting implications is provided in Kapitel 4.

3.6 Deployment

erfolgte im Code durch Docstrings im Numpy-Stil, der erlaubt, der eine nutzerfreundliche Art der Code-Dokumentation darstellt. Die Dokument

4 Discussion

The following section summarizes and discusses the main findings of this study.

4.1 Main findings

The main findings of this study correspond to the answers to the research questions defined in Section 1.3.

4.1.1 RQ 1

Research Question 1 addressed which publicly available multilingual sentiment-annotated news datasets can be used to research multilingual sentiment analysis with an LLM, , as well as the provision of the selected data for sentiment analysis within the project.

To answer the question regarding available datasets, selection criteria were developed in Section 2.1.2.1 for the required data collection. In line with the research question, the desired datasets needed to meet the following requirements: they had to be multilingual and sentiment-annotated, contain news texts, provide consistent annotation levels across languages, include sentiment annotations of standard quality, and retain the original news texts so they could be used as inputs for the LLM and for verification of sentiment labels. Additionally, the datasets had to include a sufficient and comparable number of samples across the different languages. The included languages were ideally aligned with the author’s language proficiency. In the case of target-dependent or aspect-based sentiment analysis, the datasets also needed to clearly define the target or aspect in a form suitable for use in an LLM prompt. Furthermore, based on the analysis of the client's business objectives, financial news texts were excluded from the pool of potential datasets.

A total of five multilingual news text datasets were identified in the literature and examined for compliance with the specified criteria. Only one dataset met all selection requirements: the MAD-TSC corpus by Dufraisse *et al.* [57]. After verifying its origin and ensuring that the license terms permitted use of the corpus for the purposes of this project, the dataset was downloaded from the authors’ GitHub repository, unzipped, and its file structure and completeness were analyzed. The required files were organized into eight language-specific folders within the project directory.

The original test, training, and validation data were merged and, according to the needs of the project, simplified on the one hand and supplemented with additional computed information on the other. The resulting data were saved in the project directory as PKL files for use in the study. Each PKL file thus contains 5,110 language-specific records.

The prepared data were then analyzed for data gaps and anomalies in text lengths (character and sentence length), sentiment annotations across languages, target labels, and the distribution balance of sentiment classes. Apart from an imbalanced distribution of sentiment classes, no issues were found. The imbalance was not corrected in the stored data so as not to distort the dataset. Instead, the sentiment analysis system was designed to allow the configuration of each sentiment analysis run to specify whether balanced or imbalanced data should be used. Upon completion of the data analysis, a ready-to-use dataset was thus available for further investigation.

4.1.2 RQ 2

Research Question 2 concerned the question of how an LLM can be used for automatic multilingual sentiment analysis of news texts without fine-tuning. Answering this question required selecting an LLM suitable for the intended investigation and developing a prototype capable of performing sentiment analysis with the chosen LLM.

First, selection criteria for the LLM were defined. The model needed to be a large generative language model – that is, it had to accept input in natural language and produce its output in the same form. Furthermore, it had to support the same languages – or at least three of the languages – present in the dataset used. The model's performance also needed to be high enough to handle the more nuanced sentiment analysis required for news texts. This meant the model had to be trained with a sufficiently large number of parameters and had to demonstrate strong results in relevant benchmarks.

The LLM also needed to offer an interface that allowed for automated querying, as this was essential to conducting the planned sentiment analysis. Moreover, it had to be accessible within the author's technical and financial constraints. Additionally, the client's security interests had to be taken into account – in case the client wanted to install and operate the model in their own environment. Therefore, the model needed to support on-premise deployment without transmitting sensitive data to third parties. Finally, a licensing model had to be available that would permit the client to use the model commercially.

The search for a generative large language model that met all these requirements took place in a rapidly evolving market, which continues to be highly dynamic. To gain an overview despite this, online comparison resources were consulted, and one was selected that focused

specifically on generative LLMs [76]. The goal was to select from among the well-known and popular LLMs listed in that overview one that best met the above criteria.

Based on the analysis of the client's business objectives, security concerns and licensing requirements were identified as the top priorities. The LLMs from the comparison list were therefore categorized according to whether they followed an open-source or closed-source approach. While the closed-source approach typically means that the model remains under the control of the provider – requiring all data to be transmitted to that provider – an open-source approach allows the integration and use of an LLM within the client's own infrastructure, meaning all data can be kept on-premise.

After excluding the closed-source models, the remaining candidates were evaluated for their suitability for commercial use. This excluded models such as LLaMA and its derivatives, which may only be used for research purposes.

Next, the criterion of supporting the eight languages included in the MAD-TSC corpus was applied. This eliminated the monolingual Dolly-LLM, leaving only BLOOM and various Falcon models. In terms of the number of training parameters, BLOOM was only comparable to the largest Falcon model, making the final selection a decision between these two. Language support did not serve as a decisive factor, since both models supported only four of the eight MAD-TSC languages – albeit different ones. The licenses of both models emphasized ethical usage but did not preclude ethical commercial use.

From a performance perspective, Falcon had a clear edge, achieving significantly better benchmark results than BLOOM. However, the decisive factors were ultimately technical and financial. The author lacked the resources necessary to install and operate Falcon or BLOOM in a private system. As a result, the only viable option for this study was to access one of the models via an API. At the time the decision was made, BLOOM was the only model available for use via a free API. Falcon required a paid inference endpoint, the usage costs of which could not be predicted – posing a financial risk the author could not assume.

This led to the decision to proceed with BLOOM. The development of a prototype sentiment analysis workflow involved two key components: implementing the automatic retrieval of sentiment information from BLOOM and identifying and optimizing suitable prompts.

The implementation of automated sentiment querying with BLOOM faced the significant challenge that sending large volumes of queries over the internet was subject to high latency and could take considerable time. During this process, there was a constant risk of connection interruptions, which could result in the loss of data already retrieved. This issue was addressed by sending queries in batches of 100 – or 99 samples when using balanced data – and saving the results after each successfully completed batch.

Since each batch could be tested against up to 150 prompts, which had to be managed within a data structure optimized for fast disk read/write operations, the prompts were divided into chunks of 15. Because each prompt had to be applied to 100 (or 99) samples in sequence, and each sample had to be sent as a separate query to BLOOM, a nested structure emerged: `Batch → Chunk → Query Column → Query`. When sentiment analysis was run for multiple languages simultaneously, the respective language was added as the outermost layer, resulting in the structure: `Language → Batch → Chunk → Query Column → Query`. Each level of this nested structure was assigned its own checkpoint, enabling the program to always track the most recently saved state. In the event of an interruption, processing could be resumed with minimal data loss – typically just a single Query Column of 99 or 100 queries might need to be repeated.

Various client-side interruptions were addressed using wait times and retry attempts to reduce the need for constant manual restarts. In the case of interruptions caused by rate limits imposed by Hugging Face – triggered after a certain number of processed queries – these wait times proved effective. However, due to the number and unpredictability of other interruptions and their causes, it was not always possible to identify what might prevent or mitigate a given failure. As a result, frequent manual restarts were still necessary, even though data loss was minimized thanks to the checkpointing and intermediate storage mechanisms.

The development of optimal prompts was approached through prompt engineering as an evolutionary prototyping process. This involved iterative cycles alternating between requirement definition, modeling, and implementation, followed by evaluation and subsequent revision of the requirements. Over the course of this process, four distinct prompt engineering prototypes were developed.

The first prototype was based on manual experimentation. After that, the search for optimal prompts became more systematic, using random combinations of predefined building blocks ("prompt ingredients"). Each of the Prototypes 2 through 4 generated various potential prompts, which were pre-validated by sending test queries to BLOOM. Up to 150 prompts that passed this pre-validation were then tested using the full set of 99 balanced samples.

The results were evaluated using standard metrics – accuracy, F1 score, recall, and precision. The prompts were ranked based on these scores, and the five best and five worst prompts from each prompt-engineering strategy were visualized in diagrams. The analysis sought to identify patterns in the results, examining the occurrence of prompt ingredients in valid versus invalid prompts, as well as correlations between specific ingredients and the performance of the corresponding prompts. The goal was to identify and exclude any elements that appeared to negatively impact performance. These efforts met with limited success.

However, changing the prompt structure from Strategy 1 to Strategy 2 did yield improvements, as did tightening the pre-validation rules in Strategy 4. Ultimately, a prompt was identified that achieved over 65% across all four metrics, indicating fairly strong performance. This prompt was discovered in Strategy 2, even though overall results from Strategy 4 were more consistent. Still, the best individual prompt from Strategy 2 outperformed the best prompt from Strategy 4 by a few tenths of a percentage point.

4.1.3 RQ 3

Research Question 3 addressed the monolingual and cross-lingual performance of the developed sentiment analysis system. Since multilingual sentiment analysis became impossible due to the unexpected discontinuation of support for the Serverless Inference API used to access BLOOM, no statements can be made here regarding cross-lingual performance.

The monolingual performance, however, has already been discussed in Section 4.1.2 above, as performance evaluation was an integral part of the evolutionary prototyping process.

4.2 Comparison with previous work

Compared to previous studies on sentiment analysis of news texts (see Appendix A), this study differs in its use of a large language model – BLOOM – without any domain-specific fine-tuning (a so-called zero-shot setting). The analysis was conducted exclusively through prompt engineering, without model adaptation, supervised learning, or reliance on sentiment lexicons, setting it apart from both traditional and most current approaches.

In comparison to earlier lexicon-based methods, such as those by Balahur *et al.* [33], which achieved precision and recall values between 71% and 75% for positive and negative sentiment classes in a binary classification setting, the best result in this study – an accuracy of approximately 65.7% and an F1-score of 65.4% for English-language news sentences – falls within a similar range. This is particularly notable given the greater difficulty of the three-class classification task and can be considered statistically significant.

Even when compared to Hamborg, Donnay, and Gipp [73], who—like this study—conducted target-dependent sentiment analysis, but using a domain-adapted LCF-BERT system with class-weighted cross-entropy loss, the results of the present analysis are comparable: the authors reported an accuracy of 66%, which is only slightly higher than that achieved in this study.

More recent supervised approaches, such as those by Kaya *et al.* [54] and Li *et al.* [70], report significantly higher performance metrics, with accuracies reaching up to 94.12%, thanks to

machine learning with domain-specific training. These methods benefit from domain-adapted datasets and model fine-tuning—elements that were deliberately not used in this study. Moreover, many of those studies are based on controlled or thematically restricted corpora.

In summary, while supervised and fine-tuned models still outperform zero-shot approaches using generative LLMs in terms of overall accuracy, the results presented here clearly demonstrate the practical viability of prompt-based sentiment analysis in zero-shot scenarios.

4.3 Implications

The monolingual performance achieved through prompt engineering in target-dependent sentiment analysis—without any prior fine-tuning of the model—clearly shows that generative LLMs trained with a sufficiently large number of parameters can, as hypothesized, develop a deep enough understanding of language to successfully handle the kind of subtle sentiment analysis required for news texts.

This is also encouraging for multilingual sentiment analysis, particularly since the author observed during experimentation that BLOOM was able to generate meaningful responses to German inputs, even though the LLM was not officially trained on that language. It is possible that the training of an LLM is designed to produce cross-lingual knowledge, allowing the model to transfer linguistic understanding from high-resource languages to lower-resource ones – thus filling gaps in training data for rarer languages, potentially even for languages not explicitly included in the training process at all.

4.4 Limitations

This thesis and the associated master's project were subject to various limitations that may have affected the reliability of the results:

One of the most serious constraints was the unexpected discontinuation of the serverless inference API for BLOOM on Hugging Face shortly before the multilingual part of the project was set to begin. This made it impossible to carry out the original plan of the project and study, effectively rendering the entire multilingual design of the work obsolete. As this occurred shortly before the submission deadline, there was no opportunity to revise the entire thesis accordingly. It is hoped, however, that the work at least provides a foundation for one or more future multilingual investigations.

The restriction of communication with BLOOM to the options provided by Hugging Face's inference API led to frequent connection interruptions and delays due to the need to observe rate-limit pauses. As a result, analysis of larger datasets was not feasible. Although the MAD-TSC corpus includes a total of 5,110 samples, the study could essentially only work with a

single batch of 100 or a balanced subset of 99 samples, which is far too small to draw statistically valid conclusions.

The inability to install the LLM on a local IT system or to use the Transformers library – due to a lack of resources – may have prevented the exploration of other, more standardized and efficient ways of using BLOOM for sentiment analysis. However, given that no evidence could be found in the documentation that BLOOM or the specific pipeline methods from the Transformers library already support procedures for the specific use case of target-dependent sentiment analysis, it is unclear whether such paths would have been productive.

The number of prompt engineering strategies developed in this thesis is quite small, so the results of the optimization – based mainly on heuristic assessments – cannot be considered representative in any way. It would be worthwhile to continue where this thesis leaves off and develop further strategies to determine whether 65% accuracy is the maximum achievable in target-dependent sentiment analysis with BLOOM and to possibly gain actionable insights into which factors are responsible for the generation of good or poor analysis results.

4.5 Further research

The preceding Section 4.4 already provides some indications of which questions this thesis leaves open or which aspects could be explored in more depth. In addition, the literature [126, 127] presents a relatively recent approach to optimizing prompt-driven queries to LLMs that appears promising: The use of chain-of-thought prompting could partially compensate for the lack of fine-tuning by activating step-by-step reasoning. This approach could potentially lead to a decisive improvement of prompt results.

Bibliography

[1] L. M. Rojas-Barahona, "Deep Learning for Sentiment Analysis," *Language and Linguistics Compass*, vol. 10, no. 12, pp. 701–719, 2016, doi: 10.1111/lnc3.12228.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *North American Chapter of the Association for Computational Linguistics*, 2019, pp. 4171–4186.

[3] B. Li, Y. Weng, Q. Song, B. Sun, and S. Li, "Continuing Pre-trained Model with Multiple Training Strategies for Emotional Classification," in *Proceedings of the 12th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis // The 12th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis - proceedings of the workshop: May 26, 2022 : WASSA 2022*, Dublin, Ireland, 2022, pp. 233–238. [Online]. Available: https://aclanthology.org/2022.wassa-1.22

[4] F. D. Souza, Oliveira, and J. B. de Souza Filho, "BERT for Sentiment Analysis: Pre-trained and Fine-Tuned Alternatives," *ArXiv*, abs/2201.03382, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:245837662

[5] A. K. Durairaj and A. Chinnalagu, "Transformer based Contextual Model for Sentiment Analysis of Customer Reviews: A Fine-tuned BERT," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 12, no. 11, 2021, doi: 10.14569/IJACSA.2021.0121153.

[6] M. Prytula, "Fine-tuning BERT, DistilBERT, XLM-RoBERTa and Ukr-RoBERTa Models for Sentiment Analysis of Ukrainian Language Reviews," *Artificial Intelligence*, vol. 29, AI.2024.29(2), pp. 85–97, 2024, doi: 10.15407/jai2024.02.085.

[7] K. Rajda, L. Augustyniak, P. Gramacki, M. Gruza, S. Woźniak, and T. Kajdanowicz, "Assessment of Massively Multilingual Sentiment Classifiers," in *Proceedings of the 12th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis // The 12th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis - proceedings of the workshop: May 26, 2022 : WASSA 2022*, Dublin, Ireland, 2022, pp. 125–140.

[8] N. Mughal, G. Mujtaba, S. Shaikh, A. Kumar, and S. M. Daudpota, "Comparative Analysis of Deep Natural Networks and Large Language Models for Aspect-Based Sentiment Analysis," *IEEE Access*, vol. 12, pp. 60943–60959, 2024, doi: 10.1109/ACCESS.2024.3386969.

[9] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Ilya Sutskever, "Language Models are Unsupervised Multitask Learners," in *OpenAI Blog*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:160025533

[10] S. Sagnika, A. Pattanaik, B. S. P. Mishra, and S. K. Meher, "A Review on Multi-Lingual Sentiment Analysis by Machine Learning Methods," *Journal of Engineering Science and Technology Review (JESTR)*, vol. 13, no. 2, pp. 154–166, 2020, doi: 10.25103/jestr.132.19.

[11] K. Dave, S. Lawrence, and D. M. Pennock, "Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews," in *The Twelfth International World Wide Web Conference: Proceedings: Budapest Convention Centre, 20-24 May 2003, Budapest, Hungary*, Budapest, Hungary, 2003, pp. 519–528.

[12] Y. Luo, D. Card, and D. Jurafsky, "Detecting Stance in Media on Global Warming," in *Findings of the Association for Computational Linguistics: EMNLP 2020 (The 2020 Conference on Empirical Methods in Natural Language Processing)*, 2020, pp. 3296–3315. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.296

[13] F. Bianchi, D. Nozza, and D. Hovy, "XLM-EMO: Multilingual Emotion Prediction in Social Media Text," in *Proceedings of the 12th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis // The 12th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis - proceedings of the workshop: May 26, 2022 : WASSA 2022*, Dublin, Ireland, 2022, pp. 195–203.

[14] B. Liu, *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions,* 2nd ed. Cambridge: University of Cambridge ESOL Examinations, 2020.

[15] C. Hutto and E. Gilbert, "VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text," *Proceedings of the Eighth International AAAI Conference on Web and Social Media (ICWSM)*, vol. 8, no. 1, pp. 216–225, 2014, doi: 10.1609/icwsm.v8i1.14550.

[16] A. Esuli and F. Sebastiani, "SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining," in *International Conference on Language Resources and Evaluation*, 2006, pp. 417–422.

[17] P. D. Turney, "Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews," in *Annual Meeting of the Association for Computational Linguistics*, 2002. [Online]. Available: https://api.semanticscholar.org/CorpusID:484335

[18] E. Cambria and B. White, "Jumping NLP Curves: A Review of Natural Language Processing Research [Review Article]," *IEEE Comput. Intell. Mag.*, vol. 9, no. 2, pp. 48–57, 2014, doi: 10.1109/MCI.2014.2307227.

[19] S. Sazzed, "Improving Sentiment Classification in Low-Resource Bengali Language Utilizing Cross-Lingual Self-supervised Learning," in *Natural Language Processing and Information Systems: 26th International Conference on Applications of Natural Language to Information Systems, NLDB 2021, Saarbrücken, Germany, June 23–25, 2021, Proceedings*, 2021, pp. 218–230.

[20] H. Ren, Y. Ren, X. Li, W. Feng, and M. Liu, "Natural Logic Inference for Emotion Detection," in *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annoted Big Data: 16th China National Conference, CCL 2017, and 5th International Symposium, NLP-NABD 2017, Nanjing, China, October 13-15, 2017 : proceedings*, 2017, pp. 424–436.

[21] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? Sentiment Classification Using Machine Learning Techniques," *Proceedings of the 2002 Conference on Empirical*

Methods in Natural Language Processing (EMNLP 2002), pp. 79–86, 2002, doi: 10.3115/1118693.1118704.

[22] Q. T. Ain *et al.,* "Sentiment Analysis Using Deep Learning Techniques: A Review," *International Journal of Advanced Computer Science and Applications*, vol. 8, 2017.

[23] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," in *International Conference on Learning Representations*, 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:5959482

[24] J. Pennington, *GloVe: Global Vectors for Word Representation.* [Online]. Available: https://nlp.stanford.edu/projects/glove/ (accessed: Dec. 9 2023).

[25] A. Vaswani *et al.,* "Attention is All You Need," *Advances in Neural Information Processing Systems*, vol. 30, 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[26] T. B. Brown *et al.,* "Language Models are Few-Shot Learners," *ArXiv*, abs/2005.14165, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:218971783

[27] A. Radford, Narasimhan, Karthik, Salimans, Tim, and I. Sutskever, *Improving Language Understanding with Unsupervised Learning.* [Online]. Available: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf (accessed: Oct. 22 2024).

[28] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos, "SemEval-2016 Task 5: Aspect Based Sentiment Analysis," in *International Workshop on Semantic Evaluation*, 2016, pp. 19–30. [Online]. Available: https://www.semanticscholar.org/paper/SemEval-2016-Task-5%3A-Aspect-Based-Sentiment-Pontiki-Galanis/3f8ed8a8d6eaea3480476a4869bf421feca381f8

[29] A. Balahur and M. Turchi, "Multilingual Sentiment Analysis Using Machine Translation?," in *WASSA@ACL*, 2012. [Online]. Available: https://api.semanticscholar.org/CorpusID:14292133

[30] A. Conneau *et al.,* "Unsupervised Cross-lingual Representation Learning at Scale," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 8440–8451. [Online]. Available: https://www.semanticscholar.org/paper/Unsupervised-Cross-lingual-Representation-Learning-Conneau-Khandelwal/6fec3e579c7cd4f13bdabbee2b6ac2e8ff5941c6

[31] T. Pires, E. Schlinger, and D. Garrette, "How Multilingual is Multilingual BERT?," *ArXiv*, abs/1906.01502, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:174798142

[32] M. Bednarek and H. Caple, *News discourse*. London: Bloomsbury Academic, 2019. [Online]. Available: https://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&db=nlabk&AN=1920988

[33] A. Balahur *et al.,* "Sentiment Analysis in the News," in *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, 2010, pp.

2216–2220. [Online]. Available: http://www.lrec-conf.org/proceedings/lrec2010/pdf/909_Paper.pdf

[34] Media Bias Fact Check, "Media Bias/Fact Check News," *Media Bias Fact Check*, 02 Mar., 2014. https://mediabiasfactcheck.com/ (accessed: Nov. 12 2024).

[35] C. Chan, H. Wessler, E. M. Rinke, K. Welbers, W. van Atteveldt, and S. Althaus, "How Combining Terrorism, Muslim, and Refugee Topics Drives Emotional Tone in Online News: A Six-Country Cross-cultural Sentiment Analysis," *International Journal of Communication*, vol. 14, pp. 3569–3594, 2020, doi: 568579.

[36] M. Honkanen and J. Müller, "Interjections and Emojis in Nigerian Online Communication," *World Englishes*, 2021, doi: 10.1111/weng.12544.

[37] E. Gehweiler, "Interjections and Expletives," in *Handbooks of pragmatics / eds. Wolfram Bublitz*, Vol. 8, *Historical Pragmatics*, A. H. Jucker, I. Taavitsainen, and W. Bublitz, Eds., Berlin: DE GRUYTER MOUTON, 2010, pp. 315–350.

[38] S. J. Warnett, "Hate Speech Detection in Heterogeneous and Multilingual Data," Wien, Fachhochschule Technikum, Höchstädtplatz 5, 1200 Wien, 2020.

[39] B. Felbo, A. Mislove, A. Søgaard, I. Rahwan, and S. Lehmann, "Using Millions of Emoji Occurrences to Learn Any-domain Representations for Detecting Sentiment, Emotion and Sarcasm," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, pp. 1615–1625.

[40] G. Brown and G. Yule, *Discourse Analysis,* 1st ed. Cambridge, New York: Cambridge University Press, 1983. [Online]. Available: http://ebooks.cambridge.org/ebook.jsf?bid=CBO9780511805226

[41] R. Mao, X. Li, M. Ge, and E. Cambria, "MetaPro: A Computational Metaphor Processing Model for Text Pre-Processing," *Information Fusion*, 86-87, pp. 30–43, 2022, doi: 10.1016/j.inffus.2022.06.002.

[42] H. Li, X. Cheng, K. Adson, T. Kirshboim, and F. Xu, "Annotating Opinions in German Political News," in *International Conference on Language Resources and Evaluation*, 2012. [Online]. Available: https://api.semanticscholar.org/CorpusID:15877270

[43] F. Hamborg, K. Donnay, and B. Gipp, "Automated Identification of Media Bias in News Articles: An Interdisciplinary Literature Review," *Int J Digit Libr*, vol. 20, no. 4, pp. 391–415, 2019, doi: 10.1007/s00799-018-0261-y.

[44] M. Bautin, L. Vijayarenu, and S. Skiena, "International Sentiment Analysis for News and Blogs," *ICWSM*, vol. 2, no. 1, pp. 19–26, 2008, doi: 10.1609/icwsm.v2i1.18606.

[45] M. d. P. Salas-Zárate, R. Valencia-García, A. Ruiz-Martínez, and R. Colomo-Palacios, "Feature-based Opinion Mining in Financial News: An Ontology-driven Approach," *Journal of Information Science*, vol. 43, no. 4, pp. 458–479, 2017, doi: 10.1177/0165551516645528.

[46] S. Krishnamoorthy, "Sentiment Analysis of Financial News Articles Using Performance Indicators," *Knowledge and information systems*, pp. 1–22, 2017, doi: 10.1007/s10115-017-1134-1.

[47] A. Agarwal, "Sentiment Analysis of Financial News," in *2020 12th International Conference on Computational Intelligence and Communication Networks: Proceedings : venue: Birla Institute of Applied Sciences, Bhimtal, Distt. Nainital, Uttarakhand, India*, Bhimtal, India, 2020, pp. 312–315.

[48] A. Moreno-Ortiz, J. Javier Fernández-Cruz, and C. P. Hernández, "Design and Evaluation of SentiEcon: A Fine-grained Economic/Financial Sentiment Lexicon from a Corpus of Business News // Design and Evaluation of SentiEcon: a Fine-grained Economic/Financial Sentiment Lexicon from a Corpus of Business News," in *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC2020)*, pp. 5065–5072. [Online]. Available: https://aclanthology.org/2020.lrec-1.623/ // https://api.semanticscholar.org/CorpusID:218974039

[49] S. Jain, M. Modi, and D. Gupta, "A Systematic Study on Sentiment Analysis Based on Text Mining and Deep Learning for Predictions in Stock Market Trends Through Social and News Media Data," *IJRASET*, vol. 9, no. 10, pp. 1589–1593, 2021, doi: 10.22214/ijraset.2021.38662.

[50] N. N. Sakhare and I. S. Shaik, "Spatial Federated Learning Approach for the Sentiment Analysis of Stock News Stored on Blockchain," *Spatial information research*, pp. 1–15, 2023, doi: 10.1007/s41324-023-00529-x.

[51] A. Sharaff, T. R. Chowdhury, and S. Bhandarkar, "LSTM Based Sentiment Analysis of Financial News," *SN Computer Science*, vol. 4, no. 5, p. 584, 2023, doi: 10.1007/s42979-023-02018-2.

[52] A. H. Shapiro, M. Sudhof, and D. J. Wilson, "Measuring News Sentiment," *Journal of Econometrics*, vol. 228, no. 2, pp. 221–243, 2022, doi: 10.1016/j.jeconom.2020.07.053.

[53] Y. Huang *et al.,* "A FinBERT Framework for Sentiment Analysis of Chinese Financial News," in *2024 4th International Symposium on Computer Technology and Information Science (ISCTIS)*, Xi'an, China, Jul. 2024 - Jul. 2024, pp. 796–799.

[54] M. Kaya, G. Fidan, and I. H. Toroslu, "Sentiment Analysis of Turkish Political News," in *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT 2012): Joint Conference] ; Macau, China, 4 - 7 December 2012 ; [Including Workshops ; Part of the 2012 World Intelligence Congress*, Macau, China, 2012, pp. 174–180.

[55] A. Pelicon, M. Pranjić, D. Miljković, B. Škrlj, and S. Pollak, "Zero-Shot Learning for Cross-Lingual News Sentiment Classification," *Applied Sciences*, vol. 10, no. 17, p. 5993, 2020, doi: 10.3390/app10175993.

[56] C. Mello, G. S. Cheema, and G. Thakkar, "Combining Sentiment Analysis Classifiers to Explore Multilingual News Articles Covering London 2012 and Rio 2016 Olympics," *Int J Digit Humanities*, vol. 5, no. 2, pp. 131–157, 2023, doi: 10.1007/s42803-022-00052-9.

[57] E. Dufraisse, A. Popescu, J. Tourille, A. Brun, and J. Deshayes, "MAD-TSC: A Multilingual Aligned News Dataset for Target-dependent Sentiment Classification," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long*

Papers), Toronto, Canada, 2023, pp. 8286–8305. Accessed: Apr. 24 2024. [Online]. Available: https://aclanthology.org/2023.acl-long.461/

[58] G. D. d. Arruda, N. T. Roman, and A. M. Monteiro, "An Annotated Corpus for Sentiment Analysis in Political News," (in pt), *0000-0000*, pp. 101–110, 2015. [Online]. Available: https://sol.sbc.org.br/index.php/stil/article/view/3970

[59] J. Bučar, M. Žnidaršič, and J. Povh, "Annotated News Corpora and a Lexicon for Sentiment Analysis in Slovene," *Lang Resources & Evaluation*, vol. 52, no. 3, pp. 895–919, 2018, doi: 10.1007/s10579-018-9413-3.

[60] A. Tamayo, J. Arias Londoño, D. Burgos, and G. Quiroz, "Sentiment Analysis of News Articles in Spanish Using Predicate Features," *Lenguaje*, vol. 47, no. 2, pp. 235–267, 2019, doi: 10.25100/lenguaje.v47i2.7937.

[61] O. Ring, M. K. Szabó, C. Guba, B. Váradi, and I. Üveges, ""Approaches to Sentiment Analysis of Hungarian Political News at the Sentence Level"," *Lang Resources & Evaluation*, 2024, doi: 10.1007/s10579-023-09717-5.

[62] T. O'Keefe, J. R. Curran, P. Ashwell, and I. Koprinska, "An Annotated Corpus of Quoted Opinions in News Articles," in *Annual Meeting of the Association for Computational Linguistics*, 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:7546721

[63] F. Hamborg and K. Donnay, "NewsMTSC: A Dataset for (Multi-)Target-dependent Sentiment Classification in Political News Articles," in *Conference of the European Chapter of the Association for Computational Linguistics*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:233189653

[64] H. Young, *The Digital Language Divide: How Does the Language You Speak Shape Your Experience of the Internet?* (accessed: Jul. 11 2022).

[65] D. Duval and F. Pétry, "L'analyse automatisée du ton médiatique : construction et utilisation de la version française du Lexicoder Sentiment Dictionary," *Can J Pol Sci*, vol. 49, no. 2, pp. 197–220, 2016, doi: 10.1017/S000842391600055X.

[66] C. Rauh, "Validating a Sentiment Dictionary for German Political Language—a Workbench Note," *Journal of Information Technology & Politics*, vol. 15, no. 4, pp. 319–343, 2018, doi: 10.1080/19331681.2018.1485608.

[67] C. Rauh, "Replication Data for: Validating a Sentiment Dictionary for German Political Language," 2018.

[68] M. Haselmayer and M. Jenny, "Sentiment Analysis of Political Communication: Combining a Dictionary Approach with Crowdcoding," *Quality & quantity*, vol. 51, no. 6, pp. 2623–2646, 2017, doi: 10.1007/s11135-016-0412-4.

[69] M. Haselmayer and M. Jenny, "The German Political Sentiment Dictionary (SUF edition)," 2020.

[70] J. Li, S. Fong, Y. Zhuang, and R. Khoury, "Hierarchical Classification in Text Mining for Sentiment Analysis of Online News," (in En;en), *Soft Comput*, vol. 20, no. 9, pp. 3411–3420, 2016, doi: 10.1007/s00500-015-1812-4.

[71] R. Seth and A. Sharaff, "Sentiment-Aware Detection Method of Fake News Based on Linguistic Fuzzy Bi-LSTM," in *OCIT 2023: 21st International Conference on Information*

Technology, 13-15 December 2023, Raipur, Chatisgarh, India : proceedings, Raipur, India, 2023, pp. 628–633.

[72] A. Aker *et al.,* "Corpus of News Articles Annotated with Article Level Sentiment," in *NewsIR@SIGIR*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:198189911

[73] F. Hamborg, K. Donnay, and B. Gipp, "Towards Target-Dependent Sentiment Classification in News Articles," in *Lecture Notes in Computer Science*, vol. 12646, *Diversity, Divergence, Dialogue: 16th International Conference, iConference 2021, Beijing, China, March 17-31, 2021 : Proceedings*, K. Toeppe, Ed., Cham: Springer, 2021, pp. 156–166.

[74] P. Přibáň, J. Šmíd, J. Steinberger, and A. Mištera, "A Comparative Study of Cross-lingual Sentiment Analysis," *Expert Systems with Applications*, vol. 247, 2024, doi: 10.1016/j.eswa.2024.123247.

[75] Peter Chapman, "CRISP-DM 1.0: Step-by-step Data Mining Guide," CRISP-DM Consortium, 2000. [Online]. Available: https://api.semanticscholar.org/CorpusID:59777418

[76] Alexander Thamm GmbH, *Die 14 wichtigsten großen Sprachmodelle: Ein umfassender Überblick.* [Online]. Available: https://www.alexanderthamm.com/de/blog/grosse-sprachmodelle-ein-ueberblick/ (accessed: May 10 2024).

[77] Botpress, *Die Zukunft der Spracherzeugung: Die 12 besten großen Sprachmodelle | Botpress Blog.* [Online]. Available: https://botpress.com/de/blog/the-best-large-language-models-available-today (accessed: May 10 2024).

[78] GDELT, *The GDELT Global Knowledge Graph (GKG) Data Format Codebook V2.1.* [Online]. Available: http://data.gdeltproject.org/documentation/GDELT-Global_Knowledge_Graph_Codebook-V2.1.pdf

[79] *The GDELT Project.* [Online]. Available: https://www.gdeltproject.org/ (accessed: Jan. 3 2024).

[80] GDELT, *Data: Querying, Analyzing and Downloading.* [Online]. Available: https://www.gdeltproject.org/data.html (accessed: Jan. 3 2024).

[81] L. Gatti, M. Guerini, and M. Turchi, "SentiWords: Deriving a High Precision and High Coverage Lexicon for Sentiment Analysis," *IEEE Trans. Affective Comput.*, vol. 7, no. 4, pp. 409–421, 2016, doi: 10.1109/TAFFC.2015.2476456.

[82] GDELT, *The GDELT Event Database Data Format Codebook V2.0.* [Online]. Available: http://data.gdeltproject.org/documentation/GDELT-Event_Codebook-V2.0.pdf (accessed: Jan. 4 2024).

[83] Ł. Augustyniak *et al.,* "Massively Multilingual Corpus of Sentiment Datasets and Multi-faceted Sentiment Classification Benchmark," Jun. 2023. [Online]. Available: http://arxiv.org/pdf/2306.07902.pdf

[84] R. Likert, "A Technique for the Measurement of Attitudes," *Archives of Psychology*, vol. 140, no. 1, pp. 5–55, 1932.

[85] Ł. Augustyniak *et al., mms_benchmark - MMS Dataset Citations.* [Online]. Available: https://brand24-ai.github.io/mms_benchmark/citations.html (accessed: Oct. 21 2024).

[86] *Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the Protection of Natural Persons with Regard to the Processing of Personal Data and on the Free Movement of such Data, and Repealing Directive 95/46/EC (General Data Protection Regulation)*, 2016. Accessed: Oct. 18 2024. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32016R0679&qid=1729243069750

[87] *Uredba (EU) 2016/679 Europskog parlamenta i Vijeća od 27. travnja 2016. o zaštiti pojedinaca u vezi s obradom osobnih podataka i o slobodnom kretanju takvih podataka te o stavljanju izvan snage Direktive 95/46/EZ (Opća uredba o zaštiti podataka)*, 2016. Accessed: Oct. 18 2024. [Online]. Available: https://eur-lex.europa.eu/legal-content/HR/TXT/HTML/?uri=CELEX:32016R0679&qid=1729243069750

[88] *Uredba (EU) 2016/679 Evropskega parlamenta in sveta z dne 27. aprila 2016 o varstvu posameznikov pri obdelavi osebnih podatkov in o prostem pretoku takih podatkov ter o razveljavitvi Direktive 95/46/ES (Splošna uredba o varstvu podatkov)*, 2016. Accessed: Oct. 18 2024. [Online]. Available: https://eur-lex.europa.eu/legal-content/SL/TXT/HTML/?uri=CELEX:32016R0679&qid=1729243069750

[89] M. Arcan, "A Comparison of Statistical and Neural Machine Translation for Slovene, Serbian and Croatian," in 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:159169201

[90] Sprachenlernen24.de, *Wissenswertes zur slowenischen Sprache.* [Online]. Available: https://www.sprachenlernen24.de/slowenisch-sprache/ (accessed: Oct. 18 2024).

[91] P. Malo, A. Sinha, P. Korhonen, J. Wallenius, and P. Takala, "Good Debt or Bad Debt: Detecting Semantic Orientations in Economic Texts," *Asso for Info Science & Tech*, vol. 65, no. 4, pp. 782–796, 2014, doi: 10.1002/asi.23062.

[92] M. Bastan, M. Koupaee, Y. Son, R. Sicoli, and N. Balasubramanian, "Author's Sentiment Prediction," in *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain (Online), pp. 604–615.

[93] Wikipedia, *Presseurop.* [Online]. Available: https://en.wikipedia.org/w/index.php?title=Presseurop&oldid=1269748105 (accessed: May 5 2025).

[94] Voxeurop, *Voxeurop - European journalism beyond borders.* [Online]. Available: https://voxeurop.eu/en/about-page/ (accessed: May 5 2025).

[95] Voxeurop, *Legal notice & privacy - Voxeurop.* [Online]. Available: https://voxeurop.eu/en/legal-notice-privacy/ (accessed: May 5 2025).

[96] Creative Commons, *CC BY-NC-SA 4.0 Deed - Attribution-NonCommercial-ShareAlike 4.0 International - Creative Commons.* [Online]. Available: https://creativecommons.org/licenses/by-nc-sa/4.0/deed.en (accessed: May 5 2025).

[97] Creative Commons, *CC BY-NC-ND 4.0 Deed | Attribution-NonCommercial-NoDerivs 4.0 International | Creative Commons.* [Online]. Available: https://creativecommons.org/licenses/by-nc-nd/4.0/deed.en (accessed: Apr. 9 2024).

[98] T. Honroth, J. Siebert, and P. Kelbert, *Open Source Large Language Models selbst betreiben - Blog des Fraunhofer IESE.* [Online]. Available: https://www.iese.fraunhofer.de/blog/open-source-large-language-models-selbst-betreiben/ (accessed: May 14 2024).

[99] European Court of Justice, *Judgment of the Court (Grand Chamber) of 16 July 2020 (Case C-311/18).* Accessed: Oct. 28 2024. [Online]. Available: https://eur-lex.europa.eu/legal-content/en/TXT/?uri=CELEX%3A62018CJ0311

[100] H. Touvron *et al.,* "LLaMA: Open and Efficient Foundation Language Models," Meta AI, Feb. 2023. [Online]. Available: http://arxiv.org/pdf/2302.13971

[101] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient Finetuning of Quantized LLMs," May. 2023. [Online]. Available: http://arxiv.org/pdf/2305.14314

[102] S. Mukherjee, A. Mitra, G. Jawahar, S. Agarwal, H. Palangi, and A. Awadallah, "Orca: Progressive Learning from Complex Explanation Traces of GPT-4," Jun. 2023. [Online]. Available: http://arxiv.org/pdf/2306.02707

[103] L. Zheng *et al.,* "Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena," Jun. 2023. [Online]. Available: http://arxiv.org/pdf/2306.05685

[104] Technology Innovation Institute, *Falcon-7b.* [Online]. Available: https://huggingface.co/tiiuae/falcon-7b (accessed: Oct. 28 2024).

[105] Technology Innovation Institute, *Falcon-40b.* [Online]. Available: https://huggingface.co/tiiuae/falcon-40b (accessed: Oct. 28 2024).

[106] Technology Innovation Institute, *Falcon-180B.* [Online]. Available: https://huggingface.co/tiiuae/falcon-180B (accessed: Oct. 28 2024).

[107] T. Le Scao *et al.,* "BLOOM: A 176B-Parameter Open-Access Multilingual Language Model," Nov. 2022. [Online]. Available: http://arxiv.org/pdf/2211.05100

[108] D. Hendrycks *et al.,* "Measuring Massive Multitask Language Understanding," *ArXiv*, abs/2009.03300, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:221516475

[109] Papers with Code, *MMLU Benchmark (Multi-task Language Understanding).* [Online]. Available: https://paperswithcode.com/sota/multi-task-language-understanding-on-mmlu (accessed: Nov. 2 2024).

[110] E. Almazrouei *et al.,* "The Falcon Series of Open Language Models," 2023.

[111] S. Wu *et al.,* "BloombergGPT: A Large Language Model for Finance," 2023.

[112] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, "HellaSwag: Can a Machine Really Finish Your Sentence?," 2019.

[113] Papers with Code, *HellaSwag Benchmark (Sentence Completion).* [Online]. Available: https://paperswithcode.com/sota/sentence-completion-on-hellaswag (accessed: Nov. 2 2024).

[114] Technology Innovation Institute, *Falcon 180B TII License 1.0.* [Online]. Available: https://huggingface.co/spaces/tiiuae/falcon-180b-license/blob/main/ACCEPTABLE_USE_POLICY.txt (accessed: Nov. 1 2024).

[115] BigScience, *BigScience RAIL License v1.0.* [Online]. Available: https://huggingface.co/spaces/bigscience/license (accessed: May 15 2024).

[116] Intel Corporation, *Intel-extension-for-pytorch: A Python Package for Extending the Official PyTorch that Can Easily Obtain Performance on Intel Platform.* [Online]. Available: https://github.com/intel/intel-extension-for-pytorch (accessed: Nov. 3 2024).

[117] NVIDIA Developer, *CUDA Toolkit - Free Tools and Training.* [Online]. Available: https://developer.nvidia.com/cuda-toolkit (accessed: Nov. 3 2024).

[118] Hugging Face, *Turn AI Models into APIs.* [Online]. Available: https://huggingface.co/inference-endpoints (accessed: Nov. 4 2024).

[119] Hugging Face, *Rate Limits.* [Online]. Available: https://huggingface.co/docs/api-inference/rate-limits#rate-limits (accessed: Nov. 3 2024).

[120] Hugging Face, *Serverless Inference API.* [Online]. Available: https://huggingface.co/docs/api-inference/index (accessed: Nov. 3 2024).

[121] Hugging Face, *User Access Tokens.* [Online]. Available: https://huggingface.co/docs/hub/en/security-tokens#what-are-user-access-tokens (accessed: Nov. 3 2024).

[122] E. W. Dijkstra, "On the Role of Scientific Thought," in *Texts and monographs in computer science, Selected Writings on Computing: A personal Perspective*, E. W. Dijkstra, Ed., New York, NY: Springer, 1982, pp. 60–66.

[123] P. A. Laplante and M. Kassab, *What every engineer should know about software engineering*. Boca Raton, FL // Boca Raton: CRC Press Taylor & Francis; CRC Press, 2007 // 2023. [Online]. Available: http://www.loc.gov/catdir/enhancements/fy0703/2006036497-d.html

[124] D. L. Parnas, "On the criteria to be used in decomposing systems into modules," *Commun. ACM*, vol. 15, no. 12, pp. 1053–1058, 1972, doi: 10.1145/361598.361623.

[125] R. C. Martin, *Agile software development, principles, patterns, and practices*. Harlow: Pearson, 2014. [Online]. Available: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=5833208

[126] J. Wei *et al.,* "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," 2022.

[127] H. Fei, B. Li, Q. Liu, L. Bing, F. Li, and T.-S. Chua, "Reasoning Implicit Sentiment with Chain-of-Thought Prompting," 2023.

[128] A. Graves and J. Schmidhuber, "Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures," *Neural networks : the official journal of the International Neural Network Society*, vol. 18, 5-6, pp. 602–610, 2005, doi: 10.1016/j.neunet.2005.06.042.

[129] B. Zeng, H. Yang, R. Xu, W. Zhou, and X. Han, "LCF: A Local Context Focus Mechanism for Aspect-Based Sentiment Classification," *Applied Sciences*, vol. 9, no. 16, p. 3389, 2019, doi: 10.3390/app9163389.

[130] S. Hochreiter and J. Schmidhuber, "Long Short-term Memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[131] A. McCallum and K. Nigam, "A Comparison of Event Models for Naive Bayes Text Classification," in *AAAI Conference on Artificial Intelligence*, 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:7311285

[132] Responsible AI Licensing, *Responsible AI Licenses (RAIL).* [Online]. Available: https://www.licenses.ai/ (accessed: Dec. 2 2024).
[133] Y. Liu *et al.,* "RoBERTa: A Robustly Optimized BERT Pretraining Approach," 2019.
[134] W. Peterson, T. Birdsall, and W. Fox, "The Theory of Signal Detectability," *Trans. IRE Prof. Group Inf. Theory*, vol. 4, no. 4, pp. 171–212, 1954, doi: 10.1109/TIT.1954.1057460.
[135] P. Pudil, J. Novovicová, and J. Kittler, "Floating Search Methods in Feature Selection," *Pattern Recognit. Letters*, vol. 15, pp. 1119–1125, 1994. [Online]. Available: https://api.semanticscholar.org/CorpusID:270333833
[136] C. Cortes and V. Vapnik, "Support-vector Networks," *Mach Learn*, vol. 20, no. 3, pp. 273–297, 1995, doi: 10.1007/BF00994018.
[137] Luhn. Hans Peter, "A Statistical Approach to Mechanized Encoding and Searching of Literary Information," *IBM J. Res. Dev.*, vol. 1, pp. 309–317, 1957. [Online]. Available: https://api.semanticscholar.org/CorpusID:15879823
[138] K. Spärck Jones, "A Statistical Interpretation of Term Specificity and its Application in Retrieval," *Journal of Documentation*, vol. 60, no. 5, pp. 493–502, 2004, doi: 10.1108/00220410410560573.
[139] G. Lample and A. Conneau, "Cross-lingual Language Model Pretraining," in *Neural Information Processing Systems*, 2019. [Online]. Available: https://www.semanticscholar.org/paper/Cross-lingual-Language-Model-Pretraining-Lample-Conneau/ec4eba83f6b3266d9ae7cabb2b2cb1518f727edc
[140] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/681a7f9d-17a4-800e-a215-3a869f2b3ae0 (accessed: May 6 2025).
[141] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68178d64-8eec-800e-a927-afb3d9e33b9e (accessed: Nov. 22 2024).
[142] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68178f76-f4a4-800e-bfc2-ec07f2346695 (accessed: Nov. 25 2024).
[143] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68178ac6-48dc-800e-83e4-c78859380bc3 (accessed: Nov. 9 2024).
[144] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/6817890a-d4d8-800e-bc18-c6a38caf954f (accessed: Oct. 29 2024).
[145] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/681789d8-16ac-800e-98b6-0b3839e54ec9 (accessed: Nov. 4 2024).
[146] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68179684-2840-800e-a6a1-fd2667f358b3 (accessed: Dec. 15 2024).
[147] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68179851-2138-800e-8cfe-c1de4a525f1c (accessed: Dec. 18 2024).
[148] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68179556-64c4-800e-890a-ba23951c6396 (accessed: Dec. 2 2024).
[149] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/681866b9-beec-800e-89f9-00d597ae4a33 (accessed: Dec. 20 2024).

[150] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/6817948d-4834-800e-9691-188cfc4f257a (accessed: Nov. 30 2024).
[151] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68178a48-1cbc-800e-ba44-55b2f65f2cc3 (accessed: Nov. 6 2024).
[152] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/681794de-ffbc-800e-8b76-bb59200b8458 (accessed: Dec. 1 2024).
[153] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/6817924f-ea74-800e-b41a-236916557dac (accessed: Nov. 30 2024).
[154] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68178ce0-c9d8-800e-99f9-0f25e1a0acb7 (accessed: Nov. 13 2024).
[155] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68179168-808c-800e-a6a3-1db98f2d4eb6 (accessed: Nov. 27 2024).
[156] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68179602-ea14-800e-bb64-7964e93518d8 (accessed: Dec. 3 2024).
[157] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68178967-954c-800e-82a7-c96ca4cc983a (accessed: Nov. 4 2024).
[158] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68178ddc-f500-800e-a818-6af37e345b0a (accessed: Nov. 23 2024).
[159] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/6817863f-8c44-800e-bc3b-4be828ea22a6 (accessed: Oct. 1 2024).
[160] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/681797a1-3bd0-800e-8641-18066b6d8524 (accessed: Dec. 17 2024).
[161] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68179925-3be8-800e-a558-aa288eaefd57 (accessed: Dec. 19 2024).
[162] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/681866b9-beec-800e-89f9-00d597ae4a33 (accessed: Dec. 20 2024).
[163] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/6818685b-6f18-800e-bc3a-0b5c9895e615 (accessed: May 5 2025).
[164] OpenAI, *ChatGPT.* [Online]. Available: https://chatgpt.com/share/68186767-75bc-800e-a740-7a23fe458d49 (accessed: May 1 2025).
[165] DeepL, *DeepL.* [Online]. Available: https://www.deepl.com/ (accessed: May 5 2025).
[166] *ISO 639:2023(en) Code for Individual Languages and Language Groups,* International Organization for Standardization.
[167] L. Lloyd, D. Kechagias, and S. Skiena, "Lydia: A System for Large-Scale News Analysis: (Extended Abstract," in *Lecture Notes in Computer Science*, vol. 3772, *String Processing and Information Retrieval: 12th International Conference, SPIRE 2005, Buenos Aires, Argentina, November 2 - 4, 2005 ; Proceedings*, M. Consens, Ed., Berlin, Heidelberg: Springer, 2005, pp. 161–166.
[168] A. Balahur, R. Steinberger, E. van der Goot, B. Pouliquen, and M. Kabadjov, "Opinion Mining on Newspaper Quotations," in *2009 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT): WI-IAT*

2009 ; Milano, Italy, 15 - 18 September 2009 ; [proceedings ; including workshop papers, Milan, Italy, 2009, pp. 523–526.

[169] C. Strapparava and A. Valitutti, "WordNet Affect: an Affective Extension of WordNet," in *International Conference on Language Resources and Evaluation*, 2004. [Online]. Available: https://api.semanticscholar.org/CorpusID:38166371

[170] S. Cerini, V. Compagnoni, A. Demontis, M. Formentelli, and C. Gandini, "Micro-WNOp: A Gold Standard for the Evaluation of Automatically Compiled Lexical Resources for Opinion Mining," in *Materiali linguistici*, vol. 59, *Language Resources and Linguistic Theory*, A. Sansò, Ed., Milano: Angeli, 2007, 200-210.

[171] W. Xiong, J. Xu, and M. Liang, "An Architecture for Automatic Opinion Classification in Western Online News," in *2014 IEEE Workshop on Electronics, Computer and Applications (IWECA 2014): Ottawa, Ontario, Canada, 8 - 9 May 2014*, Ottawa, ON, Canada, 2014, pp. 717–721.

[172] M. U. Islam, F. B. Ashraf, A. I. Abir, and M. A. Mottalib, "Polarity Detection of Online News Articles Based on Sentence Structure and Dynamic Dictionary," in *ICCIT : 2017 20th International Conference of Computer and Information Technology : 22-24 December 2017*, Dhaka, 2017, pp. 1–5.

[173] S. Taj, B. B. Shaikh, and A. F. Meghji, "Sentiment Analysis of News Articles: A Lexicon based Approach," in *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, 2019.

[174] K. Ghag and K. Shah, "SentiTFIDF – Sentiment Classification Using Relative Term Frequency Inverse Document Frequency," *IJACSA*, vol. 5, 2014. [Online]. Available: https://api.semanticscholar.org/CorpusID:17991530

[175] G. A. Miller, "WordNet: A Lexical Database for English," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995, doi: 10.1145/219717.219748.

[176] S. Baccianella, A. Esuli, and F. Sebastiani, "SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining," in *International Conference on Language Resources and Evaluation*, 2010. [Online]. Available: https://api.semanticscholar.org/CorpusID:13886408

[177] C. Chan and E. M. Rinke, "Online Appendix to the Article "How Combining Terrorism, Muslim, and Refugee Topics Drives Emotional Tone in Online News: A Six-Country Cross-Cultural Sentiment Analysis"," 2022.

[178] S. A. Awar, "Sentiment Analysis on News Headlines: Classic Supervised Learning vs Deep Learning Approach," *Towards Data Science*, 20 Mar., 2022. https://towardsdatascience.com/sentiment-analysis-on-news-headlines-classic-supervised-learning-vs-deep-learning-approach-831ac698e276 (accessed: May 6 2024).

[179] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: Rapid Training Data Creation with Weak Supervision," *The VLDB journal : very large data bases : a publication of the VLDB Endowment*, vol. 29, no. 2, pp. 709–730, 2020, doi: 10.1007/s00778-019-00552-1.

[180] S. Kaliappan, L. Natrayan, and A. Rajput, "Sentiment Analysis of News Headlines Based on Sentiment Lexicon and Deep Learning," in *Proceedings of the 4th International Conference on Smart Electronics and Communication (ICOSEC 2023) : 20-22, September 2023, Trichy, India*, Trichy, India, 2023, pp. 1044–1047.

[181] Creative Commons, *CC BY-NC 4.0 Deed | Attribution-NonCommercial 4.0 International | Creative Commons.* [Online]. Available: https://creativecommons.org/licenses/by-nc/4.0/ (accessed: Apr. 8 2024).

[182] Creative Commons, *CC BY-SA 4.0 Deed | Attribution-ShareAlike 4.0 International | Creative Commons.* [Online]. Available: https://creativecommons.org/licenses/by-sa/4.0/ (accessed: Feb. 12 2024).

[183] Creative Commons, *CC BY-NC-SA 3.0 Deed | Attribution-NonCommercial-ShareAlike 3.0 Unported | Creative Commons.* [Online]. Available: https://creativecommons.org/licenses/by-nc-sa/3.0/ (accessed: Feb. 12 2024).

[184] Creative Commons, *CC BY 4.0 Deed | Attribution 4.0 International | Creative Commons.* [Online]. Available: https://creativecommons.org/licenses/by/4.0/ (accessed: Feb. 12 2024).

[185] GNU, *The GNU General Public License v3.0 - GNU Project - Free Software Foundation.* [Online]. Available: https://www.gnu.org/licenses/gpl-3.0.html.en (accessed: Apr. 10 2024).

[186] Anthropic, *Introducing the next generation of Claude.* [Online]. Available: https://www.anthropic.com/news/claude-3-family (accessed: May 21 2024).

[187] Y. Bai *et al.,* "Constitutional AI: Harmlessness from AI Feedback," *ArXiv*, abs/2212.08073, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:254823489

[188] A. N. Gomez, *Introducing Command R+: A Scalable LLM Built for Business.* [Online]. Available: https://cohere.com/blog/command-r-plus-microsoft-azure (accessed: Oct. 27 2024).

[189] OpenAI *et al.,* "GPT-4 Technical Report," Mar. 2023. [Online]. Available: http://arxiv.org/pdf/2303.08774

[190] R. Thoppilan *et al.,* "LaMDA: Language Models for Dialog Applications," 2022.

[191] Aleph Alpha API, *What is Luminous.* [Online]. Available: https://docs.aleph-alpha.com/docs/Deprecated%20Luminous/Deprecated-Luminous/Deprecated-Luminous/ (accessed: Oct. 26 2024).

[192] A. Chowdhery *et al.,* "PaLM: Scaling Language Modeling with Pathways," Apr. 2022. [Online]. Available: http://arxiv.org/pdf/2204.02311

[193] A. Mitra *et al.,* "Orca 2: Teaching Small Language Models How to Reason," 2023.

[194] M. Conover *et al., Free Dolly: Introducing the World's First Open and Commercially Viable Instruction-Tuned LLM - The Databricks Blog.* [Online]. Available: https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm (accessed: May 21 2024).

[195] Q. Malartic *et al.,* "Falcon2-11B Technical Report," Jul. 2024. [Online]. Available: http://arxiv.org/pdf/2407.14885

[196] G. Penedo *et al.,* "The RefinedWeb Dataset for Falcon LLM: Outperforming Curated Corpora with Web Data, and Web Data Only," Jun. 2023. [Online]. Available: https://arxiv.org/pdf/2306.01116v1

List of Figures

List of Tables

List of Listings

List of Abbreviations

AI	Artificial Intelligence
Annot.	Annotation
API	Application Programming Interface
AUROC	Area Under the Receiver Operating Characteristic
BERT	Bidirectional Encoder Representations from Transformers [2]
Bi-LSTM	Bidirectional Long Short-Term Memory [128]
CPU	Central Processing Unit
CRISP-DM	CRoss-Industry Standard Process for Data Mining [75]
CUDA	Compute Unified Device Architecture [117]
GDELT	Global Database of Events, Language, and Tone [79]
GDPR	General Data Protection Regulation [86]
GloVe	Global Vectors [24]
GPT	Generative Pre-trained Transformer
GPU	Graphics Processing Unit
HPC	High-Performance Computing
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
JRC	Joint Research Center of the European Commission [33]
Lang.	Language
LCF	Local Context Focus [129]
LLM	Large Language Model
LSTM	Long Short-Term Memory [130]
MAD-TSC	Multilingual Aligned Dataset for Target-dependent Sentiment Classification [57]
MMS	Massively Multilingual Corpus of Sentiment Datasets [83]
mBERT	Multilingual BERT [2]
MMLU	Massive Multitask Language Understanding [108]
NBM	Naïve Bayes Multinomial classifier [131]
NLP	Natural Language Processing
RAIL	Responsible AI License [132]
RAM	Random-Access Memory
RoBERTa	Robustly optimized BERT pretraining Approach [133]
ROC	Receiver Operating Characteristic [134]
RQ	Research Question
SFFS	Sequential Forward Floating Selection [135]
SVM	Support Vector Machine [136]
TF-IDF	Term Frequency-Inverse Document Frequency [137, 138]
TSC	Target-dependent Sentiment Classification
URL	Uniform Resource Locator

VADER	Valence Aware Dictionary for sEntiment Reasoning [15]
WWW	World Wide Web
XLM	Cross-lingual Language Model [139]
XLM-R	XLM-RoBERTa [30]

List of Language Codes

AR	Arabic
DE	German
EL	Greek
EN	English
ES	Spanish
FR	French
HR	Croatian
IT	Italian
JA	Japanese
KO	Korean
NL	Dutch
PT	Portuguese
RO	Romanian
SL	Slovenian
TR	Turkish
ZH	Chinese

Documentation Table – AI-Based Tools

AI-Based Tool	Purpose of Use	Prompt, Source, Page, Paragraph…
ChatGPT (4o)	Suggestions for translating the thesis text	“Please translate into American English: …“ / “Übersetze ins Amerikanische Englisch: …“ / “Translate the German part into American English please: …”, “Correct my English: …”, “Please correct my English (American English) and integrate the German parts smoothly” etc, [140–164], entire thesis
Deepl Translate	Alternative suggestions for translating the thesis text	Input oft the German thesis text, [165], Sections 1 and 2

A Sentiment Analyses of News Texts

Authors	Title	Lang. [166]	Approach	Most Effective Techniques	Metrics of Best Techniques
Bautin *et al.*, 2008 [44]	"International Sentiment Analysis for News and Blogs"	EN (AR, ZH, FR, DE, IT, JA, KO, ES via machine translation into EN)	Aggregation of entity-related sentiments per country	Machine translation into English, Lydia [167], normalization of country polarity scores	-
Balahur *et al.*, 2009 [168], 2010 [33]	"Sentiment Analysis in the News"	EN	Lexicon-based, target-dependent analysis of quotes	Subjectivity pre-filtering and combination of sentiment lexicons (SentiWordNet [16], WordNet-Affect [169], MicroWNOp [170] + JRC Tonality, an internal list of opinion words of the Joint Research Centre of the European Commission) on a text window of 6 words	Precision (positive polarity): 73 % Precision (negative polarity): 71% Recall (positive polarity): 75% Recall (negative polarity): 69%
Kaya, Fidan und Toroslu, 2012 [54]	"Sentiment Analysis of Turkish Political News"	• TR • EN	Document level	Maximum entropy with stemmed unigrams	76.78%

Authors	Title	Lang. [166]	Approach	Most Effective Techniques	Metrics of Best Techniques
Xiong, Xu and Liang, 2014 [171]	“An Architecture for Automatic Opinion Classification in Western Online News”	EN	Text retrieval of English news texts using a Chinese input front-end, sentiment classification at document level	?	?
Li *et al.*, 2016 [70]	“Hierarchical classification in text mining for sentiment analysis of online news”	EN	Document-Level, reducing data volume to emphasize relevant features	SVM [136] for word vector generation, maximum entropy for sentiment classification	Accuracy: 94.12%
Islam *et al.*, 2017 [172]	“Polarity Detection of Online News Articles Based on Sentence Structure and Dynamic Dictionary”	EN	Sentence-level, aggregation of sentiments from sub-sentences	Lexicon-based approach tailored to sentence structure	Accuracy: 91.07%
Bučar, Žnidaršič, and J. Povh, 2018 [59]	“Annotated news corpora and a lexicon for sentiment analysis in Slovene”	SL	Language-specific sentiment lexicon, news texts annotated at sentence, paragraph and document level	NBM [131] classifier used on balanced data for two-class and three-class classification on sentence-level	• For two-class classification: 97.83% +/- 0.98% • For three-class classification: 79.85% +/- 1.93%
Aker *et al.*, 2019 [72]	“Sentiment-Aware Detection Method of Fake News Based on Linguistic Fuzzy Bi-LSTM”	EN	Document level, fake and non-fake news articles	Combination of BERT [2] with fuzzy-based Bi-LSTM [128] architecture	Accuracy: 96.88%

Authors	Title	Lang. [166]	Approach	Most Effective Techniques	Metrics of Best Techniques
Taj, Shaikh und Meghji, 2019 [173]	“Sentiment Analysis of News Articles: A Lexicon based Approach”	EN	Document level	Term Frequency-Inverse Document Frequency (TF-IDF) [174] weights, WordNet [175] and SentiWordNet 3.0 [176]	?
Tamayo *et al.*, 2019 [60]	“Sentiment Analysis of News Articles in Spanish using Predicate Features”	ES	Document level, Sentiment analysis with SVM [136] + sequential forward floating selection (SFFS) [135]	SVM with Gaussian kernel [136] based on predicate features	Accuracy: 69%
Chan *et al.*, 2020 [35, 177]	“How Combining Terrorism, Muslim, and Refugee Topics Drives Emotional Tone in Online News: A Six-Country Cross-Cultural Sentiment Analysis”	• AR • DE • EN • TR	Fear and pity annotated at sentence level, sentiment dictionaries for fear, joy, sadness, anger and pity	Dictionary-based for three different kinds of subjects (terrorism, islam, refugee)	• Precision: 85% (AR, terrorism) – 92% (EN, refugee) • Recall: 93.5% (AR, islam) - 98,6% (EN, islam)
Pelicon *et al.*, 2020 [55]	“Zero-Shot Learning for Cross-Lingual News Sentiment Classification”	• SL • HR	Zero-shot cross-lingual sentiment analysis, document, paragraph and sentence level	Manually annotated Croatian corpus, text representation from the beginning and end of an article, mBERT [2] with sentiment training and fine-tuning	F1-Score • For Slovene: 66.33 • For Croatian: 54.77

Authors	Title	Lang. [166]	Approach	Most Effective Techniques	Metrics of Best Techniques
Hamborg, Donnay and Gipp, 2021 [73]	“Towards Target-Dependent Sentiment Classification in News Articles”	EN	Target-dependent sentiment analysis	Domain-adapted LCF-BERT [129] with class-weighted cross-entropy loss	Average recall: 69,8 Accuracy: 66.0
Awar, 2022 [178]	“Sentiment Analysis on News Headlines: Classic Supervised Learning vs Deep Learning Approach”	EN	Headline level, practical guide, no research	Snorkel [179] for binary labelling of test data based on predefined word lists, deep learning (sequential model)	Accuracy: 96%
Kaliappan, Natrayan and Rajput, 2023 [180]	“Sentiment Analysis of News Headlines Based on Sentiment Lexicon and Deep Learning”	EN	Headline level	Snorkel [179] for binary labelling of test data based on predefined word lists, deep learning	“High accuracy ratings”
Mello, Cheema and Thakkar, 2023 [56]	“Combining sentiment analysis classifiers to explore multilingual news articles covering London 2012 and Rio 2016 Olympics”	• EN • PT	Sentiment Analysis at headline level comparing and combining different algorithms	Majority voting by the three best algorithms (the Valence Aware Dictionary for sEntiment Reasoning, VADER [15], and two BERT versions trained on different datasets), excluding inconclusive headlines and ignoring them in the statistics	Accuracy: 65% - 74.7%, depending on the news outlet

Authors	Title	Lang. [166]	Approach	Most Effective Techniques	Metrics of Best Techniques
Seth and Sharaff, 2023 [71]	“Sentiment-Aware Detection Method of Fake News Based on Linguistic Fuzzy Bi-LSTM”	EN	Sentiment Analysis used to better identify fake news at document level	Combination of BERT [2] embedding with hedge based fuzzy logic with a bidirectional long short-term memory (Bi-LSTM) [128] algorithm	Accuracy: 96.88%
Ring *et al.*, 2024 [61]	“Approaches to sentiment analysis of Hungarian political news at the sentence level”	HU	Manually annotated dataset and sentiment dictionary newly created for political texts, binary sentiment Analysis at sentence level (dictionary-based vs. different machine learning algorithms)	Removing mixed and neutral sentences from the dataset and using huBERT finetuned first for 3-class classification and then for binary sentiment classification	F1-score: 90%

B Sentiment-Annotated General News Corpora

Authors	Title	Lang. [166]	Approach
Li *et al.* 2012 [42]	"Annotating Opinions in German Political News"	DE	Manually annotated opinion frames specifying mainly target, source, and word or phrase that is the "text anchor". Two types of annotations: "thorough" including sentiments requiring extra-textual knowledge of politics, and "context-independent" where the sentiment is explicit. The data is not publicly available.
O'Keefe *et al.*, 2013 [62]	"An Annotated Corpus of Quoted Opinions in News Articles"	EN	Quotes annotated as supporting, neutral, or opposing a position statement.
Arruda, Roman and Monteiro, 2015 [58]	"An Annotated Corpus for Sentiment Analysis in Political News"	PT	Paragraph level, target-dependent sentiment analysis.
Bučar, Žnidaršič, and J. Povh, 2018 [59]	"Annotated news corpora and a lexicon for sentiment analysis in Slovene"	SL	News texts annotated manually at sentence, paragraph and document level.
Pelicon *et al.*, 2020 [55]	"Zero-Shot Learning for Cross-Lingual News Sentiment Classification"	HR	News texts annotated manually at sentence, paragraph and document level.
Hamborg and Donnay, 2021 [63]	"NewsMTSC: A Dataset for (Multi-) Target-dependent Sentiment Classification in Political News Articles"	EN	Dataset manually annotated for target-dependent sentiment analysis.
Mello, Cheema and Thakkar 2023 [56]	"Combining sentiment analysis classifiers to explore multilingual news articles covering London 2012 and Rio 2016 Olympics	• EN • PT	Two datasets, one of them manually annotated at headline level, the other automatically annotated at headline and document level via English machine translations of the Portuguese articles.
Ring *et al.*, 2024 [61]	"Approaches to sentiment analysis of Hungarian political news at the sentence level"	• HU	Dataset manually annotated at sentence level.
Dufraisse et. al., 2024 [57]	"MAD-TSC: A Multilingual Aligned News Dataset for Target-dependent Sentiment Classification"	• EN • ES • DE • FR • IT • NL • PT • RO	Parallel, sentiment-annotated corpus for target-dependent sentiment analysis.

C MMS News Datasets

Name	License	URL	Lang [166]	Samples	Annot. Level					Scale	Annotators	Conclusions
					Headline	Document	Paragraph	Sentence	Target			
MMS (Massively Multilingual Corpus of Sentiment Datasets) [83]	CC BY-NC 4.0 [181]	[14]		13,971						3-point polarity	At least 3 per sample	• Carefully compiled collection, high quality, ready to use • Drawbacks: ○ Imbalanced numbers of samples ○ No knowledge of HR and SL ○ Different annotation levels: sl and hr annotate at text level, en_per_sent at document or paragraph level, en_vader_nyt and en_financial_phrasebank at sentence level.
• en_financial_ phrasebank_ sentences_75agree			• EN	• 3,448				x				
• en_per_sent			• EN	• 5,333		x						
• en_vader_nyt			• EN	• 5,190				x				
• hr_sentiment_news_ document			• HR	• 2,025		x						
• sl_sentinews			• SL	• 10,417 documents		x						

[14] https://huggingface.co/datasets/Brand24/mms

D MMS News: Original Datasets

Name	License	URL	Lang. [166]	Samples	Annot. Level					Scale	Annotators	Conclusions
					Headline	Document	Paragraph	Sentence	Target			
Sentiment Annotated Dataset of Croatian News [55]	CC BY-NC-ND [97]	[15]	HR	• 2025 documents • 12,032 paragraphs • 25,074 sentences		x	x	x		3-point polarity, originally 5-point polarity	2-6 per document	Source of MMS/ hr_sentiment_news_document Same annotation rules as for the SentiNews data
SentiNews 1.0 [59]	CC BY-SA 4.0 [182]	[16]	SL	• 10,427 documents • 89,999 paragraphs • 168,899 sentences		x	x	x		5-point polarity	2-6 per document	Source of MMS/sl_sentinews

[15] https://www.clarin.si/repository/xmlui/handle/11356/1342 or http://hdl.handle.net/11356/1342

[16] https://github.com/19Joey85/Sentiment-annotated-news-corpus-and-sentiment-lexicon-in-Slovene

Name	License	URL	Lang. [166]	Samples	Annot. Level					Scale	Annotators	Conclusions
					Headline	Document	Paragraph	Sentence	Target			
Financial_phrasebank [91]	CC BY-NC-SA 3.0 [183]	[17]	EN.	4,840				x		3-point polarity	5-8 per sentence	> 75% agreement subset (3,448 samples) is source of MMS/ en_financial_phrasebank_ sentences_75agree.
					Possible problem: sentiments are only assessed as far as economic and/or financial topics are concerned; other sentiments are labeled as neutral.							
PerSenT [92]	CC BY 4.0 [184]	[18]	EN	• 5,339 documents • 38,352 paragraphs		x	x			3-point polarity	3 per document	Source of MMS/en_per_sent
NYT [15]	Cite paper	[19]	EN	• 500 New York Times opinion editorials • 5,190 sentences				x		Intensity (-4 to 4)	20 per sentence	Source of MMS/en_vader_nyt

[17] https://huggingface.co/datasets/financial_phrasebank
[18] https://stonybrooknlp.github.io/PerSenT/
[19] https://comp.social.gatech.edu/papers/

E Olympia – Dataset 1

Name	License	URL	Lang. [166]	Samples	Annot. Level					Scale	Annotators	Conclusions
					Headline	Document	Paragraph	Sentence	Target			
Sentiment-annotation-olympic-news, Dataset 1 [56] • guardian_london • dailymail_london	CC BY 4.0 [184], GNU GPLv3 [185]	[20], [21]	 • EN • EN	717 headlines • 217 headlines • 135 headlines	x x x					3-point polarity	1 domain expert	• English and Portuguese news articles mentioning the legacy of the Olympics in 2012 and 2016 in London (UK) and Rio (Brazil). • Full-text headlines and links to the news articles at the searched news outlets. • The articles in **Dataset 1** are from o The Guardian o Daily Mail o Globo o Estadao
• globo_rio • estadao_rio			• PT • PT	• 245 headlines • 120 headlines	x x							
	• Drawbacks: o Only few data o Only one annotator o No knowledge of PT o Access to the news outlets sometimes requires paid subscriptions. o The scraped and processed news articles are not available											

[20] https://github.com/caiocmello/sentiment-annotation-olympic-news
[21] https://zenodo.org/records/6323964#.Yh_UdXXP3RZ

F Olympic News – Dataset 2 (Only EN)

Name	License	URL	Lang. [166]	Samples	Annot. Level					Scale	Annotators	Conclusions
					Headline	Document	Paragraph	Sentence	Target			
Sentiment-annotation-olympic-news, Dataset 2 [56]	CC BY 4.0 [184], GNU GPLv3 [185]	[22], [23]	• EN	807 headlines	x	x				3-point polarity + 'other' where there was no unanimity between the classifiers (in 102 cases)	Combination of three classifiers	• The English articles in **Dataset 2** are from o The Guardian o Daily Mail o The Telegraph o BBC • Dataset 2 also contains Portuguese news articles, which were machine-translated into English and then automatically annotated in the same way as the English ones. Since translations are not suitable for serving as a possible data source for the present project, and the original Portuguese news are only referenced by links to paid news
• london_bbc			• EN	• 198 headlines	x	x						
• london_dailymail			• EN	• 135 headlines	x	x						
• london_telegraph			• EN	• 196 headlines	x	x						
• london_theguardian			• EN	• 217 headlines	x	x						
• rio_bbc			• EN	• 7 headlines	x	x						
• rio_dailymail			• EN	• 9 headlines	x	x						
• rio_telegraph			• EN	• 4 headlines	x	x						

[22] https://github.com/caiocmello/sentiment-annotation-olympic-news
[23] https://zenodo.org/records/6326348#.YiEuq9vLfRZ

Name	License	URL	Lang. [166]	Samples	Annot. Level					Scale	Annotators	Conclusions
					Headline	Document	Paragraph	Sentence	Target			
• rio_theguardian			• EN	• 41 headlines	x	x						outlets, they are not enumerated here.
				• **Drawbacks** (in addition to the ones mentioned for Dataset 1): ○ Only English ○ Automatic annotation								

G Generative LLMs

The following overview is based on the overview published by the Alexander Thamm GmbH [76].

Model	Year	Developer	URL	Parameters
Closed Source				
Claude [186, 187]	2023	Anthropic	[24]	~ >130 billion
Command R+ [188]	2023	Cohere Technologies Inc.	[25]	Considerable number
GPT-3.5 [26]	2022	OpenAI	[26]	175 billion
GPT-4 [189]	2023	OpenAI	[27]	Extensive number (>175 billion)
LaMDA [190]	2021	Google Brain	-	137 billion
Luminous [191]	2023	Aleph Alpha	[28], [29]	70 billion.
PaLM [192]	2023	Google	[30]	• 8 billion • 62 billion • 540 billion

[24] https://www.anthropic.com/

[25] https://cohere.com/

[26] https://chatgpt.com/g/g-F00faAwkE-open-a-i-gpt-3-5

[27] https://openai.com/index/gpt-4/

[28] https://docs.aleph-alpha.com/api/

[29] https://app.aleph-alpha.com/login?redirect=%2Fplayground%2Fcompletion

[30] https://ai.google.dev/palm_docs

Model	Year	Developer	URL	Parameters
Open Source for Non-commercial or Research Purposes Only				
Guanaco-65B [101]	2023	Tim Dettmers	[31]	65 billion
LLaMA [100]	2023	Meta AI	[32], [33]	Various sizes: • 7 billion • 13 billion • 33 billion • 65 billion
Orca [102, 193]	2023	Microsoft Research	[34, 35]	• 7 billion • 13 billion
Vicuna [103]	2023	LMSYS	[36, 37]	• 7 billion • 13 billion • 33 billion
Open Source				
Bloom [107]	2022	BIG Science Initiative	[38]	>176 billion
Dolly 2.0 [194]	2023	Databricks	[39]	12 billion, based on the EleutherAI Pythia models
Falcon [195, 196]	2023	Technology Innovation Institute (Abu Dhabi)	[40, 41]	• 7 billion • 40 billion • 180 billion

[31] https://github.com/artidoro/qlora
[32] https://github.com/facebookresearch/llama
[33] https://www.llama.com
[34] https://huggingface.co/microsoft/Orca-2-7b
[35] https://huggingface.co/microsoft/Orca-2-13b
[36] https://github.com/lm-sys/FastChat
[37] https://huggingface.co/lmsys
[38] https://huggingface.co/bigscience/bloom
[39] https://huggingface.co/databricks/dolly-v2-12b
[40] https://falconllm.tii.ae/falcon-models.html
[41] https://huggingface.co/tiiuae/falcon-7b, https://huggingface.co/tiiuae/falcon-40b, https://huggingface.co/tiiuae/falcon-180B

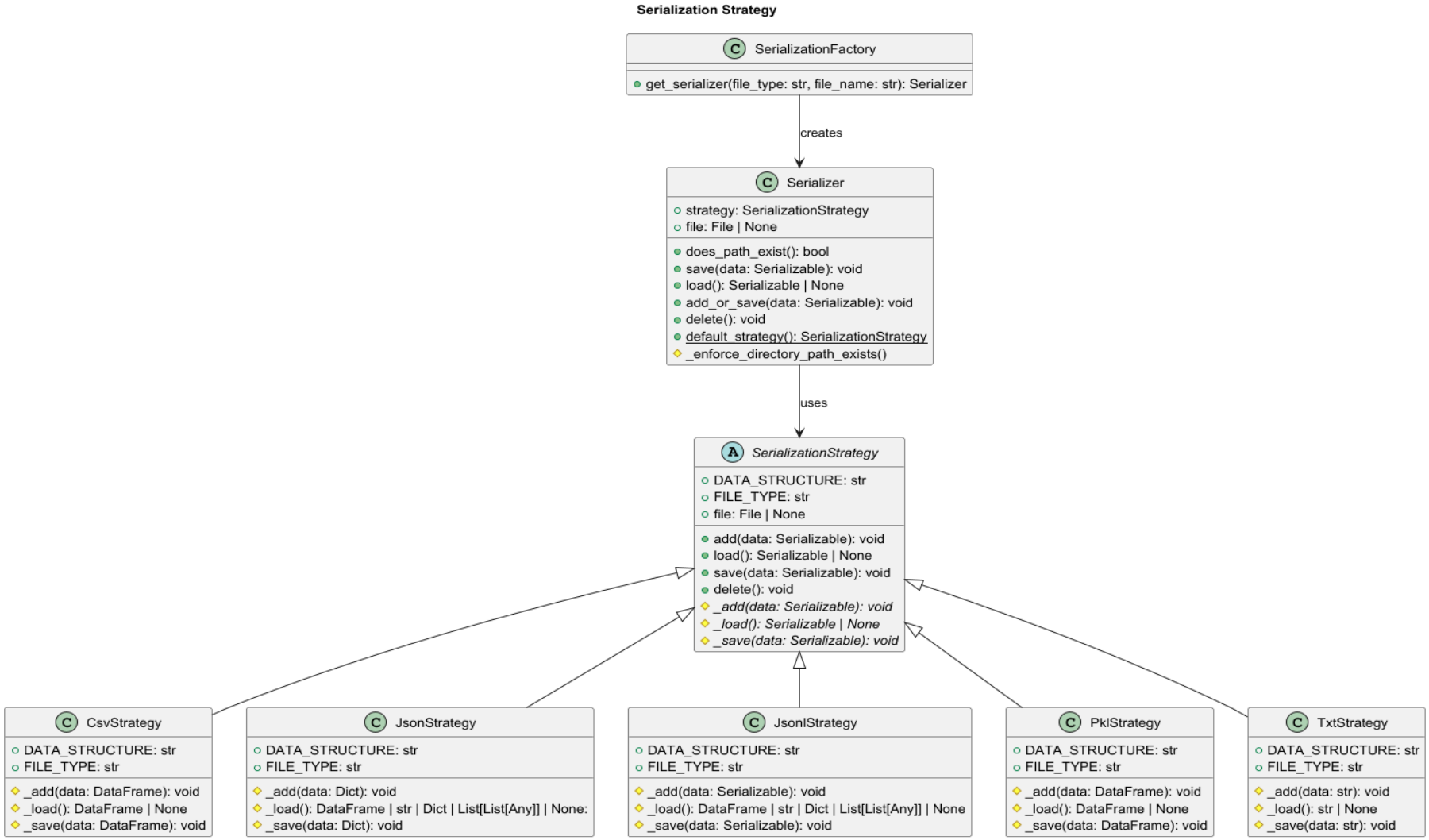
H MAD-TSC Statistics

	mad_tsc_de	mad_tsc_en	mad_tsc_es	mad_tsc_fr	mad_tsc_it	mad_tsc_nl	mad_tsc_pt	mad_tsc_ro
n_elements	5110	5110	5110	5110	5110	5110	5110	5110
min_length	19	26	22	24	23	24	22	22
max_length	563	415	776	762	790	786	604	571
mean_length	209.392564	192.424658	209.879843	205.058317	198.449902	212.449315	200.297456	202.895303
median_length	201.0	188.0	203.0	197.0	191.0	204.0	194.0	196.0
std_dev_length	85.467039	72.13464	84.133524	81.819269	78.915209	86.281259	79.411963	79.462413
min_sentences	1	1	1	1	1	1	1	1
max_sentences	6	3	5	5	4	5	5	4
mean_sentences	1.48728	1.102935	1.17593	1.192955	1.176321	1.306262	1.195303	1.179061
median_sentences	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
std_dev_sentences	0.709893	0.324463	0.438605	0.451557	0.425304	0.559916	0.454009	0.428748
min_words	4	5	4	5	4	4	3	4
max_words	76	66	126	122	117	125	96	87
mean_words	28.643249	31.425832	34.206654	34.001761	31.141487	32.4818	31.957926	31.154207
median_words	27.0	31.0	33.0	33.0	30.0	32.0	31.0	30.0
std_dev_words	11.425939	11.634778	13.668128	13.469633	12.252759	13.067764	12.61454	12.107874
min_polarity	2	2	2	2	2	2	2	2
max_polarity	6	6	6	6	6	6	6	6
mean_polarity	3.773386	3.773386	3.773386	3.773386	3.773386	3.773386	3.773386	3.773386
median_polarity	4.0	4.0	4.0	4.0	4.0	4.0	4.0	4.0
std_dev_polarity	1.541085	1.541085	1.541085	1.541085	1.541085	1.541085	1.541085	1.541085
unique_polarity	[2.0, 4.0, 6.0]	[2.0, 4.0, 6.0]	[2.0, 4.0, 6.0]	[2.0, 4.0, 6.0]	[2.0, 4.0, 6.0]	[2.0, 4.0, 6.0]	[2.0, 4.0, 6.0]	[2.0, 4.0, 6.0]
n_positive	1260	1260	1260	1260	1260	1260	1260	1260
n_negative	1839	1839	1839	1839	1839	1839	1839	1839
n_neutral	2011	2011	2011	2011	2011	2011	2011	2011
%_positive	24.657534	24.657534	24.657534	24.657534	24.657534	24.657534	24.657534	24.657534
%_negative	35.988258	35.988258	35.988258	35.988258	35.988258	35.988258	35.988258	35.988258

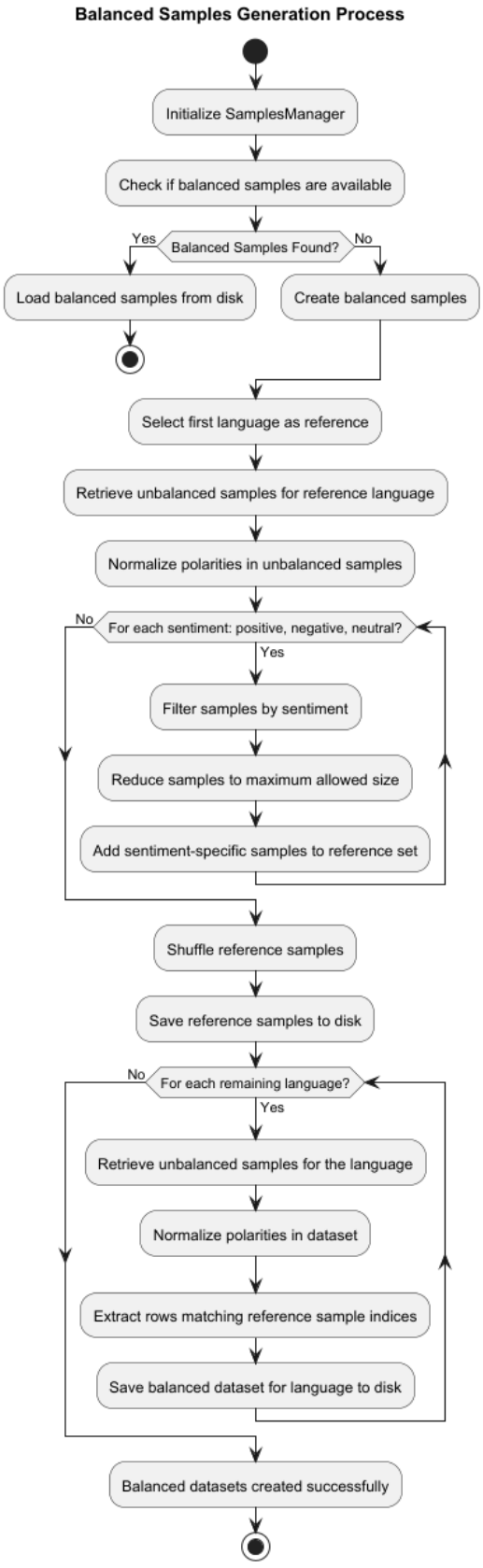
%_neutral	39.354207	39.354207	39.354207	39.354207	39.354207	39.354207	39.354207	39.354207
n_unique_elements	2240	2243	2262	2222	2268	2241	2237	2245
n_most_frequent_unique_elements	1	1	1	1	1	1	1	1
n_least_frequent_unique_elements	1650	1658	1682	1651	1678	1673	1660	1673
most_frequent_unique_elements	[Angela Merkel, Merkel, Berlusconi]	[Angela Merkel, Merkel, Berlusconi]	[Angela Merkel, Merkel, Berlusconi]	[Angela Merkel, Merkel, Berlusconi]	[Angela Merkel, Merkel, Berlusconi]	[Angela Merkel, Merkel, Berlusconi]	[Angela Merkel, Merkel, Berlusconi]	[Angela Merkel, Merkel, Berlusconi]
least_frequent_unique_elements	[Ceaucescus, Diane Abbot, Florian Möllers]	[Batasuna, Paul, Florian Möllers]	[Egemen Bagis, Ahmad Masadeh, Antonis Smaras]	[Ronny van Reet, Robert Kaliňák, Florian Möllers]	[Stephan Kessler, Carl Tham, Florian Möllers]	[Fabre, Roumiania Jelevia, Anders Behing Breivik]	[Sebastian Münster, Terese Burauskaite, Florian Möllers]	[Ronny van Reet, Robert Kaliňák, Nečs]
max_frequency	183	199	197	199	182	179	199	166
min_frequency	1	1	1	1	1	1	1	1
mean_frequency	2.28125	2.278199	2.259063	2.29973	2.253086	2.280232	2.284309	2.276169
median_frequency	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
dev_frequency	6.854606	6.929375	6.847559	7.01219	6.761379	6.850509	6.943265	6.805182
max_percentage_of_occurrences	0.035812	0.038943	0.038552	0.038943	0.035616	0.035029	0.038943	0.032485
min_percentage_of_occurrences	0.000196	0.000196	0.000196	0.000196	0.000196	0.000196	0.000196	0.000196
mean_percentage_of_occurrences	0.000446	0.000446	0.000442	0.00045	0.000441	0.000446	0.000447	0.000445
median_percentage_of_occurrences	0.000196	0.000196	0.000196	0.000196	0.000196	0.000196	0.000196	0.000196
std_dev_percentage_of_occurrences	0.001341	0.001356	0.00134	0.001372	0.001323	0.001341	0.001359	0.001332

I Code Diagrams

I.1 Serialization Strategy

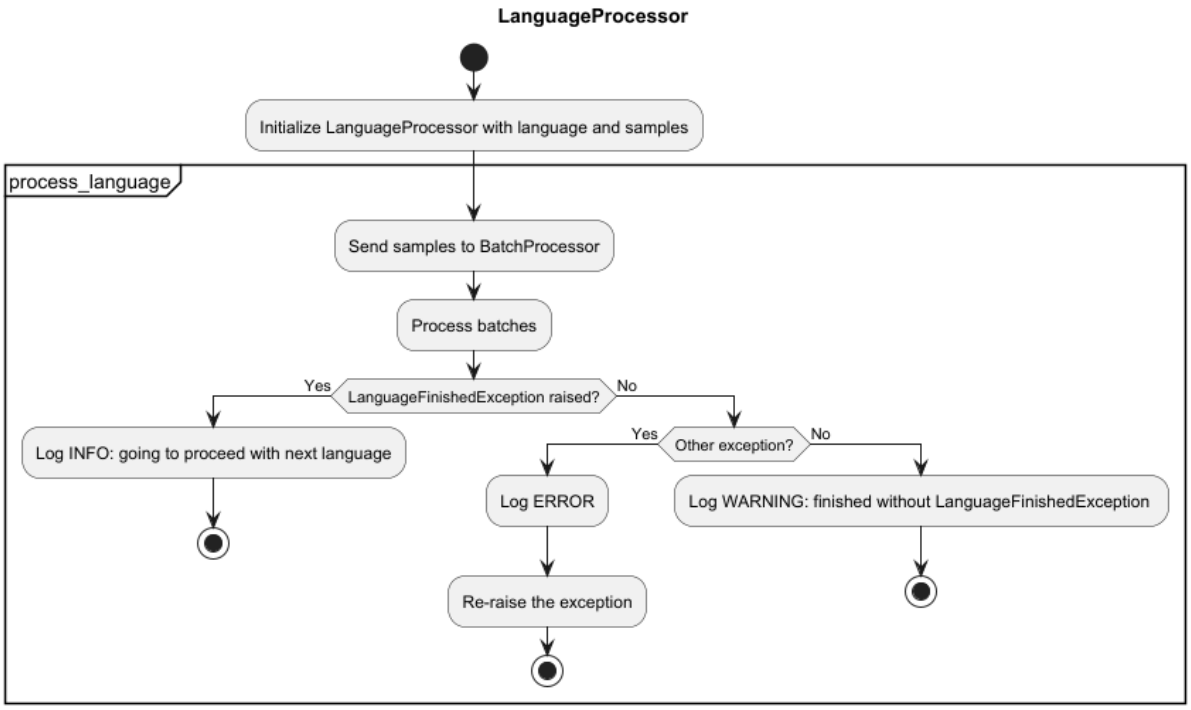


I.2 Balanced Samples Generation

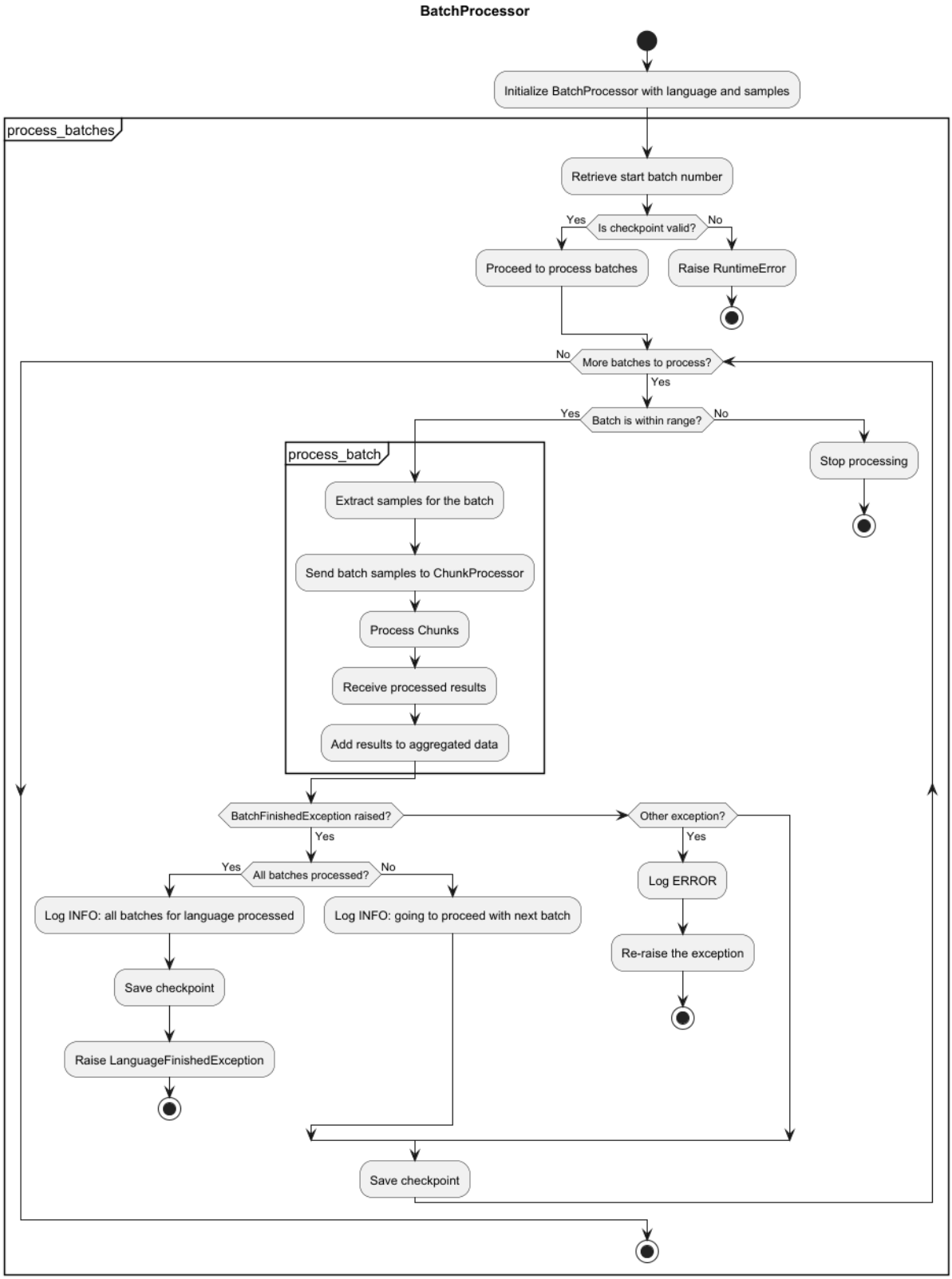


I.3 Sentiment Retrieval: Activity Diagrams

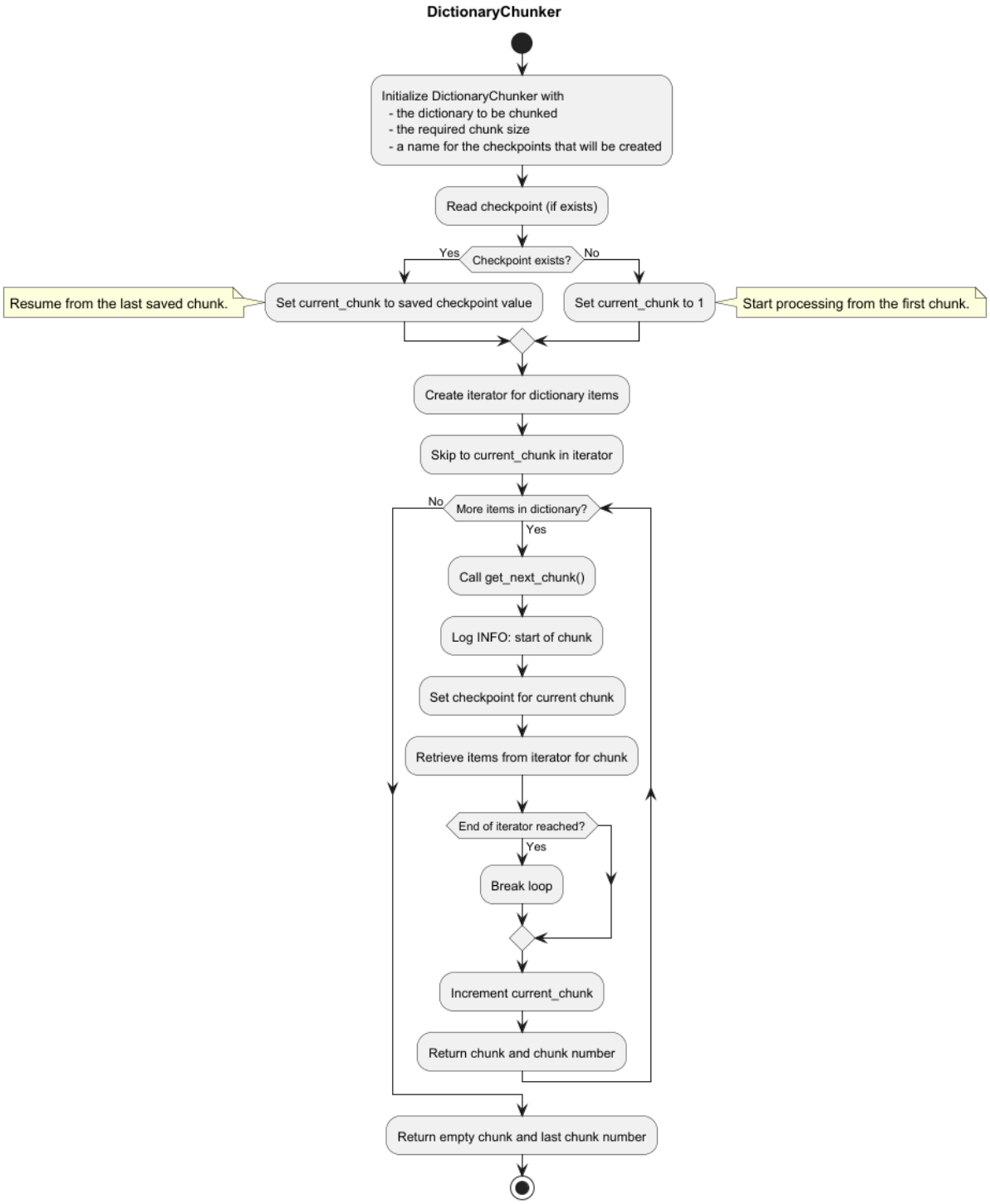
I.3.1 LanguageProcessor



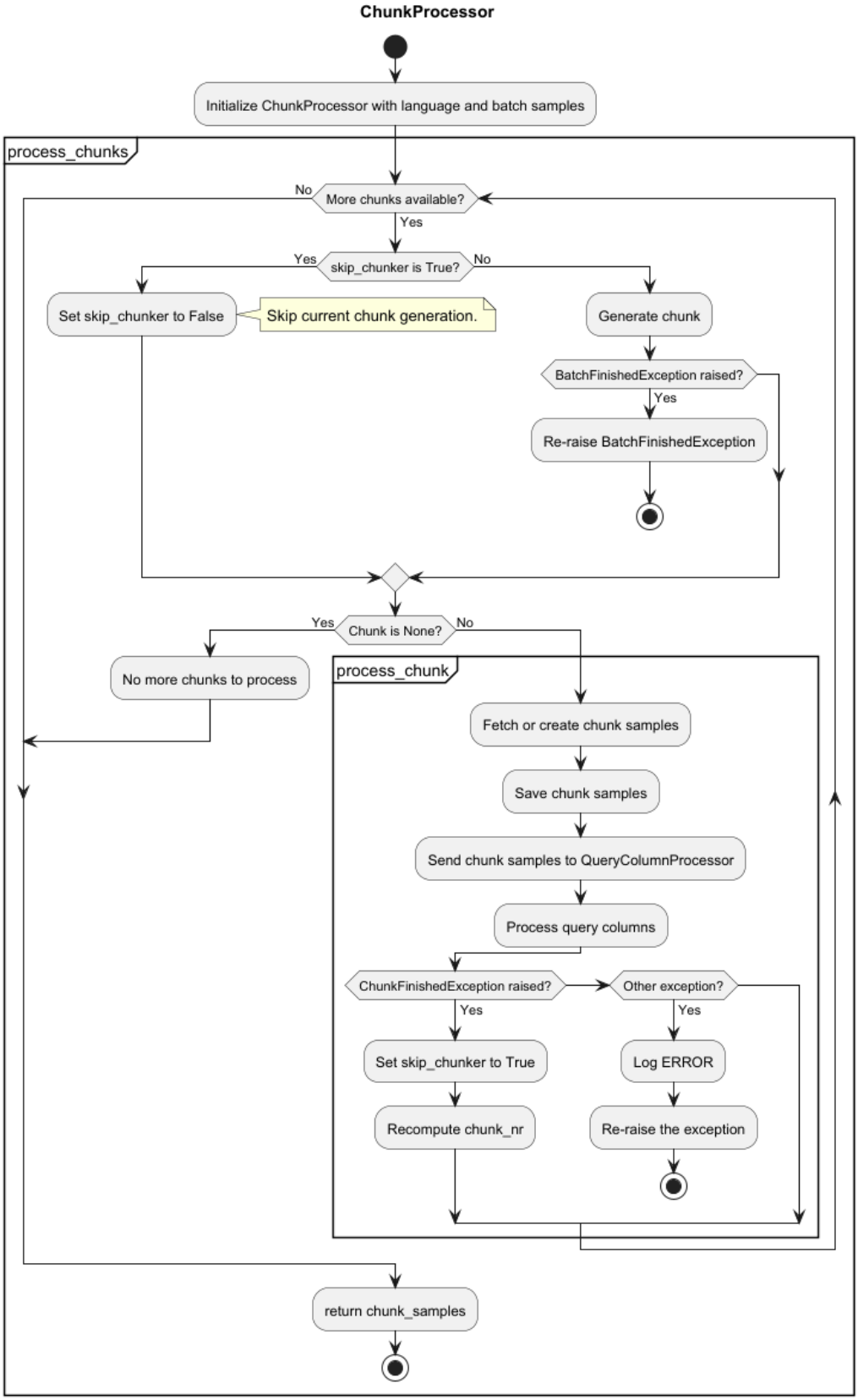
I.3.2 BatchProcessor



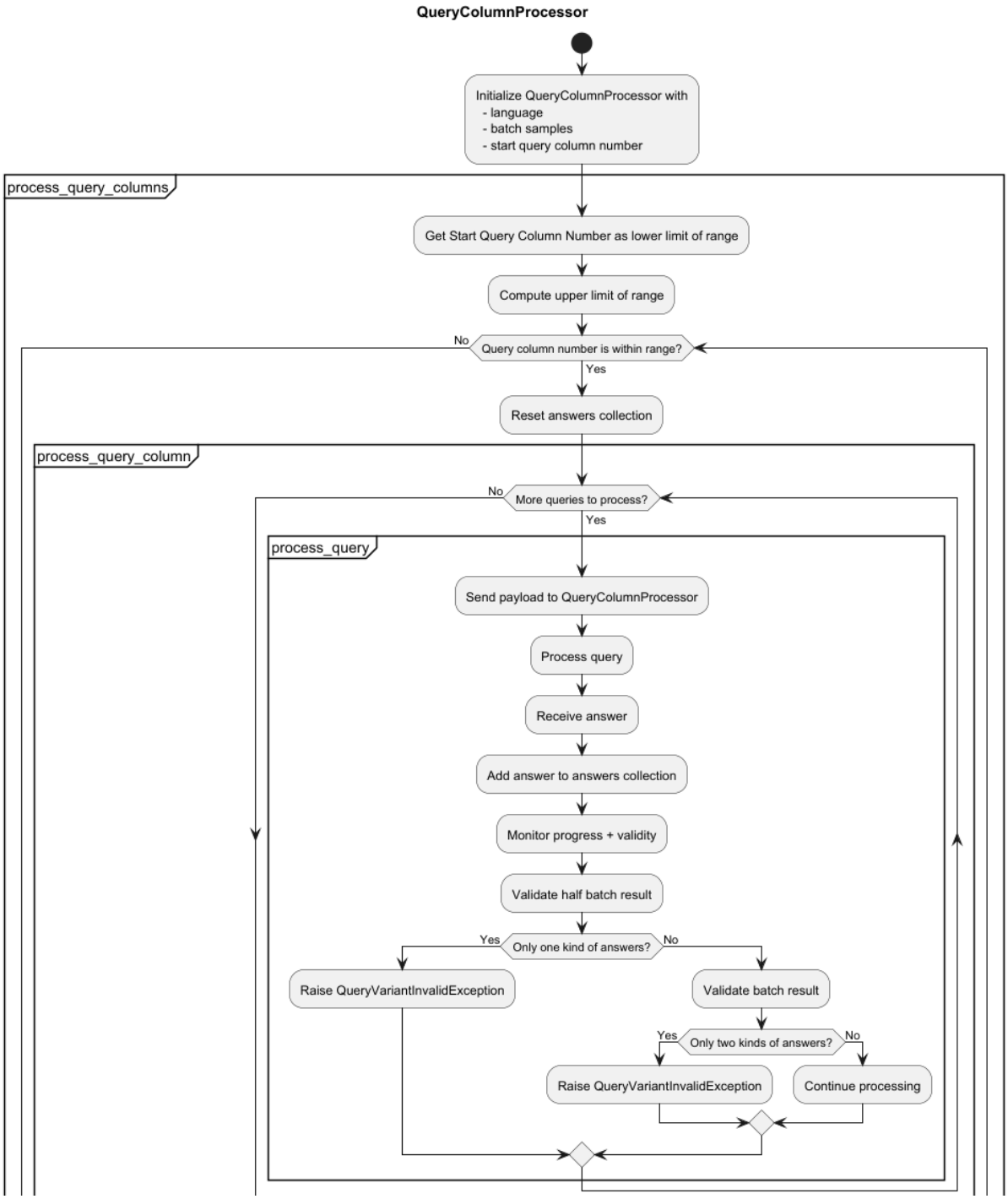
I.3.3 DictionaryChunker

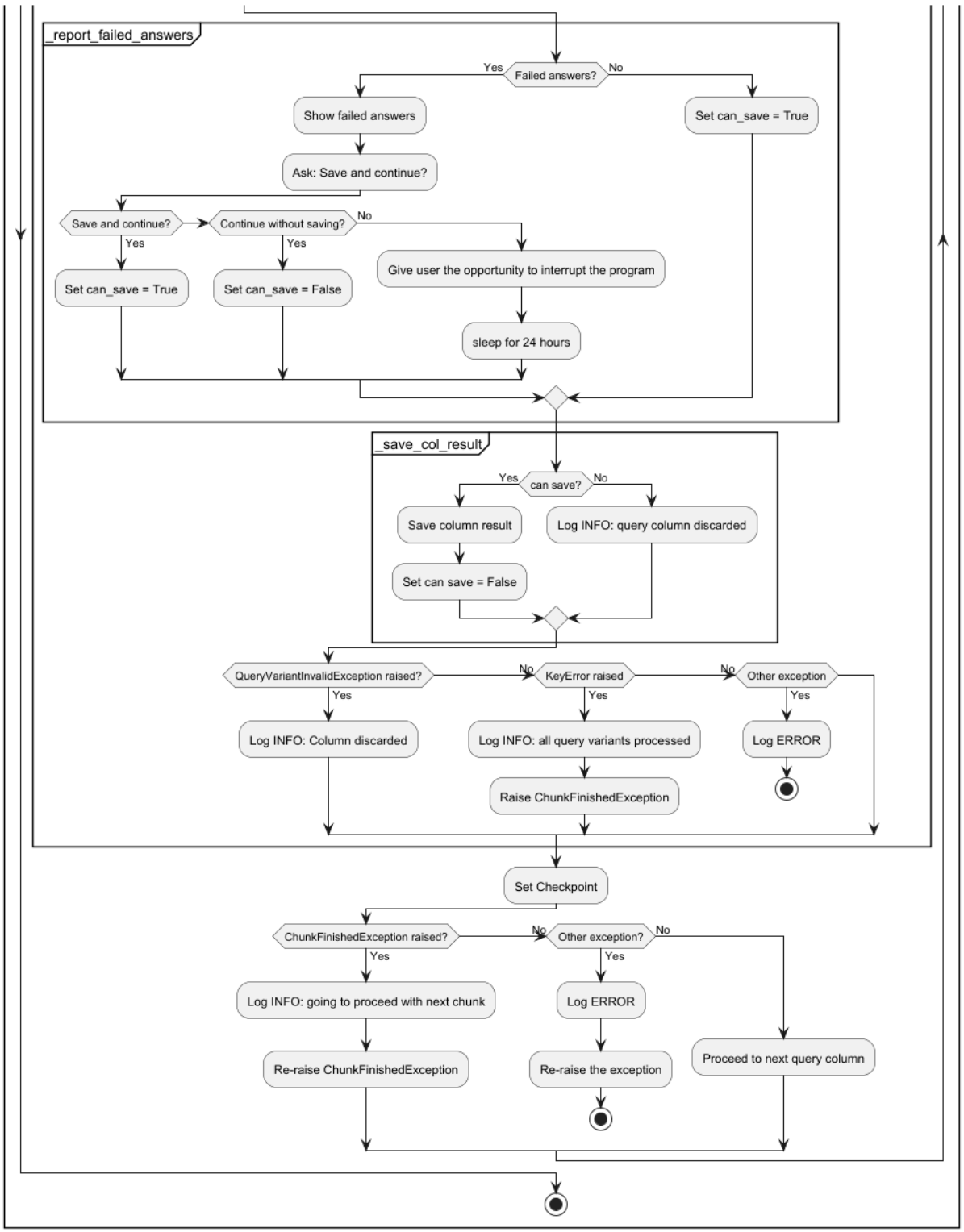


I.3.4 ChunkProcessor

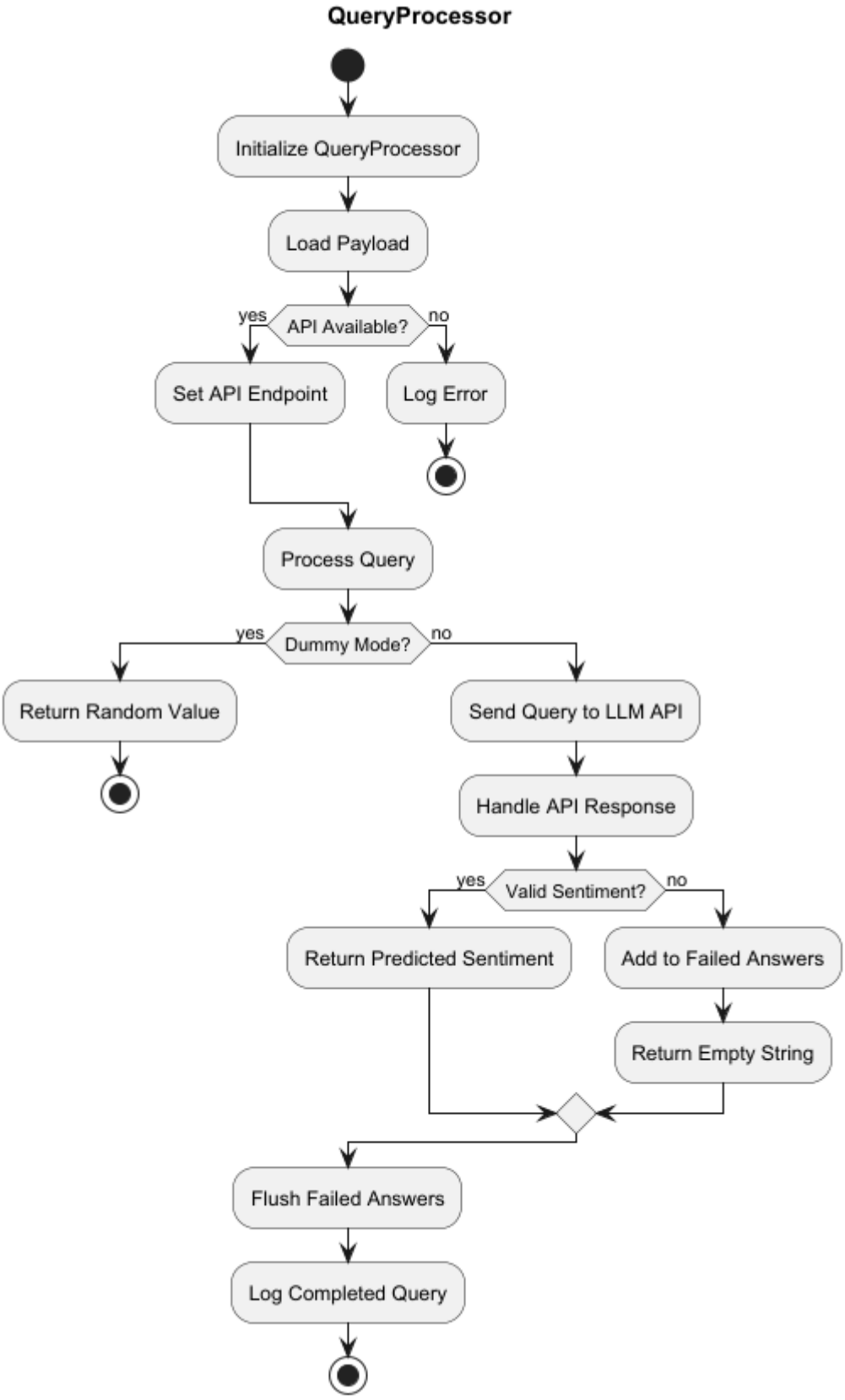


I.3.5 QueryColumnProcessor




_report_failed_answers
Failed answers?
Yes
No
Show failed answers
Set can_save = True
Ask: Save and continue?
Save and continue?
Continue without saving?
No
Yes
Yes
Set can_save = True
Set can_save = False
Give user the opportunity to interrupt the program
sleep for 24 hours
_save_col_result
can save?
Yes
No
Save column result
Log INFO: query column discarded
Set can save = False
QueryVariantInvalidException raised?
No
KeyError raised
No
Other exception
Yes
Yes
Yes
Log INFO: Column discarded
Log INFO: all query variants processed
Log ERROR
Raise ChunkFinishedException
Set Checkpoint
ChunkFinishedException raised?
No
Other exception?
No
Yes
Yes
Log INFO: going to proceed with next chunk
Log ERROR
Re-raise ChunkFinishedException
Re-raise the exception
Proceed to next query column

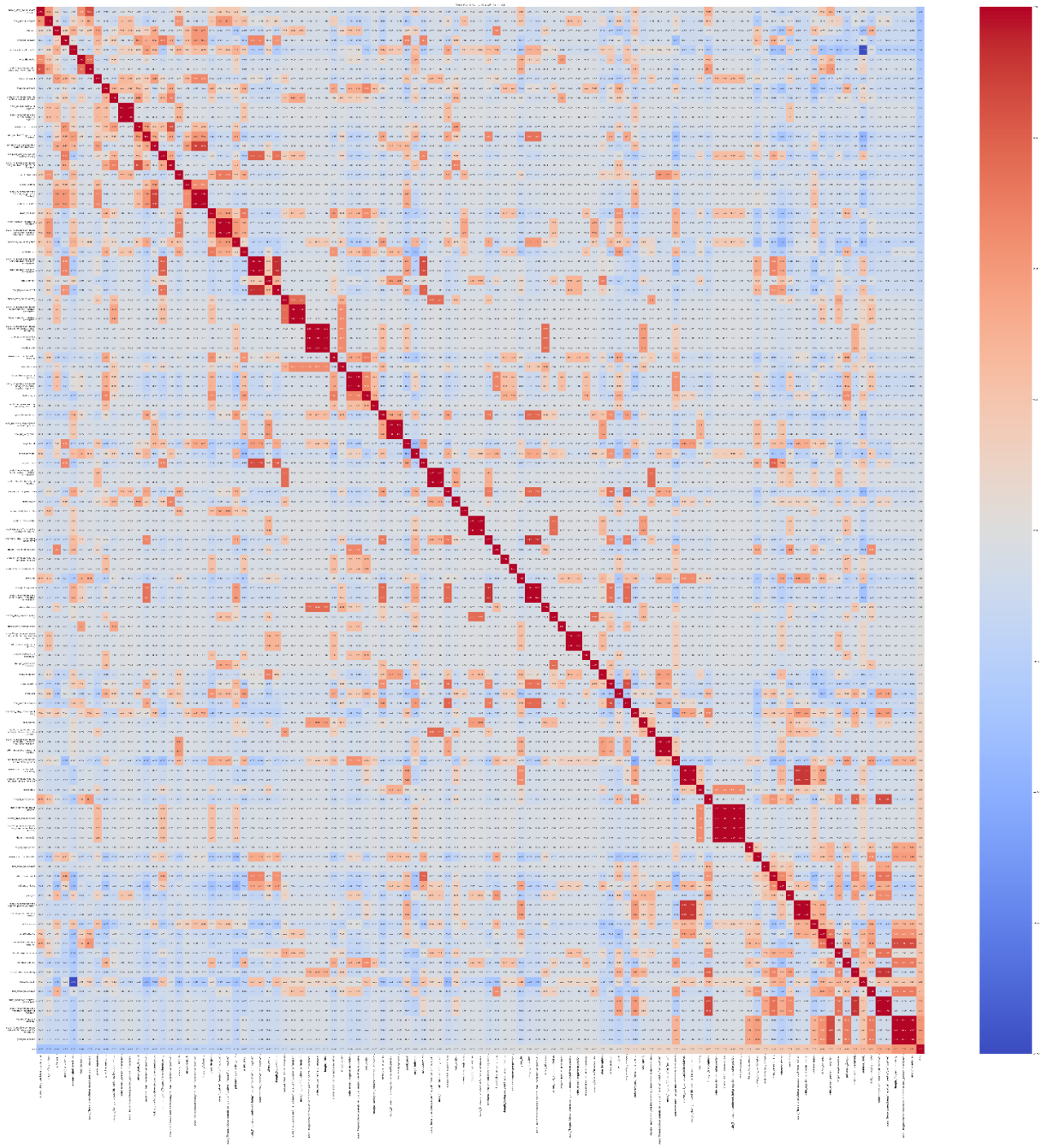
I.3.6 QueryProcessor



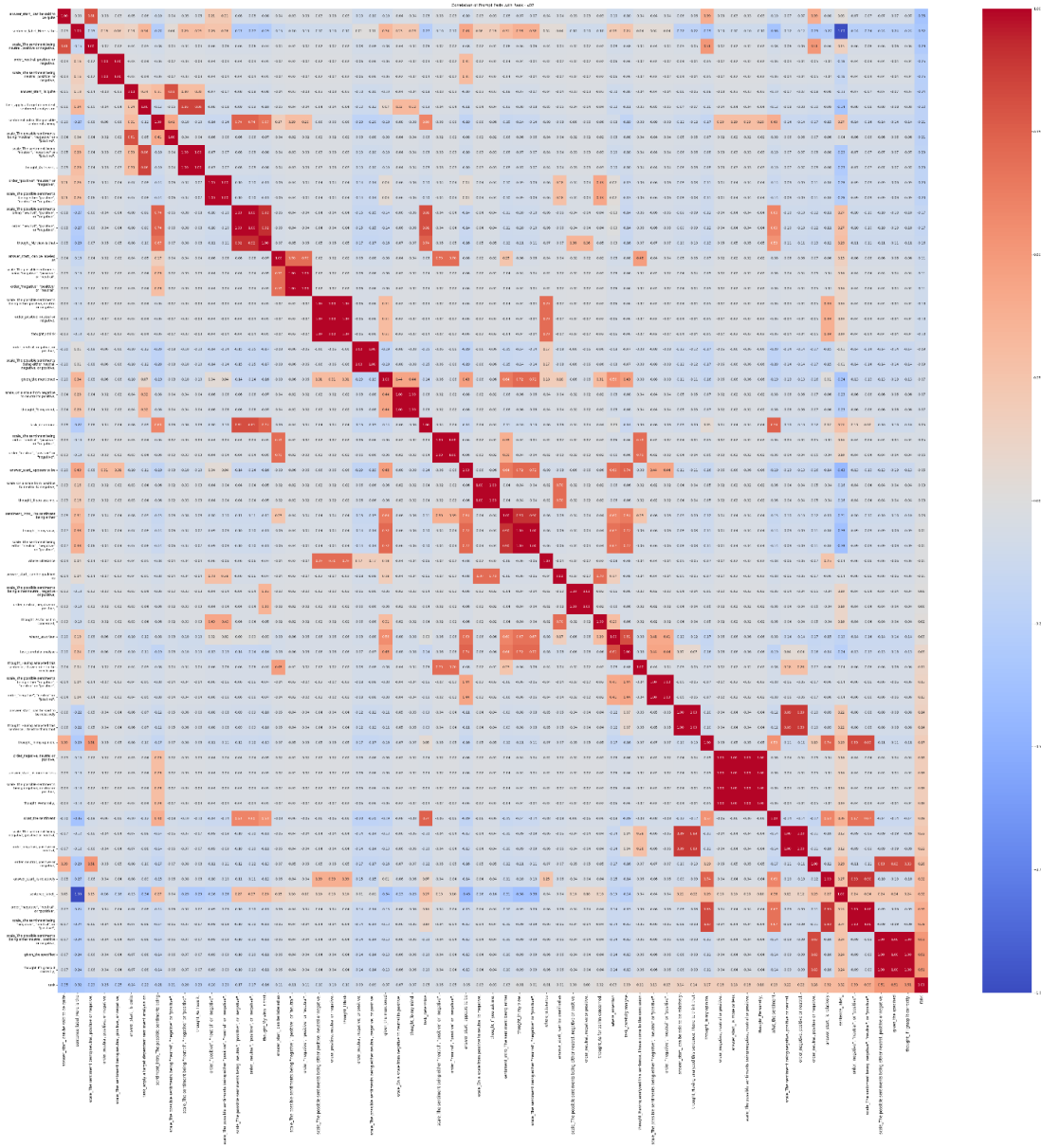
J PromptEngineeringStrategy2: Korrelation Heatmaps

J.1 Vollständige Korrelationsmatrix

The correlation matrix of prompt ingredients is sorted by the correlation with the ranking (last column/row) in descending order

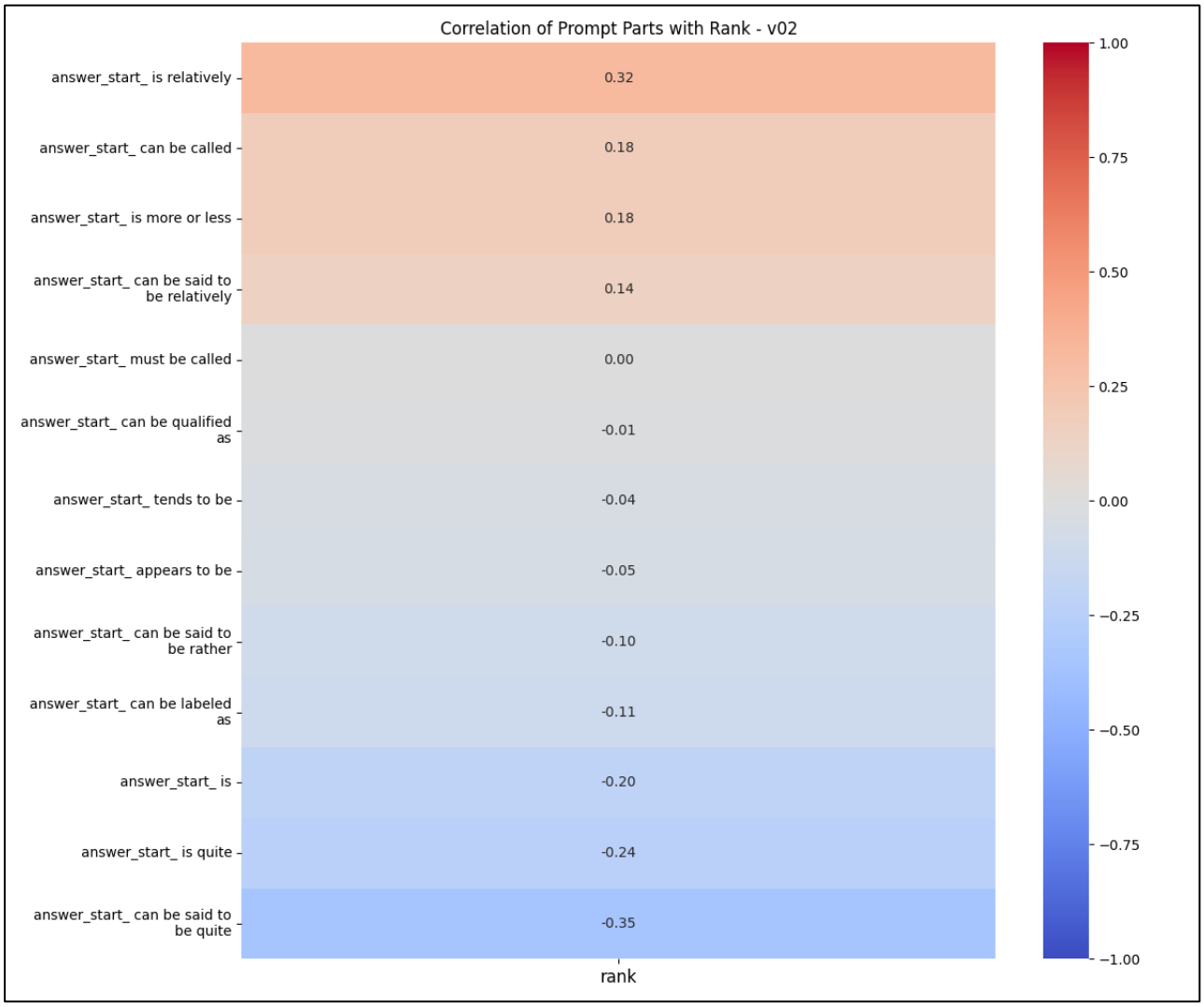


J.2 Korrelationsmatrix with ingredients having high correlation values (absolute value > 0.70)

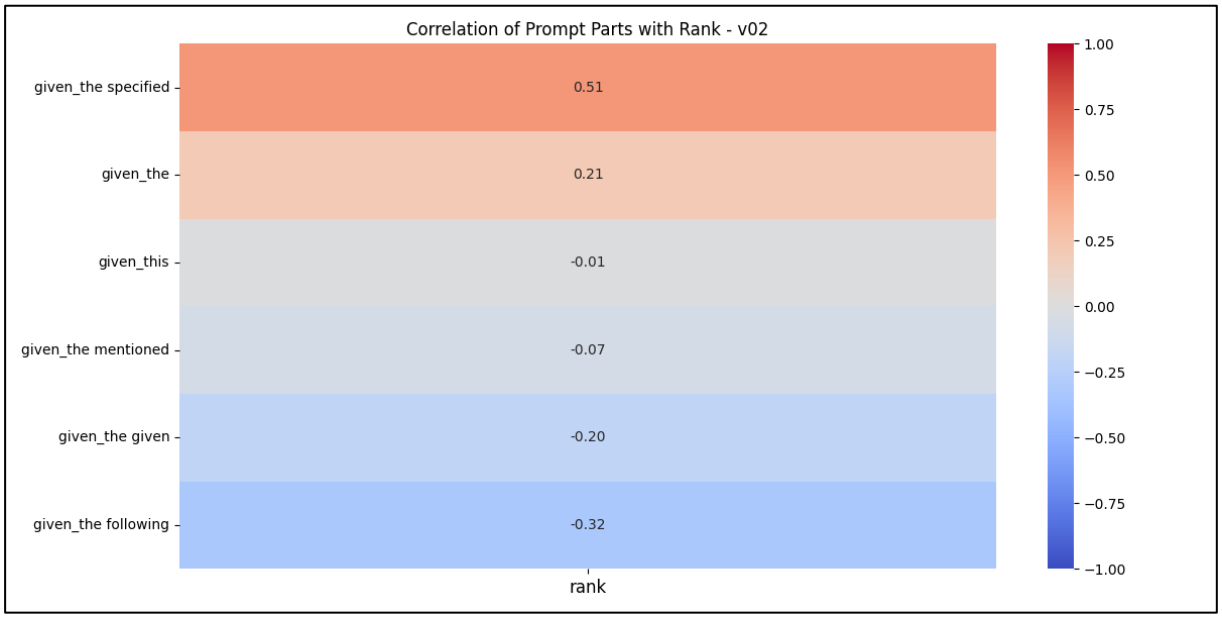


K PromptEngineeringStrategy2: Prompt Ingredients Heatmaps

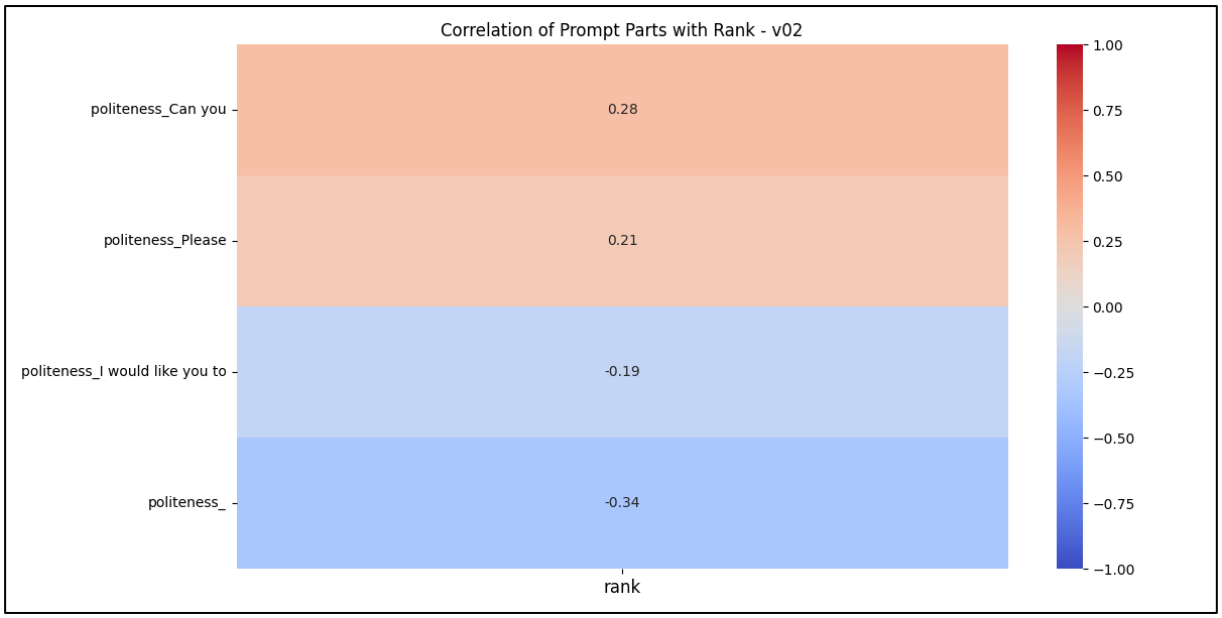
K.1 Answer_starts



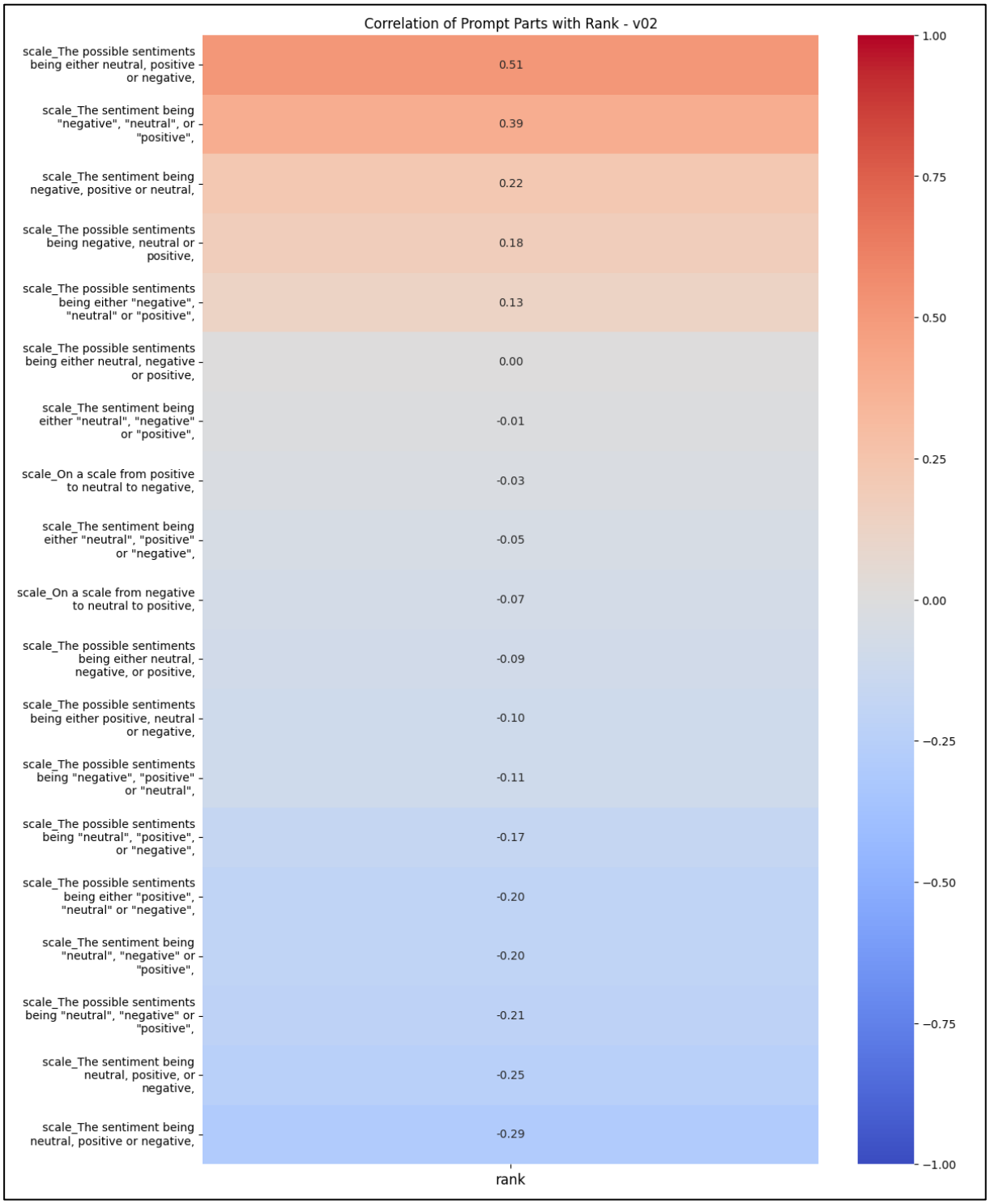
K.2 Givens



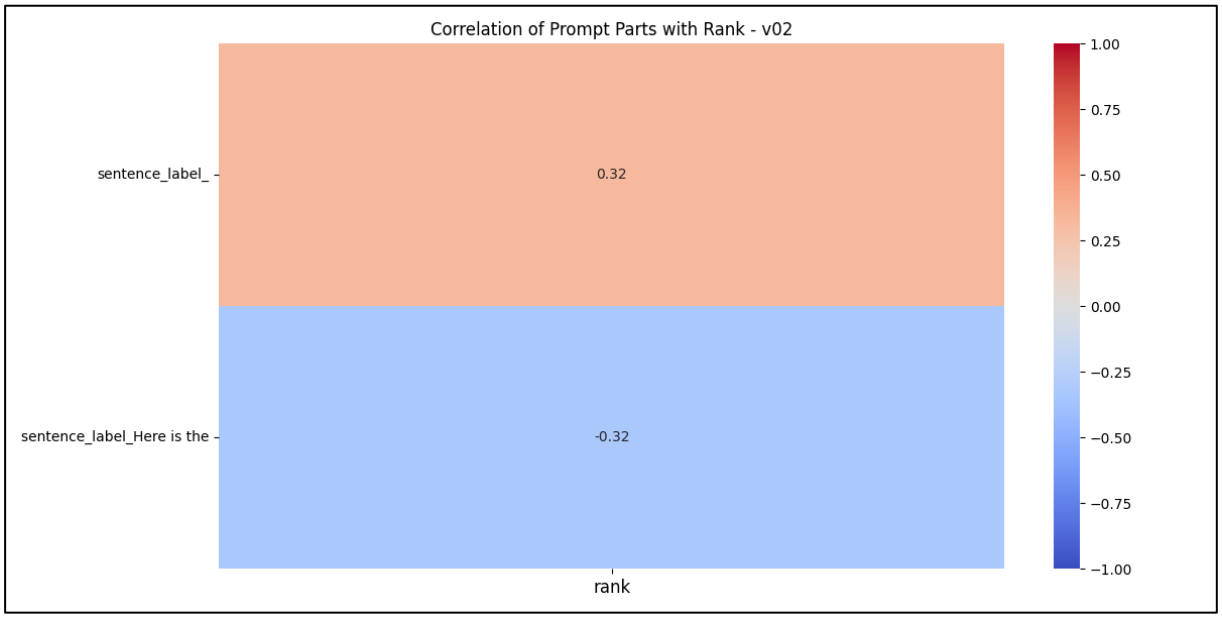
K.3 Politenesses



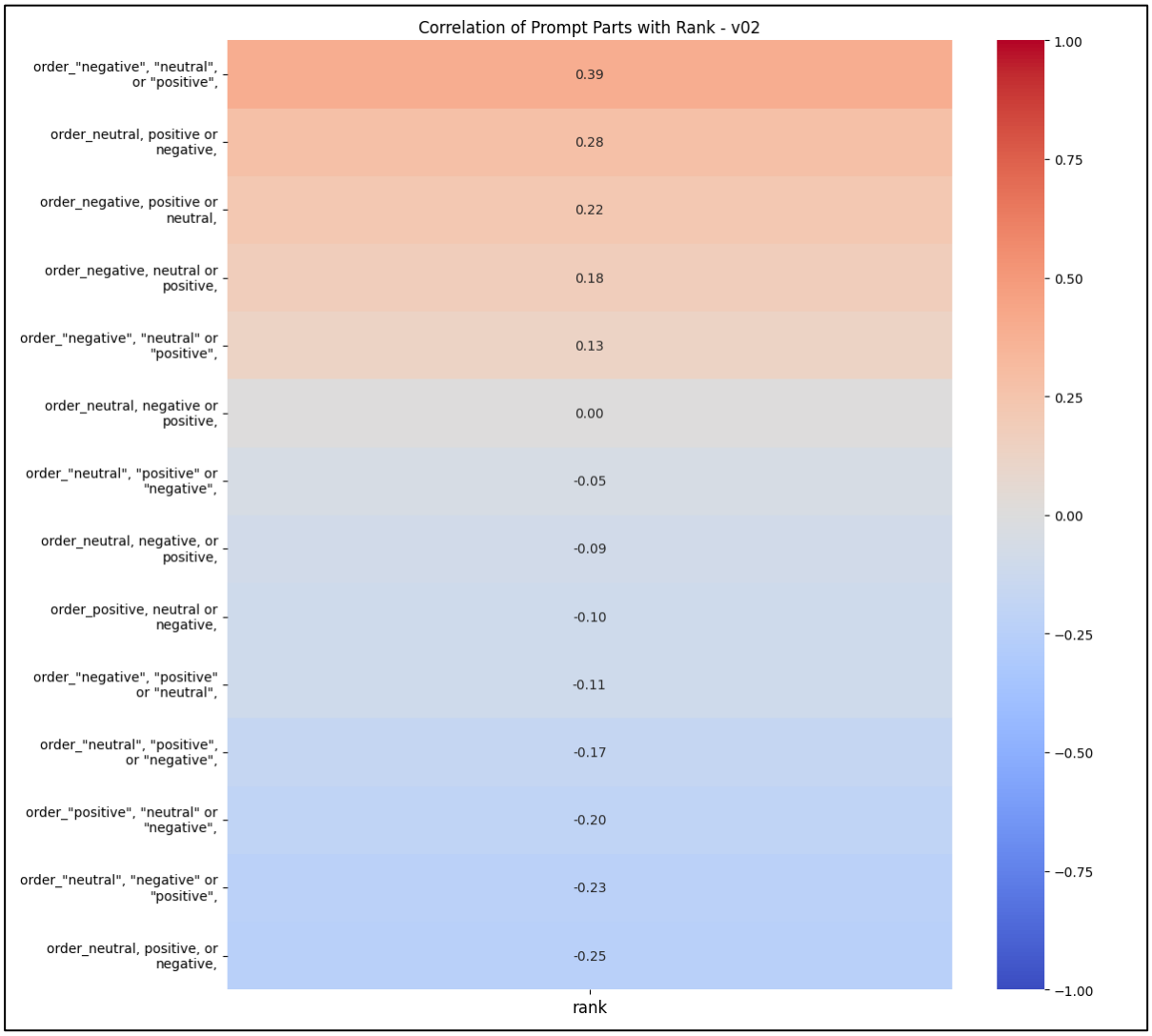
K.4 Scale


Correlation of Prompt Parts with Rank - v02
scale_The possible sentiments being either neutral, positive or negative,
scale_The sentiment being "negative", "neutral", or "positive",
scale_The sentiment being negative, positive or neutral,
scale_The possible sentiments being negative, neutral or positive,
scale_The possible sentiments being either "negative", "neutral" or "positive",
scale_The possible sentiments being either neutral, negative or positive,
scale_The sentiment being either "neutral", "negative" or "positive",
scale_On a scale from positive to neutral to negative,
scale_The sentiment being either "neutral", "positive" or "negative",
scale_On a scale from negative to neutral to positive,
scale_The possible sentiments being either neutral, negative, or positive,
scale_The possible sentiments being either positive, neutral or negative,
scale_The possible sentiments being "negative", "positive" or "neutral",
scale_The possible sentiments being "neutral", "positive", or "negative",
scale_The possible sentiments being either "positive", "neutral" or "negative",
scale_The sentiment being "neutral", "negative" or "positive",
scale_The possible sentiments being "neutral", "negative" or "positive",
scale_The sentiment being neutral, positive, or negative,
scale_The sentiment being neutral, positive or negative,
0.51
0.39
0.22
0.18
0.13
0.00
-0.01
-0.03
-0.05
-0.07
-0.09
-0.10
-0.11
-0.17
-0.20
-0.20
-0.21
-0.25
-0.29
rank
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

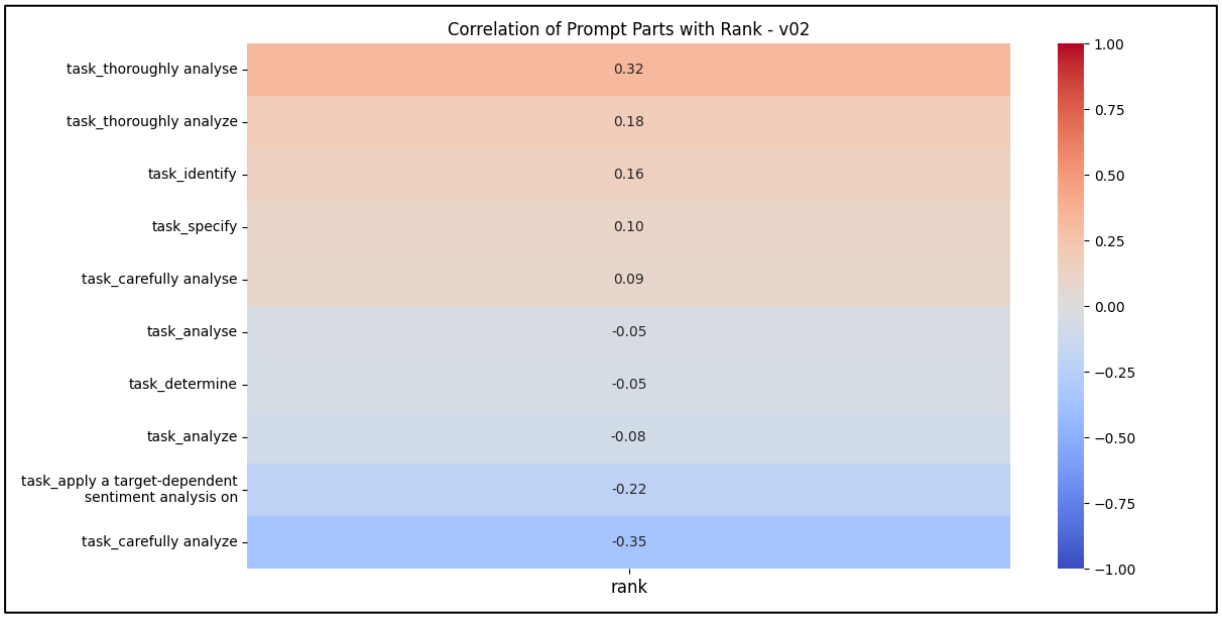
K.5 Sentence_labels



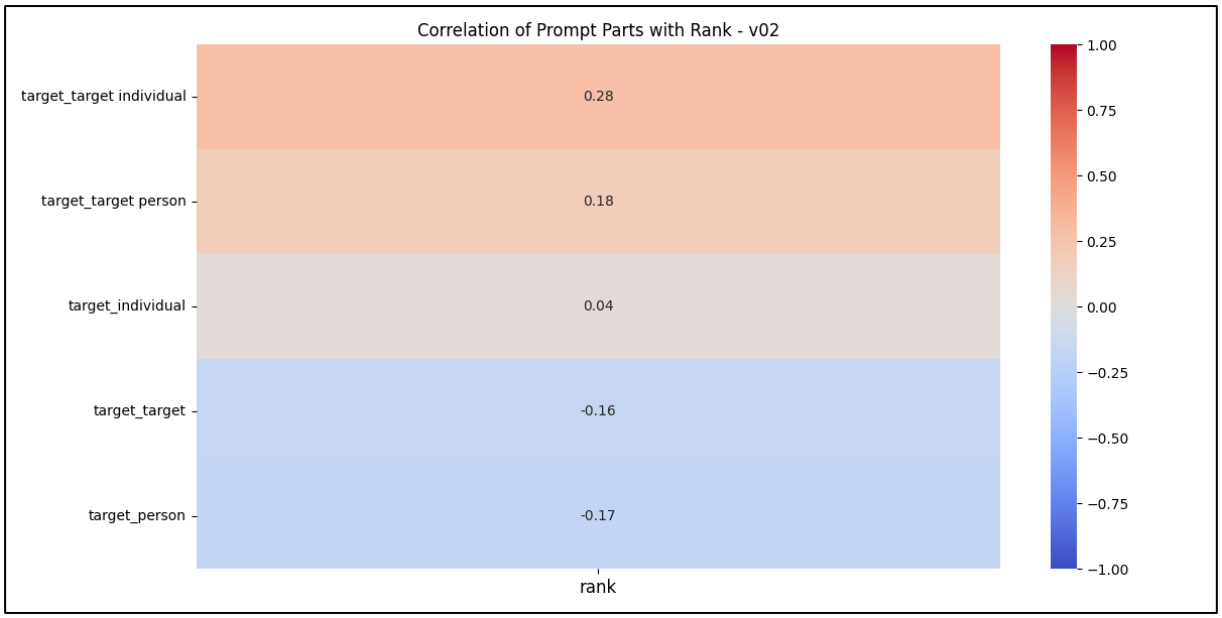
K.6 Sentiment_orders



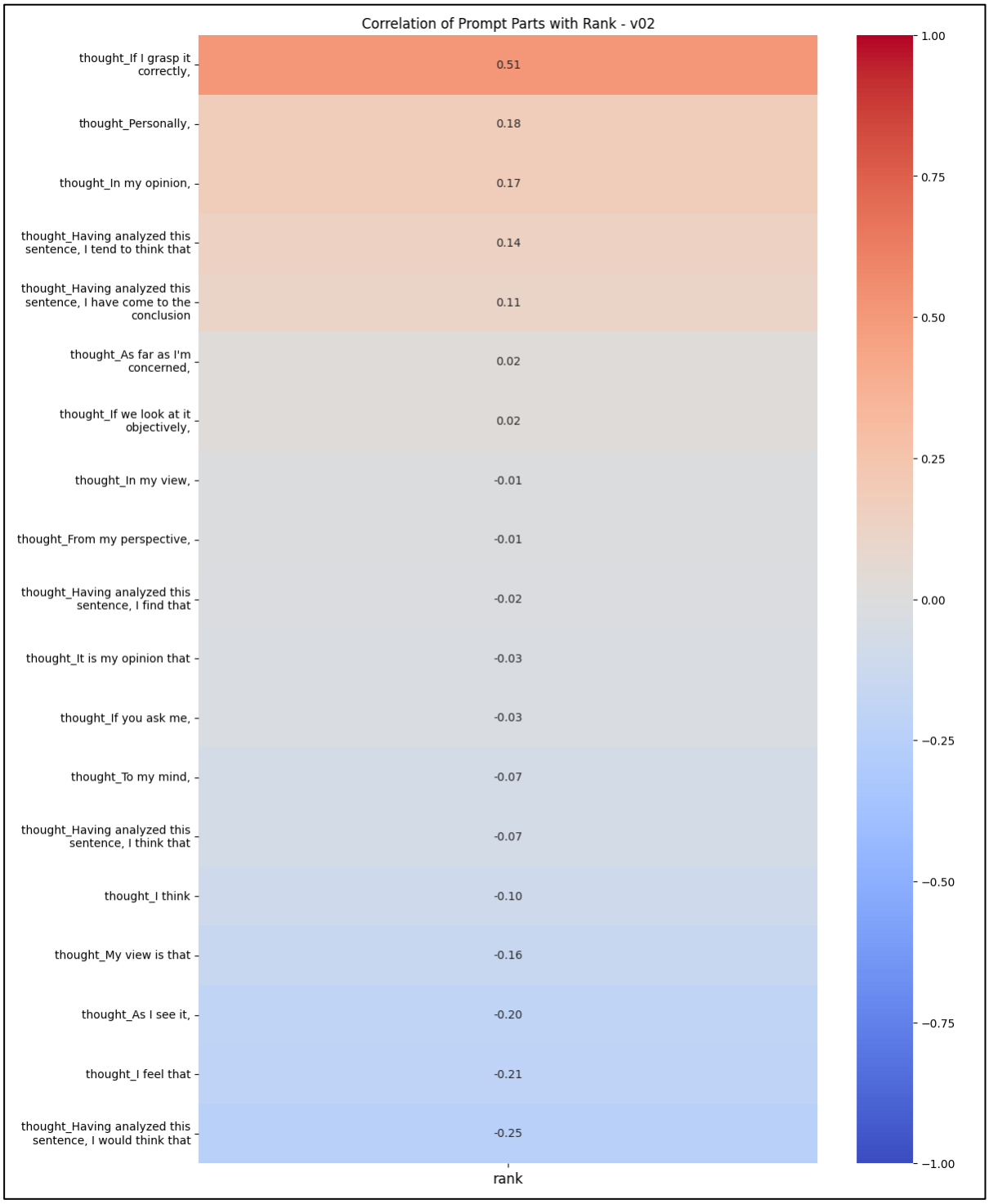
K.7 Tasks



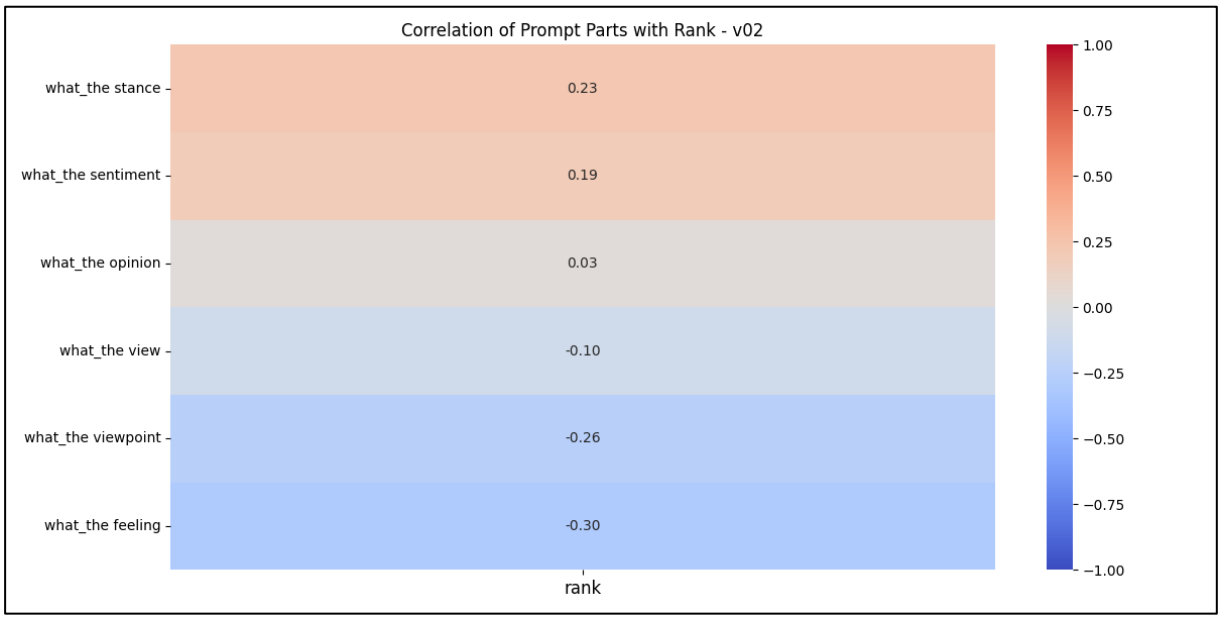
K.8 Targets



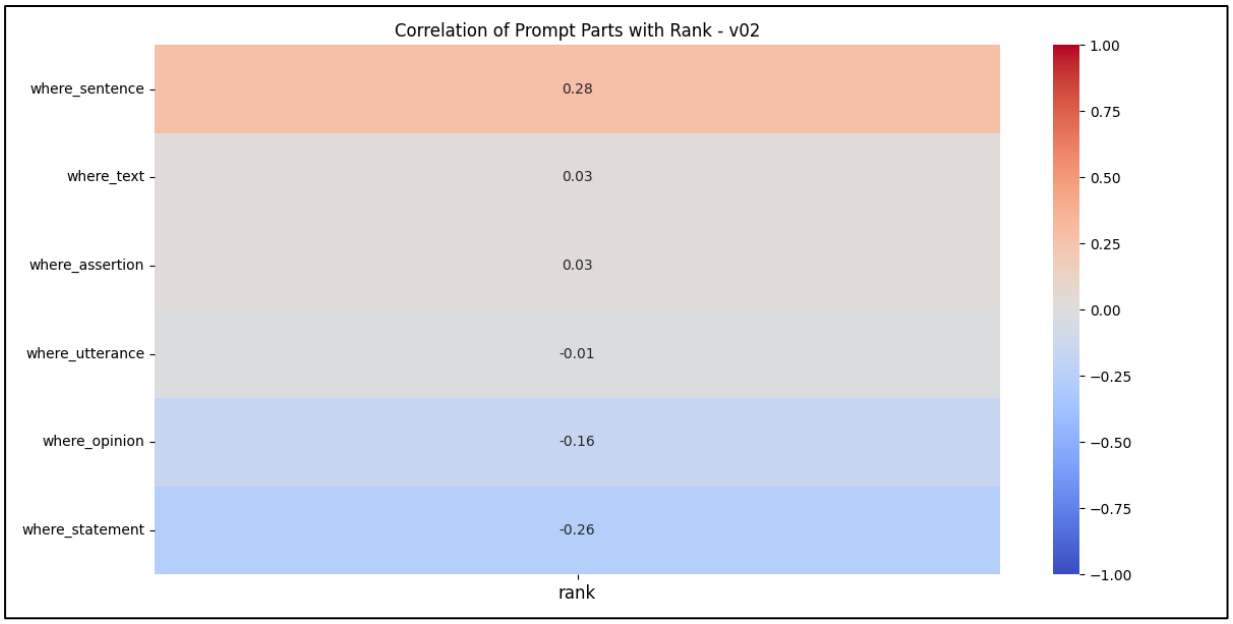
K.9 Thoughts


Correlation of Prompt Parts with Rank - v02
thought_If I grasp it correctly,
thought_Personally,
thought_In my opinion,
thought_Having analyzed this sentence, I tend to think that
thought_Having analyzed this sentence, I have come to the conclusion
thought_As far as I'm concerned,
thought_If we look at it objectively,
thought_In my view,
thought_From my perspective,
thought_Having analyzed this sentence, I find that
thought_It is my opinion that
thought_If you ask me,
thought_To my mind,
thought_Having analyzed this sentence, I think that
thought_I think
thought_My view is that
thought_As I see it,
thought_I feel that
thought_Having analyzed this sentence, I would think that
0.51
0.18
0.17
0.14
0.11
0.02
0.02
-0.01
-0.01
-0.02
-0.03
-0.03
-0.07
-0.07
-0.10
-0.16
-0.20
-0.21
-0.25
rank
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

K.10 Whats



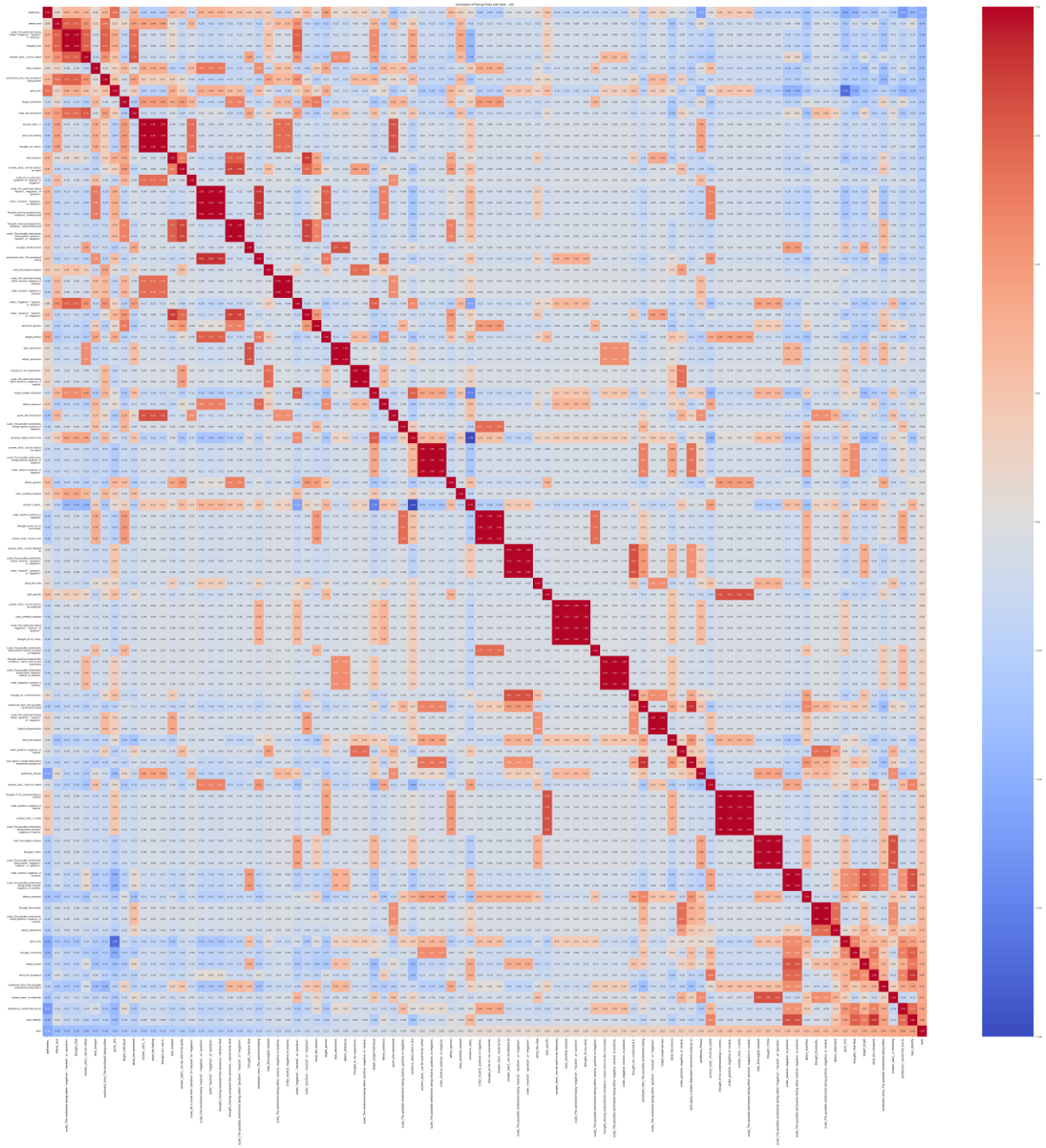
K.11 Wheres



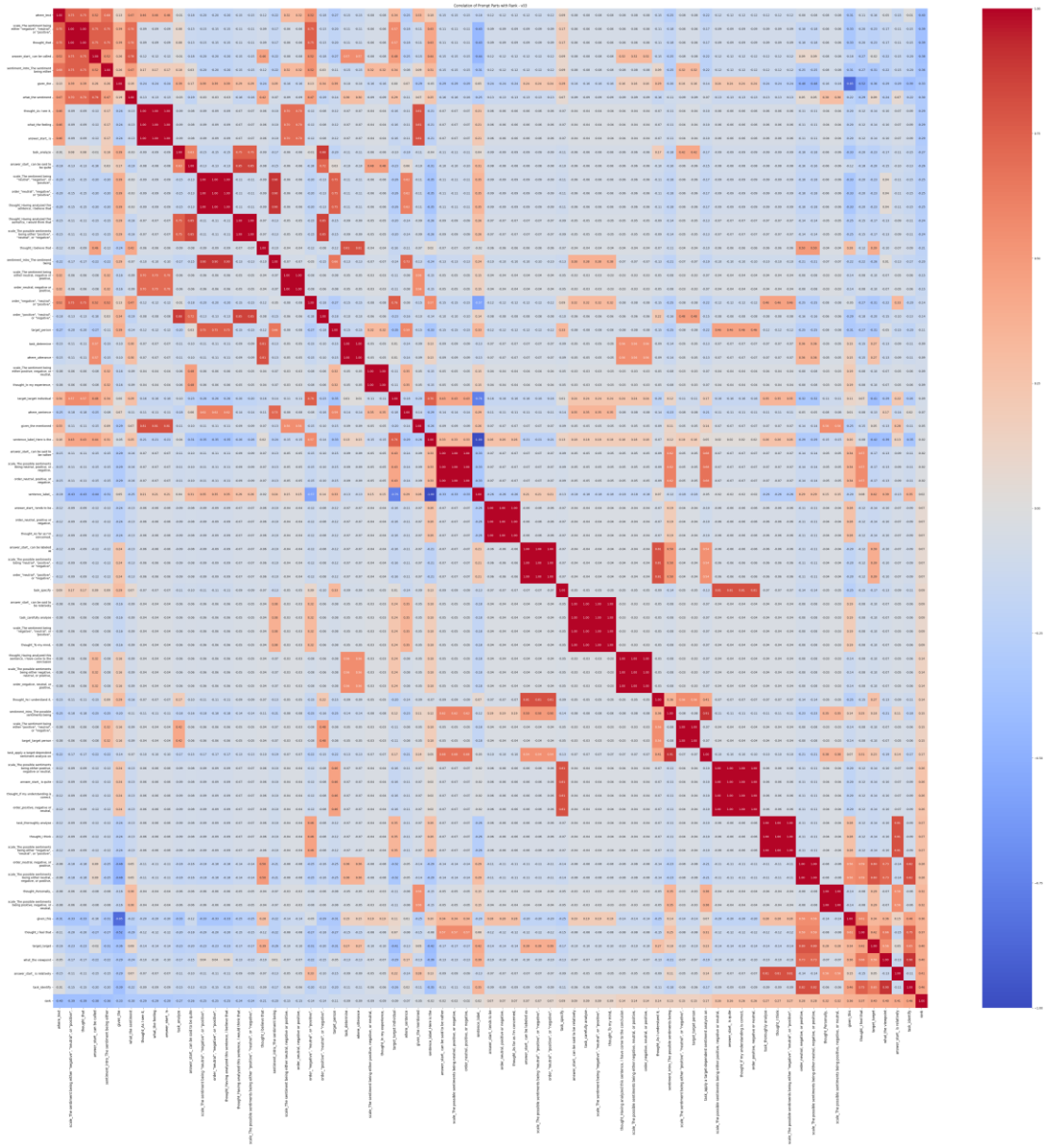
L PromptEngineeringStrategy3: Korrelation Heatmaps

L.1 Vollständige Korrelationsmatrix

The correlation matrix of prompt ingredients is sorted by the correlation with the ranking (last column/row) in descending order

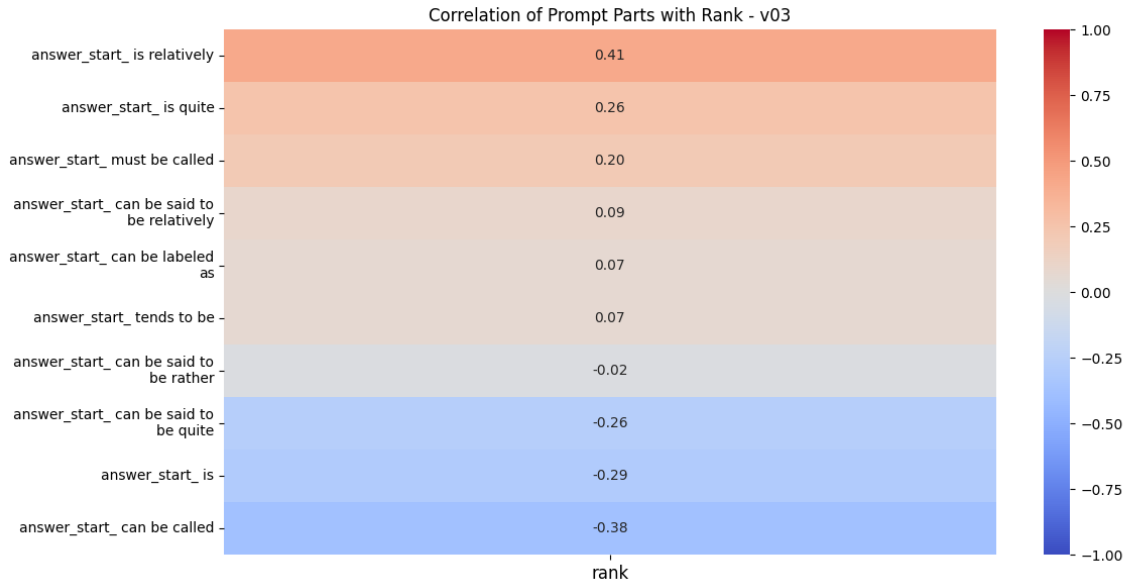


L.2 Korrelationsmatrix with ingredients having high correlation values (absolute value > 0.70)

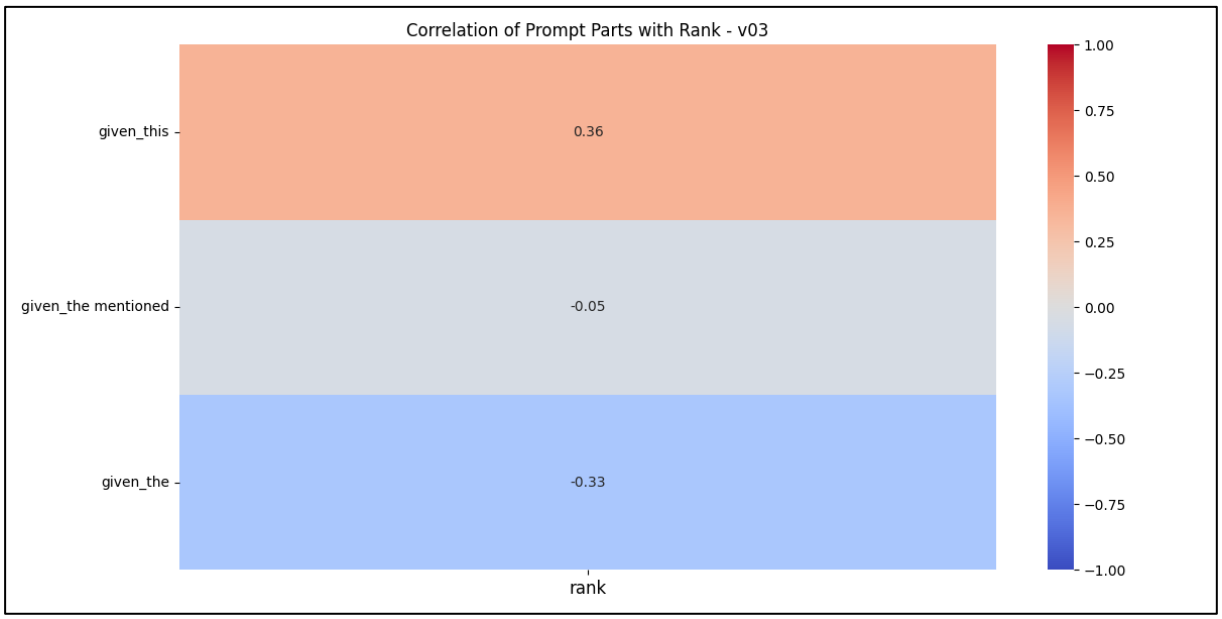


M PromptEngineeringStrategy3: Prompt Ingredients Heatmaps

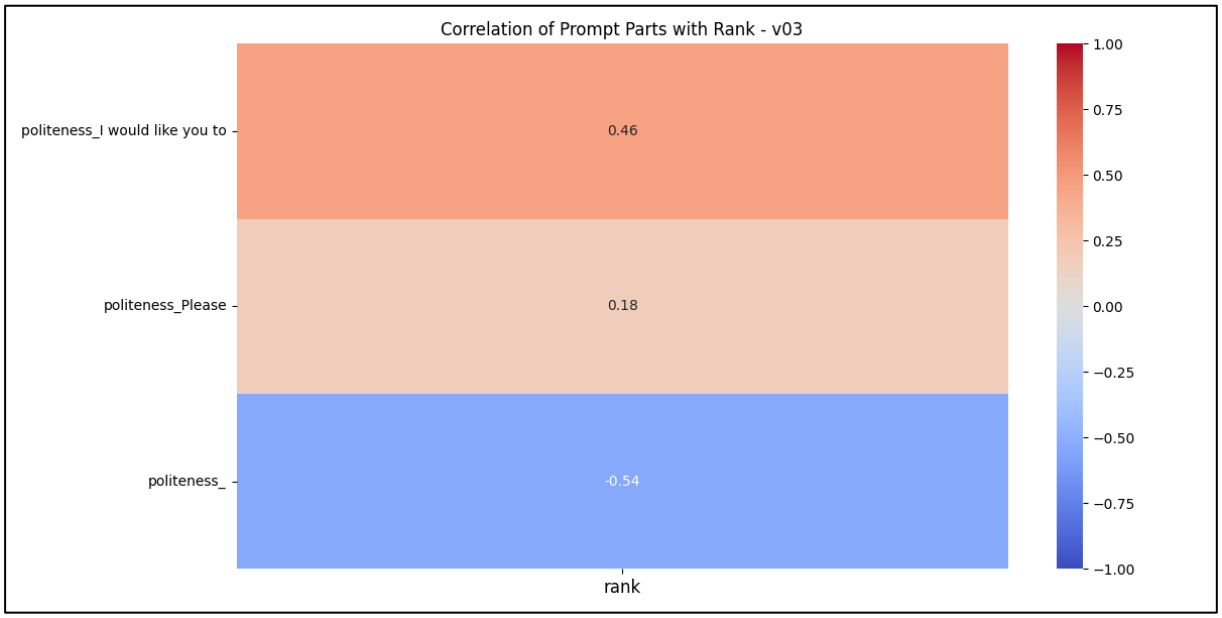
M.1 Answer_starts



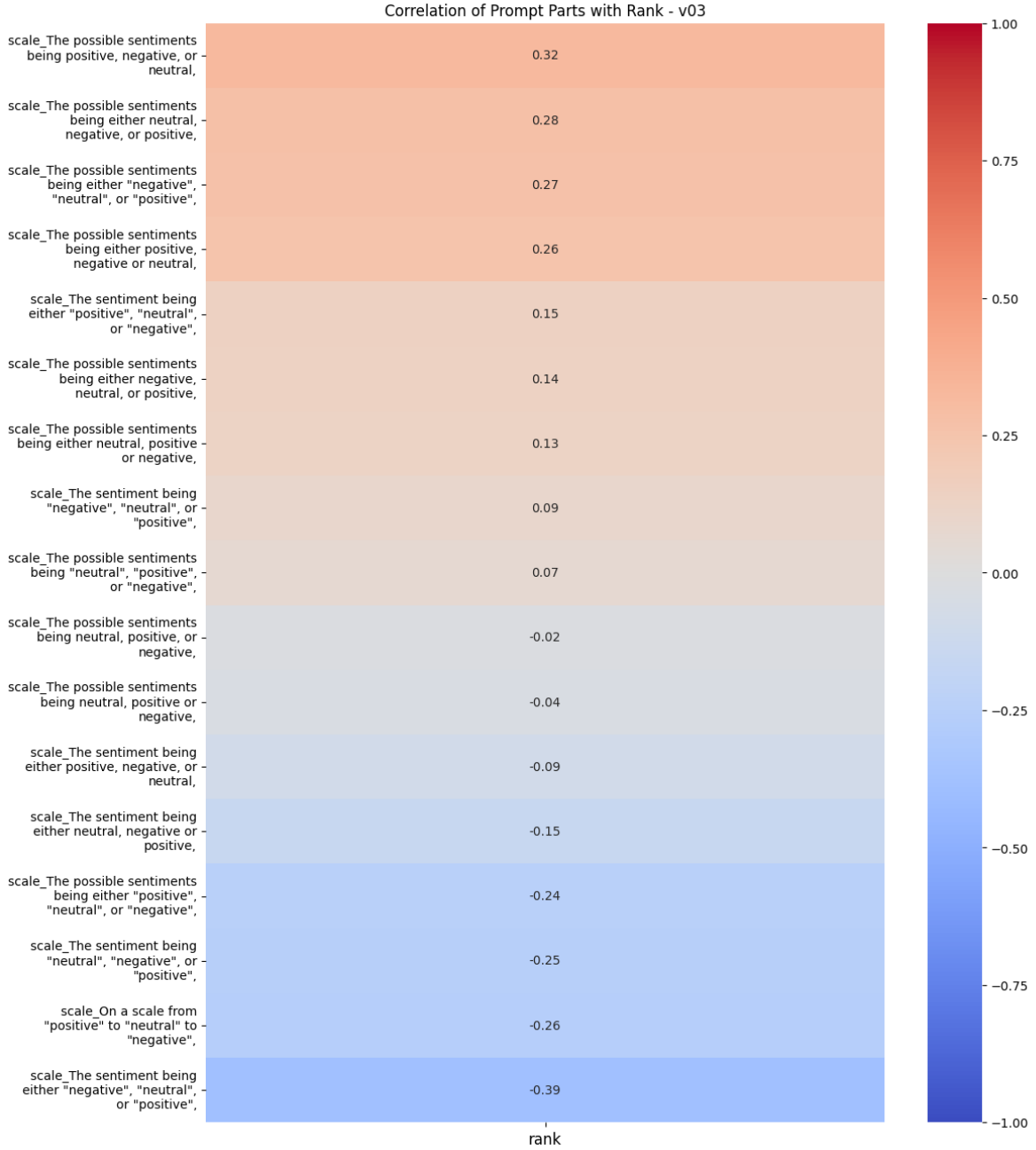
M.2 Givens



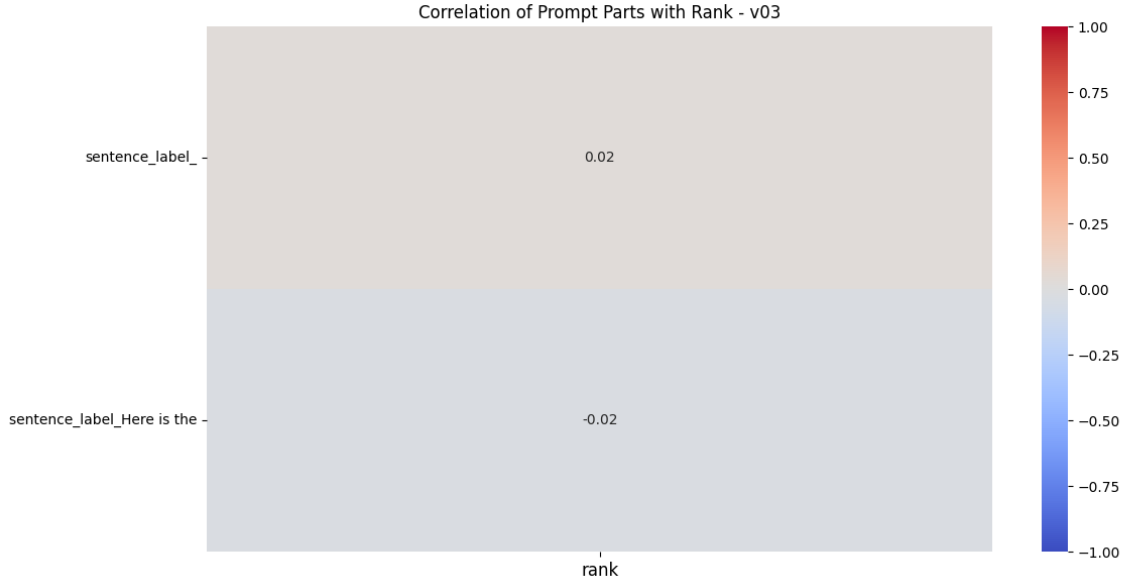
M.3 Politenesses



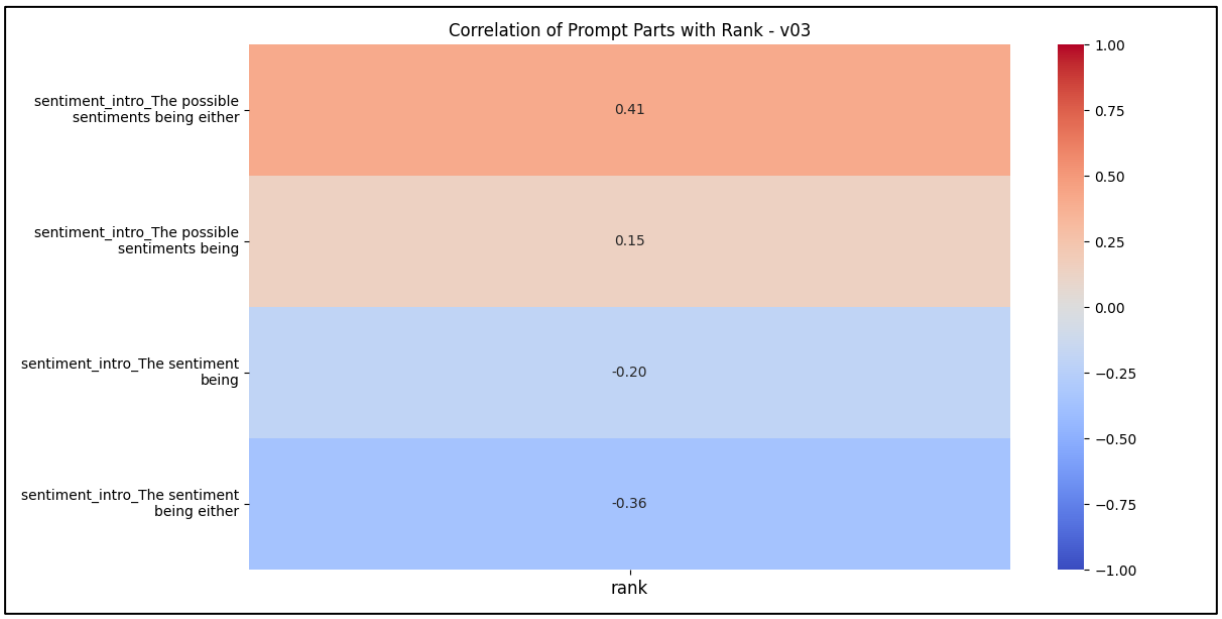
M.4 Scale


Correlation of Prompt Parts with Rank - v03
scale_The possible sentiments being positive, negative, or neutral,
scale_The possible sentiments being either neutral, negative, or positive,
scale_The possible sentiments being either "negative", "neutral", or "positive",
scale_The possible sentiments being either positive, negative or neutral,
scale_The sentiment being either "positive", "neutral", or "negative",
scale_The possible sentiments being either negative, neutral, or positive,
scale_The possible sentiments being either neutral, positive or negative,
scale_The sentiment being "negative", "neutral", or "positive",
scale_The possible sentiments being "neutral", "positive", or "negative",
scale_The possible sentiments being neutral, positive, or negative,
scale_The possible sentiments being neutral, positive or negative,
scale_The sentiment being either positive, negative, or neutral,
scale_The sentiment being either neutral, negative or positive,
scale_The possible sentiments being either "positive", "neutral", or "negative",
scale_The sentiment being "neutral", "negative", or "positive",
scale_On a scale from "positive" to "neutral" to "negative",
scale_The sentiment being either "negative", "neutral", or "positive",
0.32
0.28
0.27
0.26
0.15
0.14
0.13
0.09
0.07
-0.02
-0.04
-0.09
-0.15
-0.24
-0.25
-0.26
-0.39
rank
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

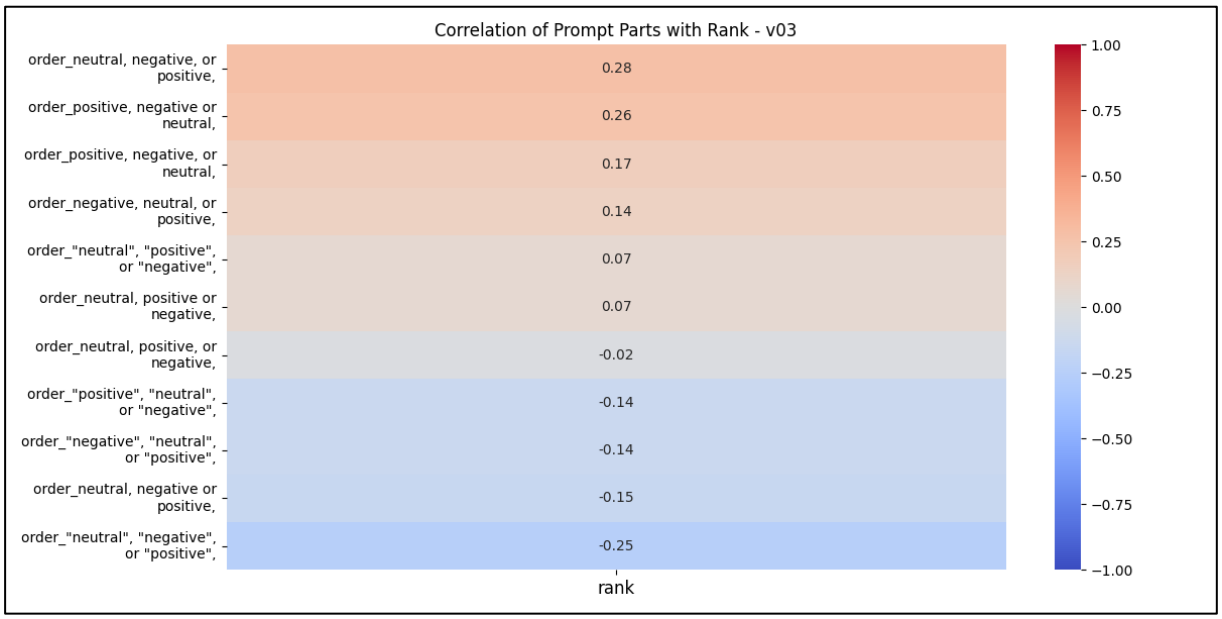
M.5 Sentence_labels



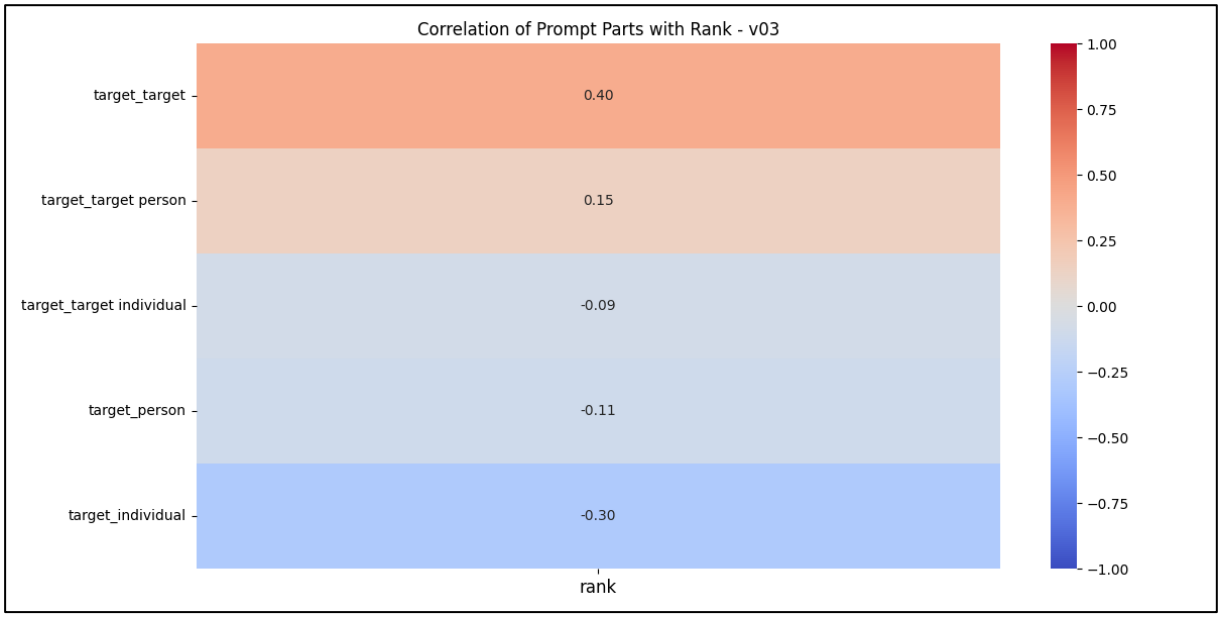
M.6 Sentiment_introductions



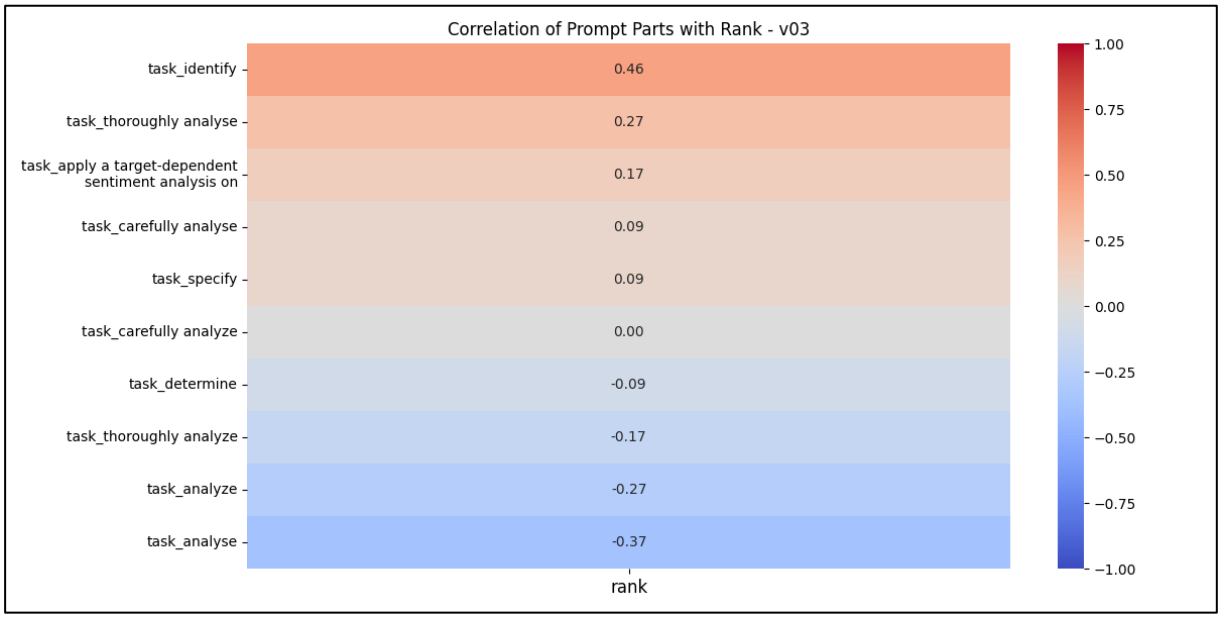
M.7 Sentiment_orders



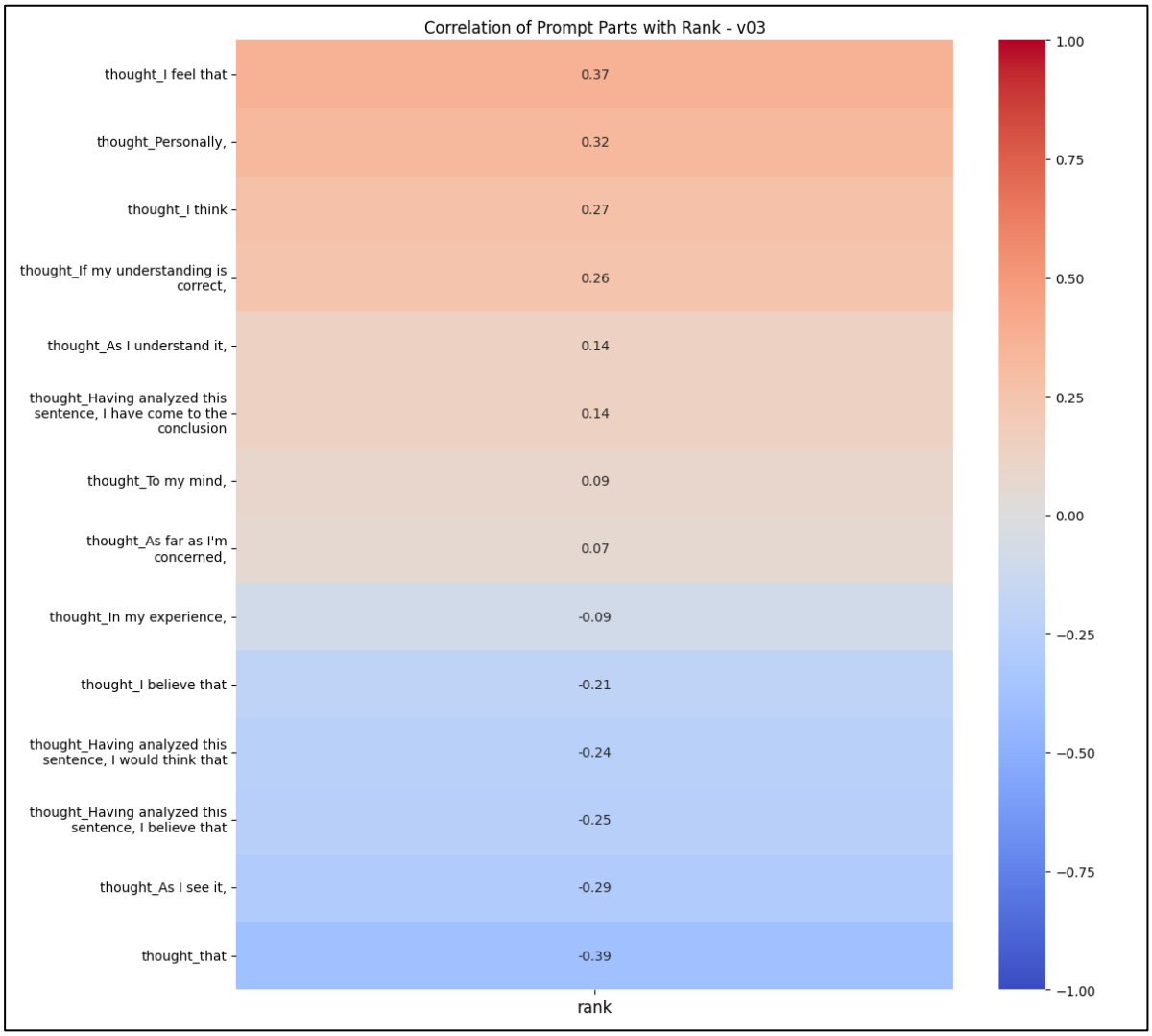
M.8 Targets



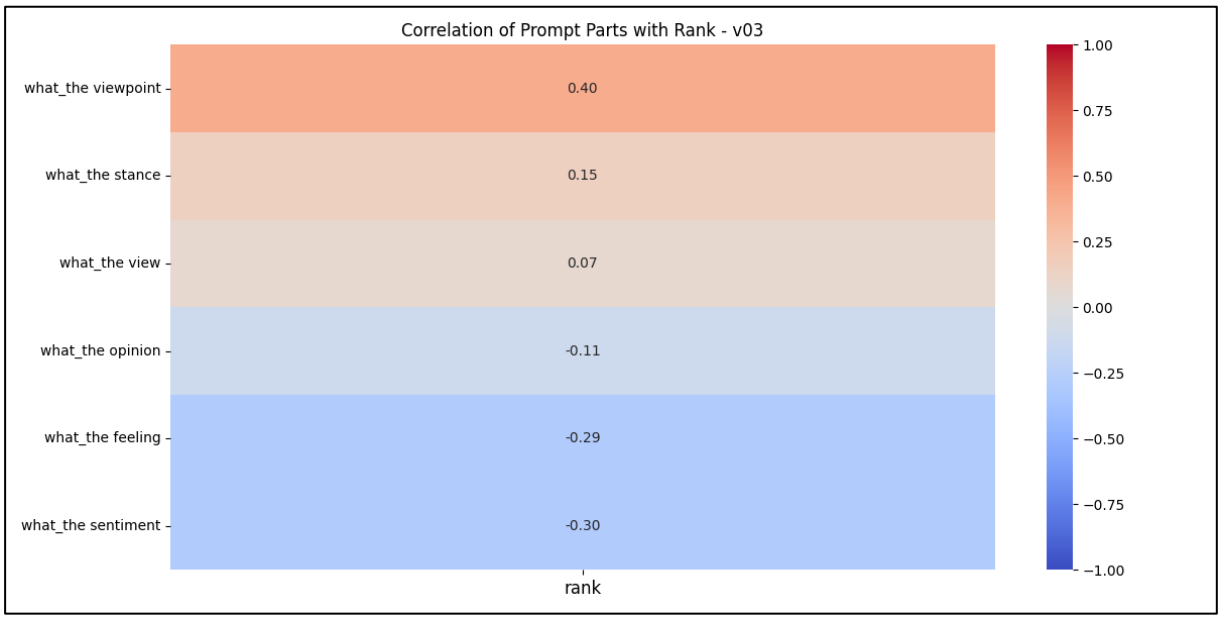
M.9 Tasks


Correlation of Prompt Parts with Rank - v03
task_identify
task_thoroughly analyse
task_apply a target-dependent sentiment analysis on
task_carefully analyse
task_specify
task_carefully analyze
task_determine
task_thoroughly analyze
task_analyze
task_analyse
0.46
0.27
0.17
0.09
0.09
0.00
-0.09
-0.17
-0.27
-0.37
rank
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

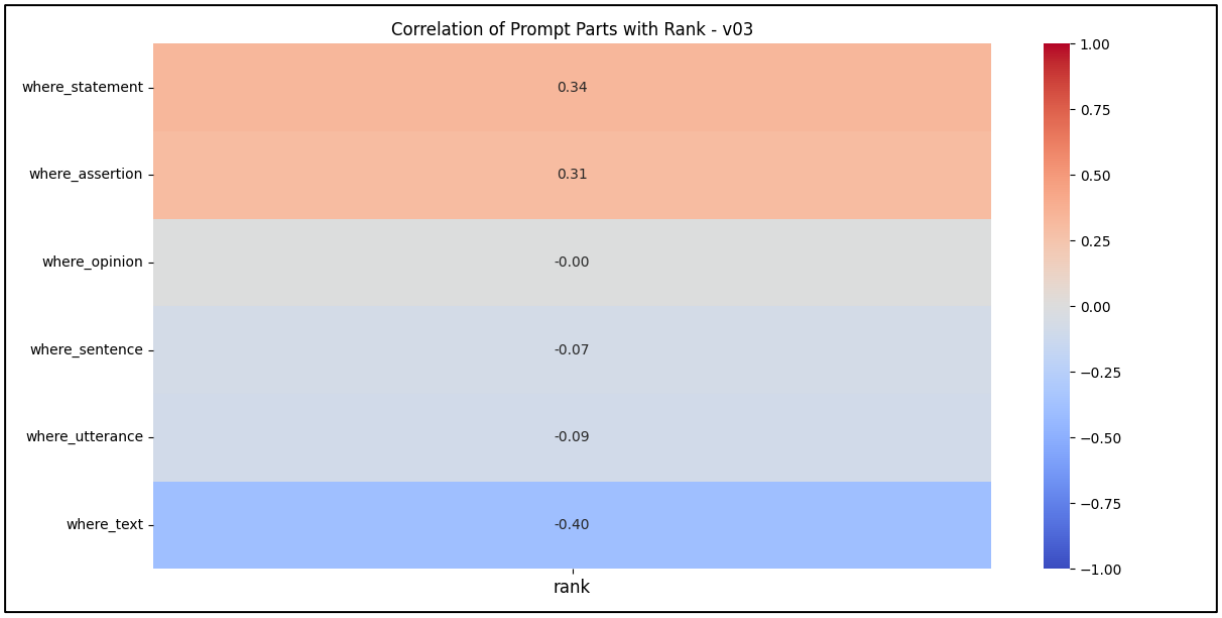
M.10 Thoughts



M.11 Whats



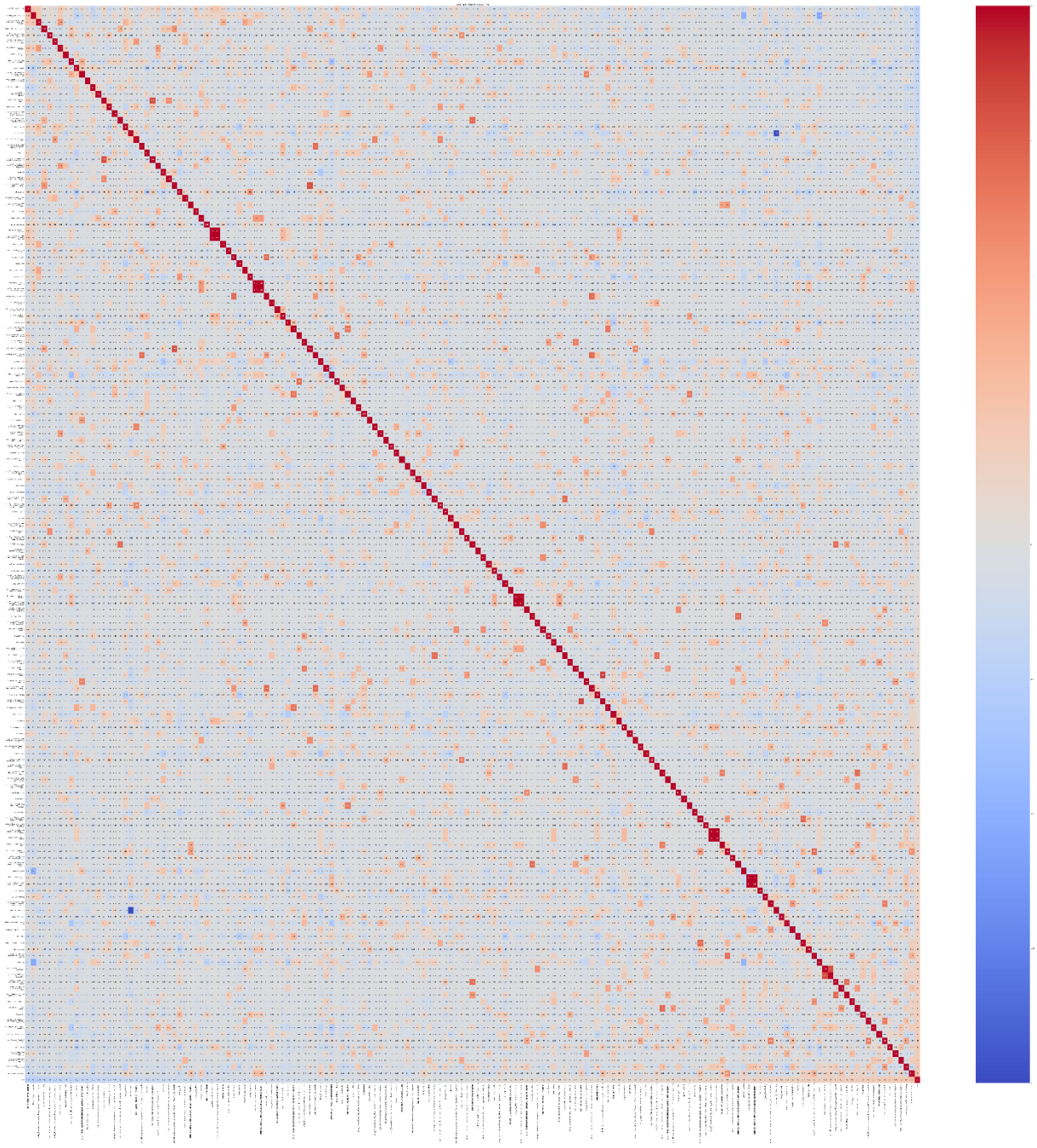
M.12 Wheres



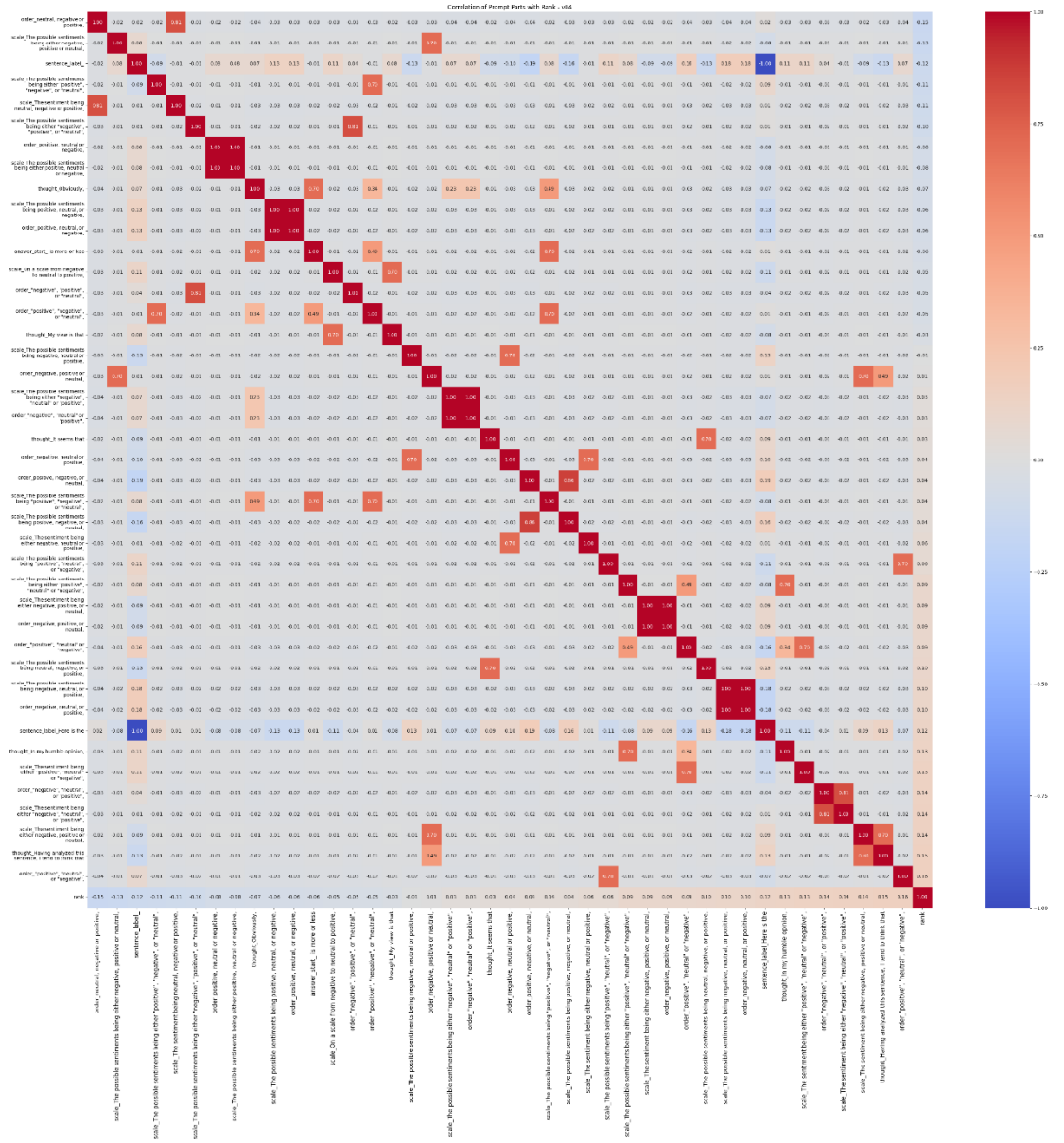
N PromptEngineeringStrategy4: Korrelation Heatmaps

N.1 Vollständige Korrelationsmatrix

The correlation matrix of prompt ingredients is sorted by the correlation with the ranking (last column/row) in descending order

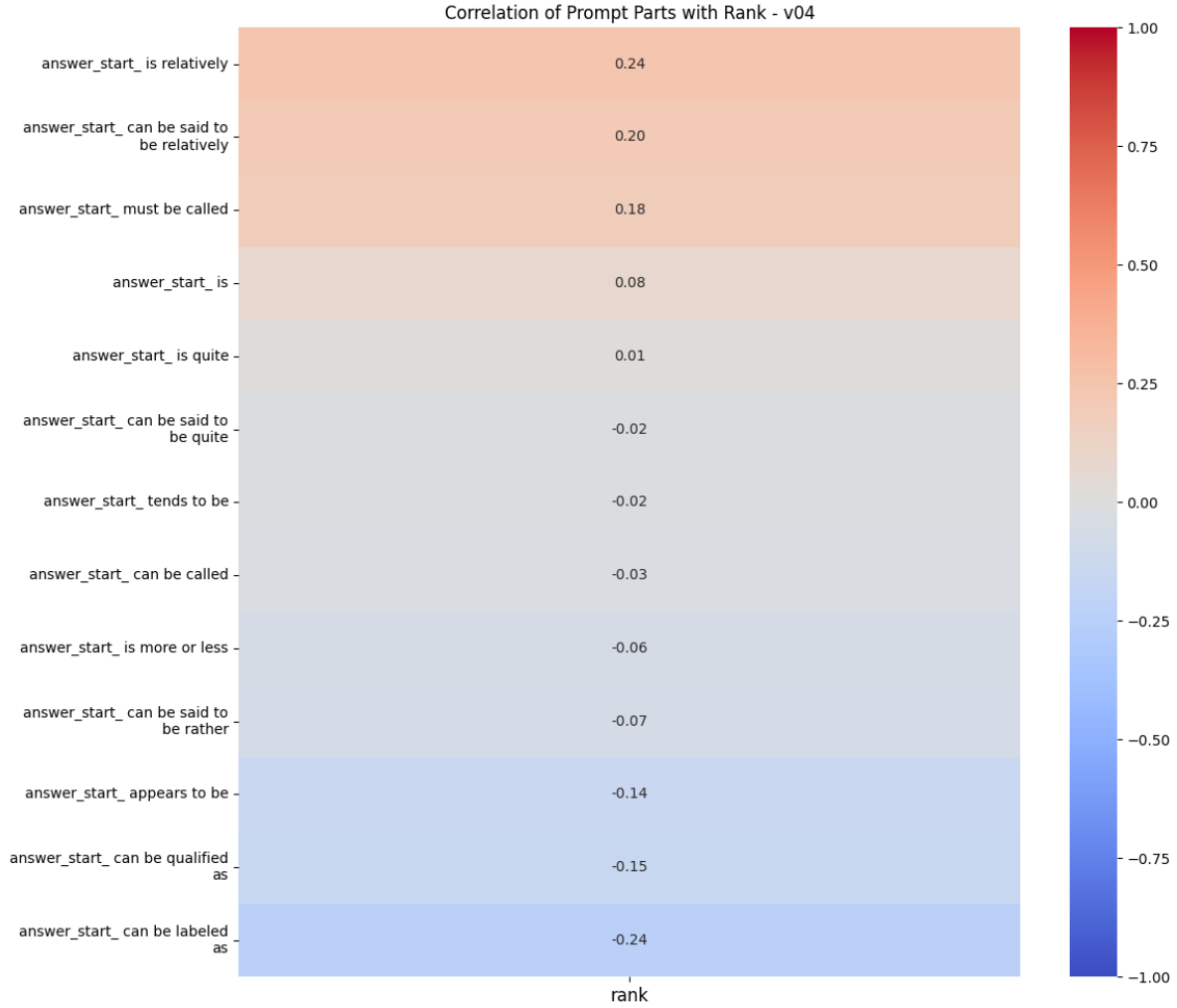


N.2 Korrelationsmatrix with ingredients having high correlation values (absolute value > 0.70)

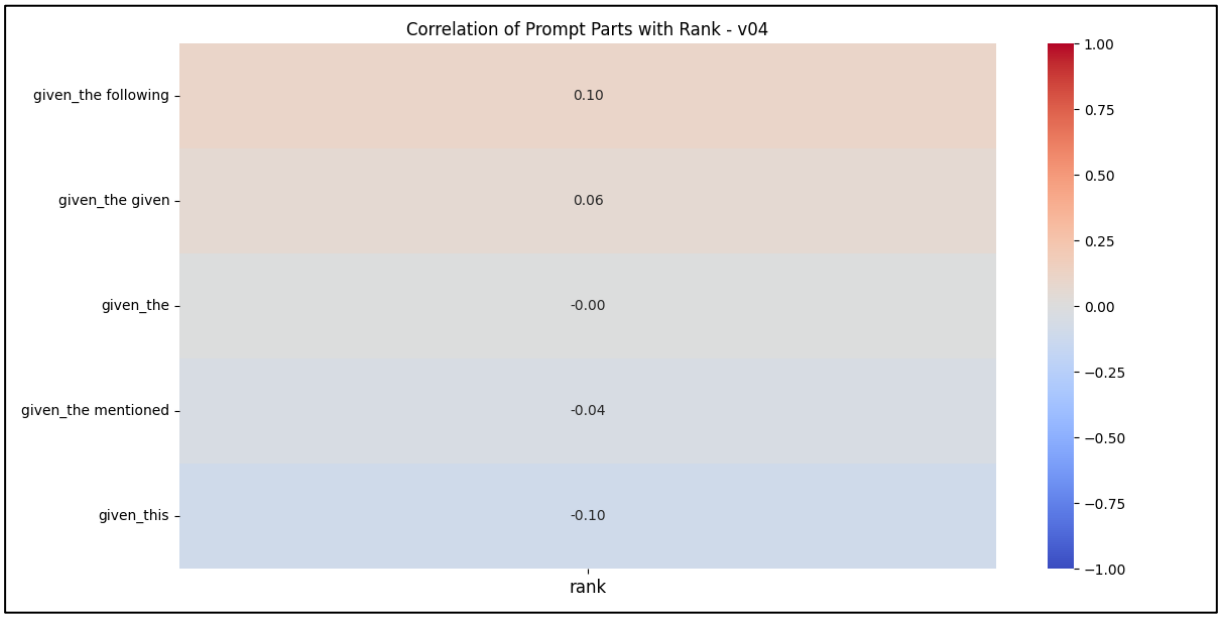


O PromptEngineeringStrategy4: Prompt Ingredients Heatmaps

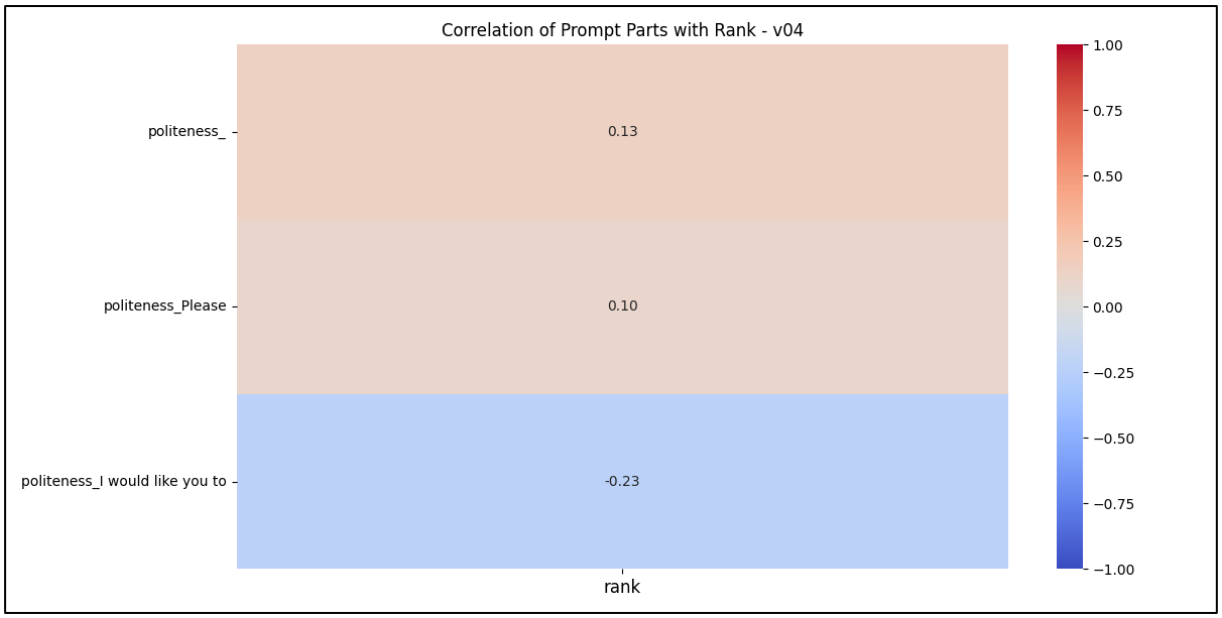
O.1 Answer_starts



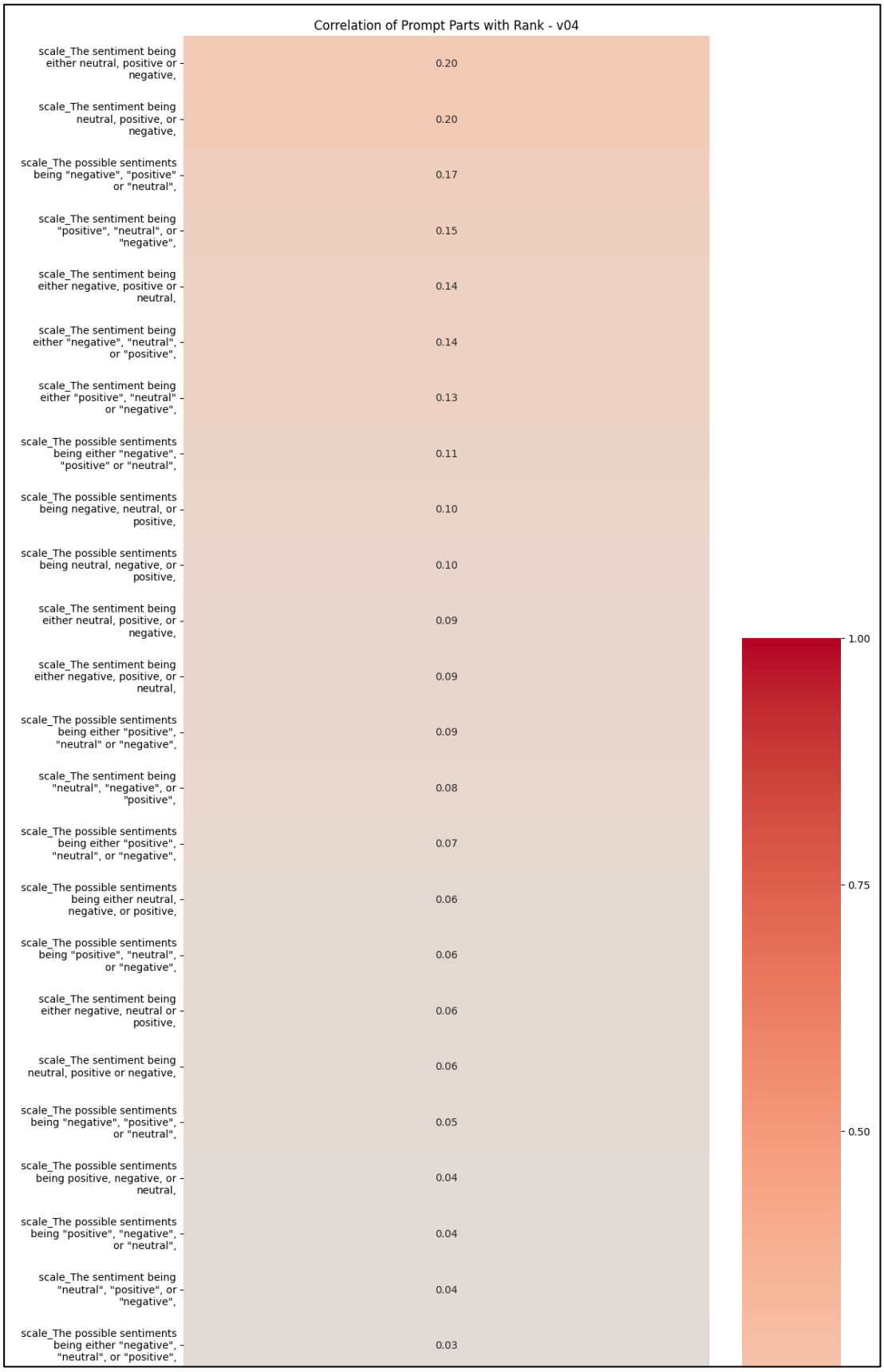
O.2 Givens



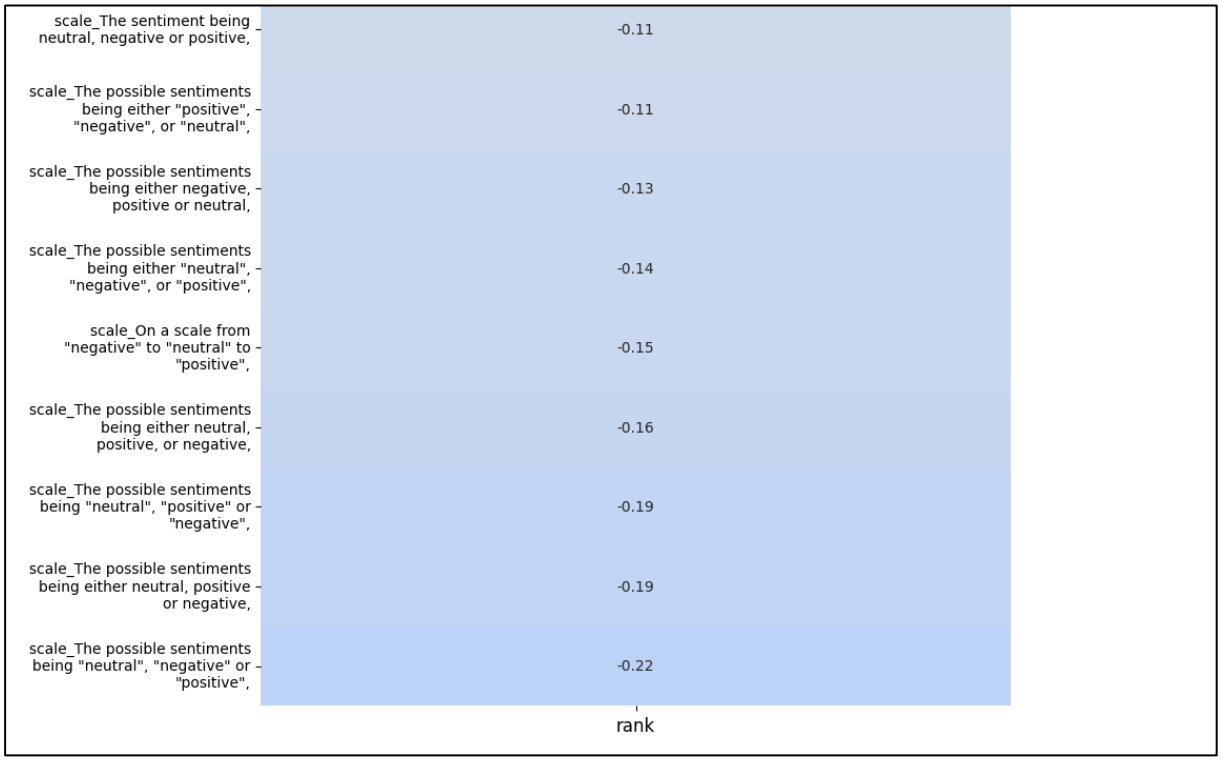
O.3 Politenesses



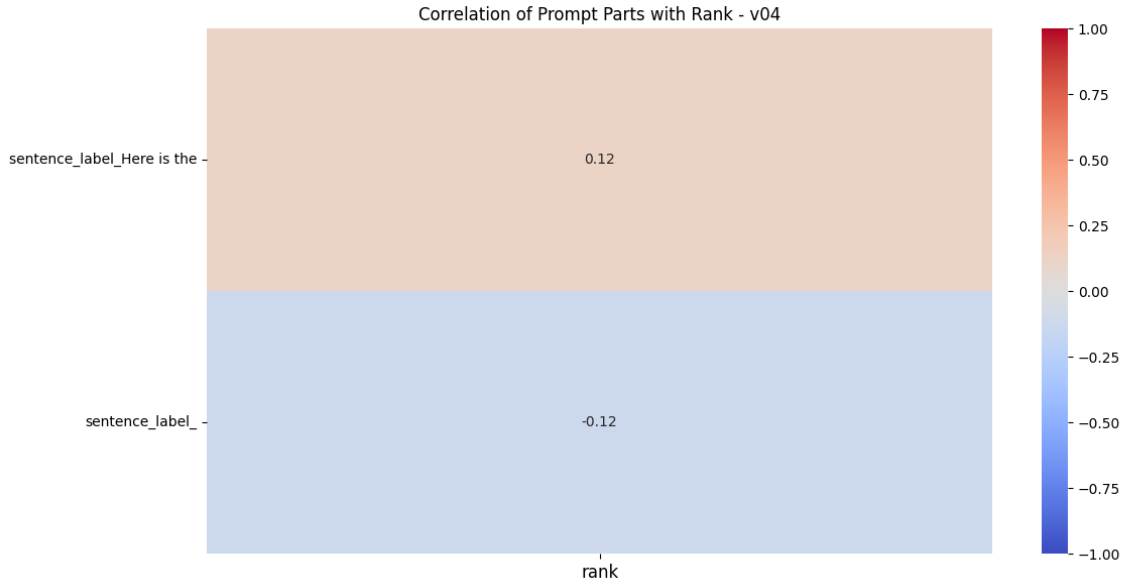
O.4 Scale


Correlation of Prompt Parts with Rank - v04
scale_The sentiment being either neutral, positive or negative, 0.20
scale_The sentiment being neutral, positive, or negative, 0.20
scale_The possible sentiments being "negative", "positive" or "neutral", 0.17
scale_The sentiment being "positive", "neutral", or "negative", 0.15
scale_The sentiment being either negative, positive or neutral, 0.14
scale_The sentiment being either "negative", "neutral", or "positive", 0.14
scale_The sentiment being either "positive", "neutral" or "negative", 0.13
scale_The possible sentiments being either "negative", "positive" or "neutral", 0.11
scale_The possible sentiments being negative, neutral, or positive, 0.10
scale_The possible sentiments being neutral, negative, or positive, 0.10
scale_The sentiment being either neutral, positive, or negative, 0.09
scale_The sentiment being either negative, positive, or neutral, 0.09
scale_The possible sentiments being either "positive", "neutral" or "negative", 0.09
scale_The sentiment being "neutral", "negative", or "positive", 0.08
scale_The possible sentiments being either "positive", "neutral", or "negative", 0.07
scale_The possible sentiments being either neutral, negative, or positive, 0.06
scale_The possible sentiments being "positive", "neutral", or "negative", 0.06
scale_The sentiment being either negative, neutral or positive, 0.06
scale_The sentiment being neutral, positive or negative, 0.06
scale_The possible sentiments being "negative", "positive", or "neutral", 0.05
scale_The possible sentiments being positive, negative, or neutral, 0.04
scale_The possible sentiments being "positive", "negative", or "neutral", 0.04
scale_The sentiment being "neutral", "positive", or "negative", 0.04
scale_The possible sentiments being either "negative", "neutral", or "positive", 0.03
1.00
0.75
0.50

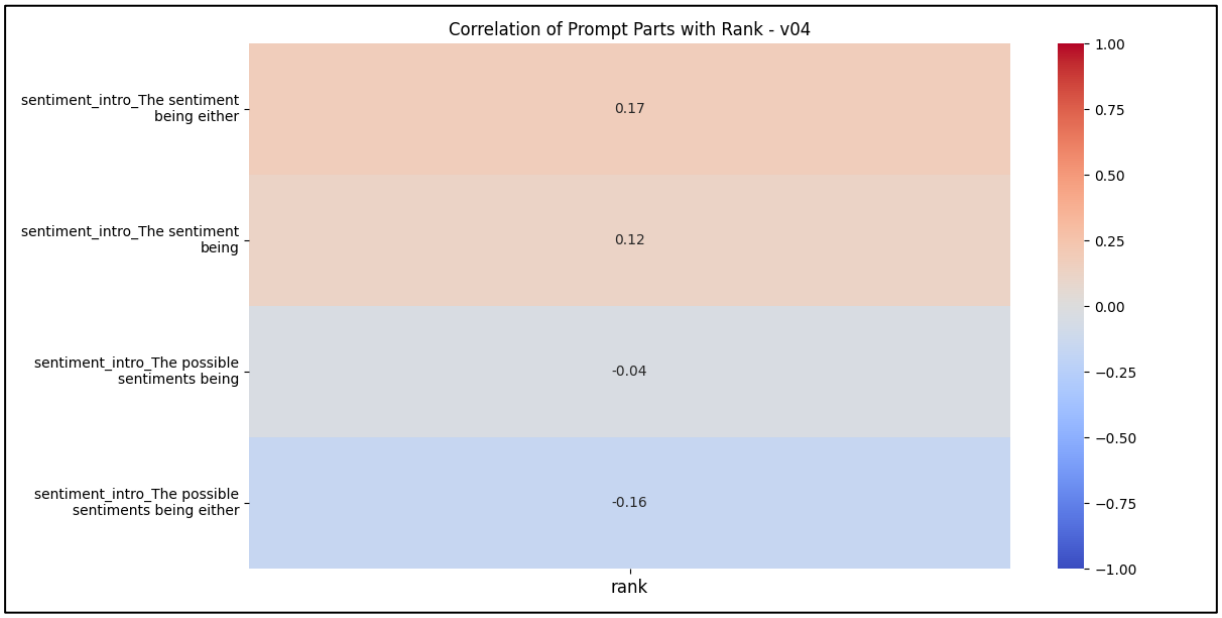

scale_The sentiment being either "negative", "positive" or "neutral",
0.03
scale_The possible sentiments being either "negative", "neutral" or "positive",
0.03
scale_The possible sentiments being either "neutral", "negative" or "positive",
0.02
scale_The possible sentiments being "neutral", "positive", or "negative",
0.01
scale_On a scale from "positive" to "neutral" to "negative",
0.01
scale_The possible sentiments being either positive, negative, or neutral,
0.01
scale_The possible sentiments being either "neutral", "positive", or "negative",
0.00
scale_The possible sentiments being either "neutral", "positive" or "negative",
-0.01
scale_The possible sentiments being negative, neutral or positive,
-0.01
scale_The possible sentiments being neutral, positive, or negative,
-0.01
scale_The possible sentiments being neutral, positive or negative,
-0.02
scale_The sentiment being "neutral", "negative" or "positive",
-0.02
scale_The sentiment being either "neutral", "positive" or "negative",
-0.02
scale_The sentiment being either neutral, negative, or positive,
-0.03
scale_The sentiment being neutral, negative, or positive,
-0.05
scale_On a scale from negative to neutral to positive,
-0.05
scale_The possible sentiments being "neutral", "negative", or "positive",
-0.06
scale_On a scale from positive to neutral to negative,
-0.06
scale_The possible sentiments being positive, neutral, or negative,
-0.06
scale_The possible sentiments being either positive, neutral or negative,
-0.08
scale_The possible sentiments being "positive", "neutral" or "negative",
-0.09
scale_The possible sentiments being either "negative", "positive", or "neutral",
-0.10
scale_The sentiment being either neutral, negative or positive,
-0.10
scale_The sentiment being either "neutral", "negative" or "positive",
-0.11
0.25
0.00
−0.25
−0.50
−0.75
−1.00



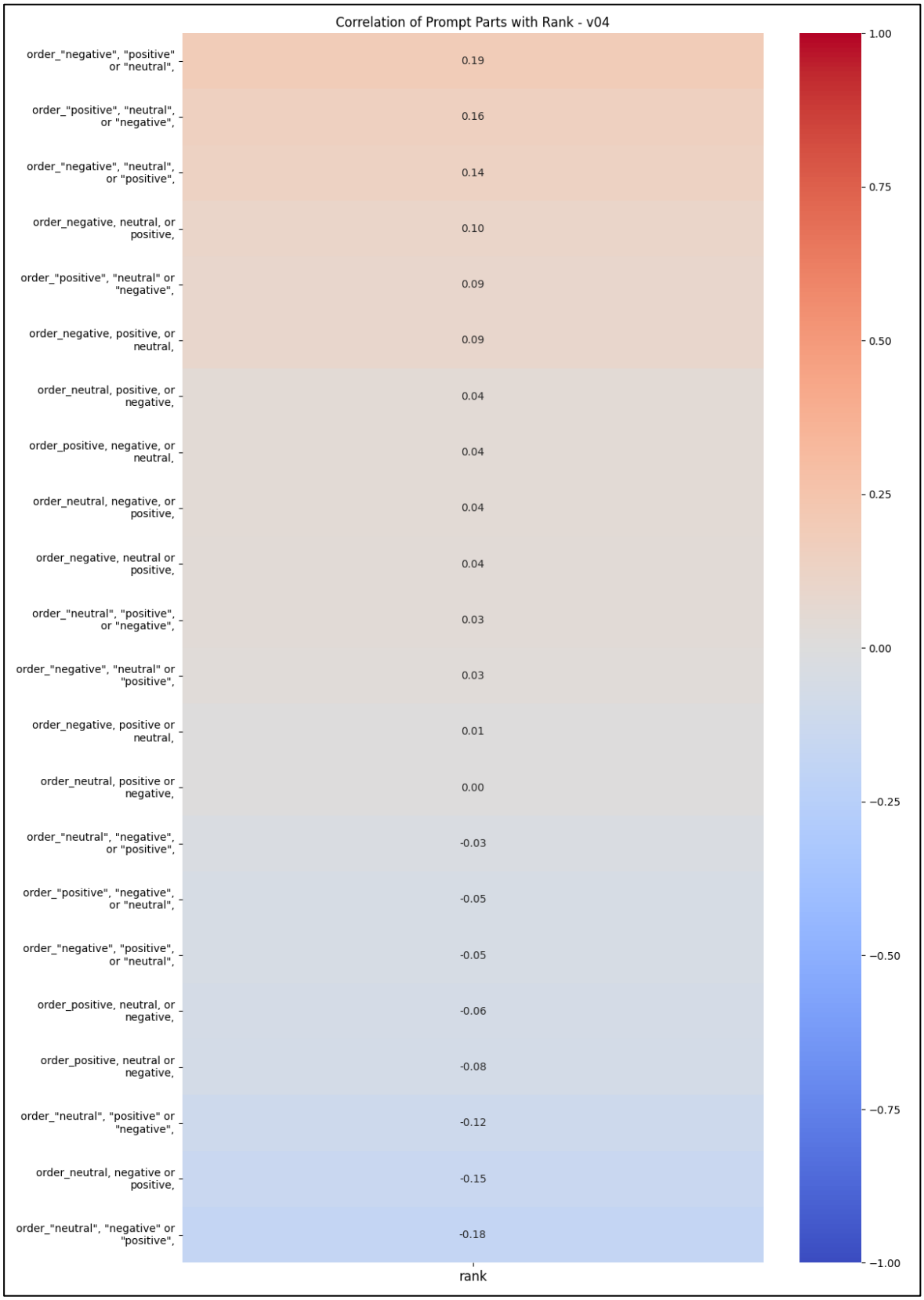
O.5 Sentence_labels



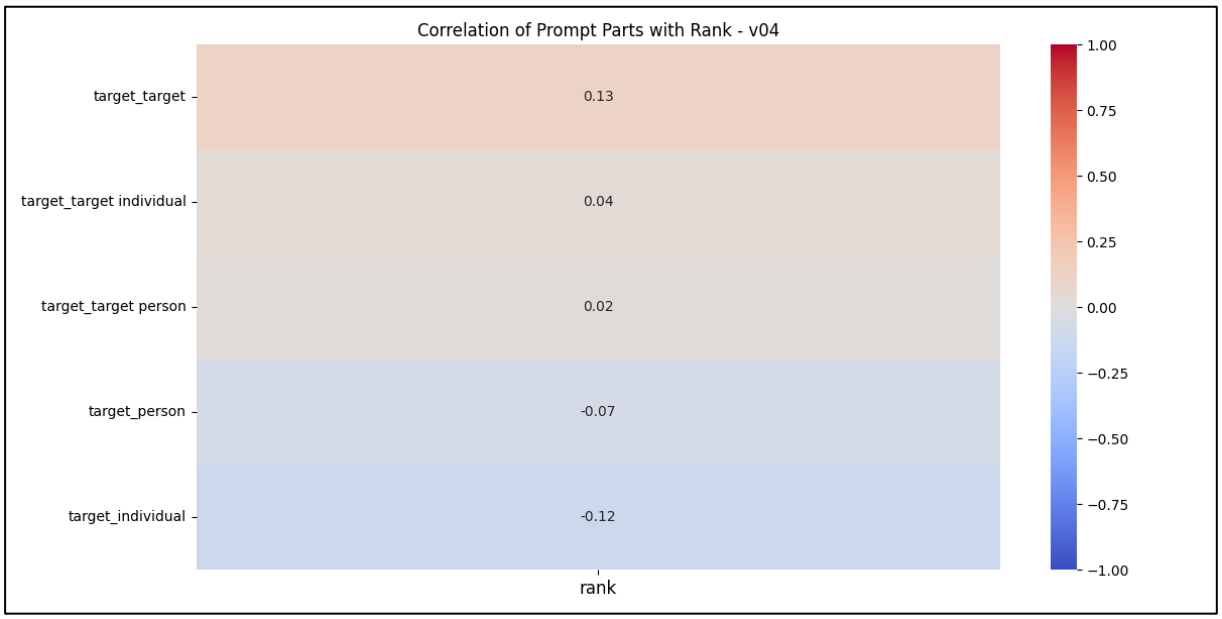
O.6 Sentiment_introductions



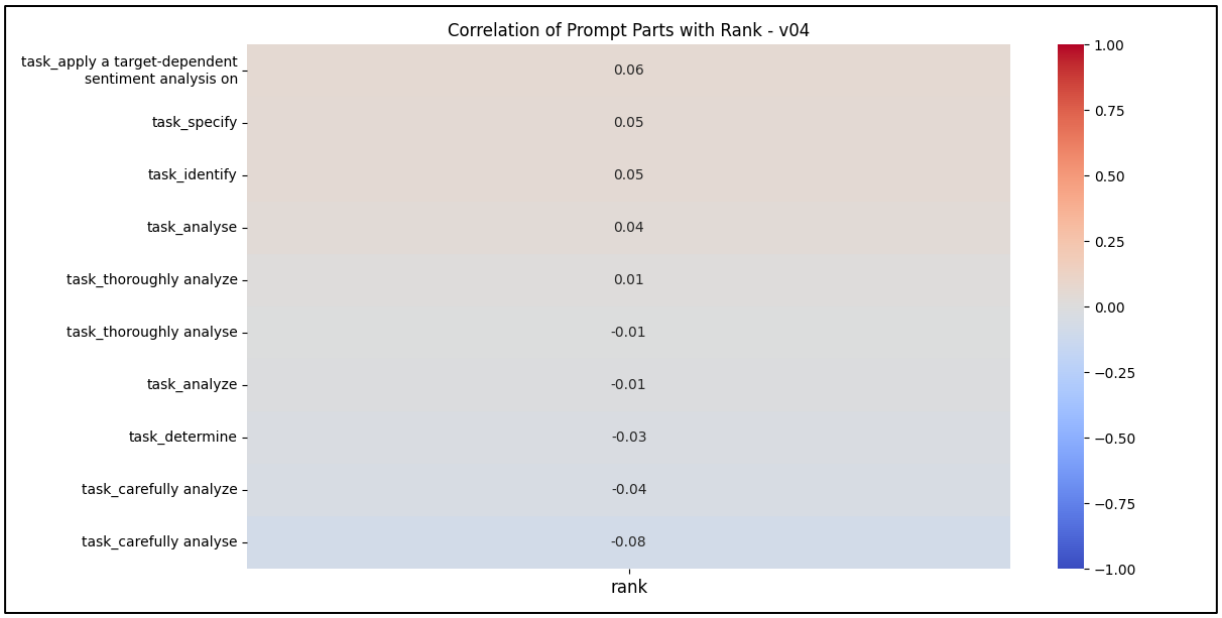
O.7 Sentiment_orders



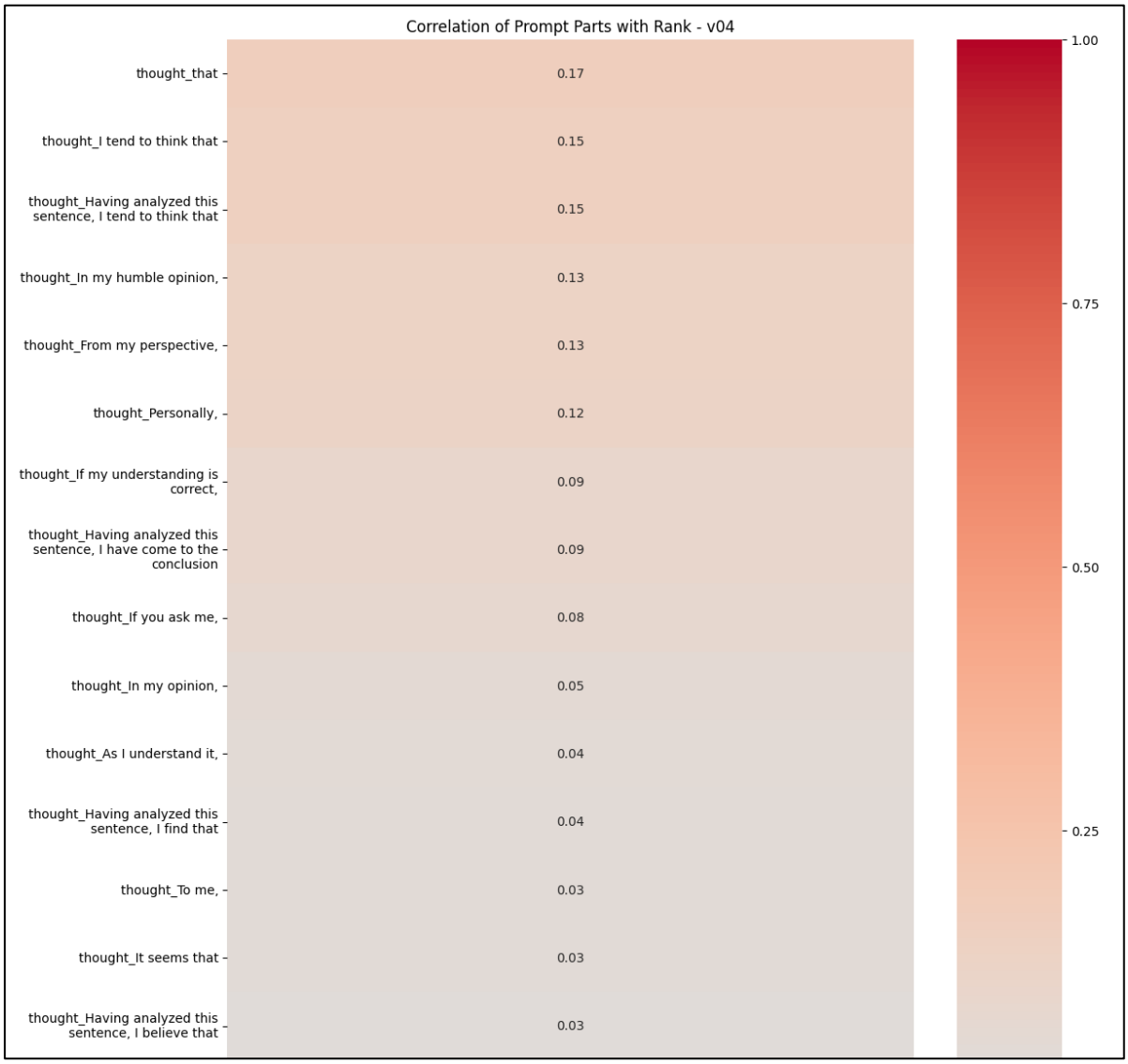
O.8 Targets

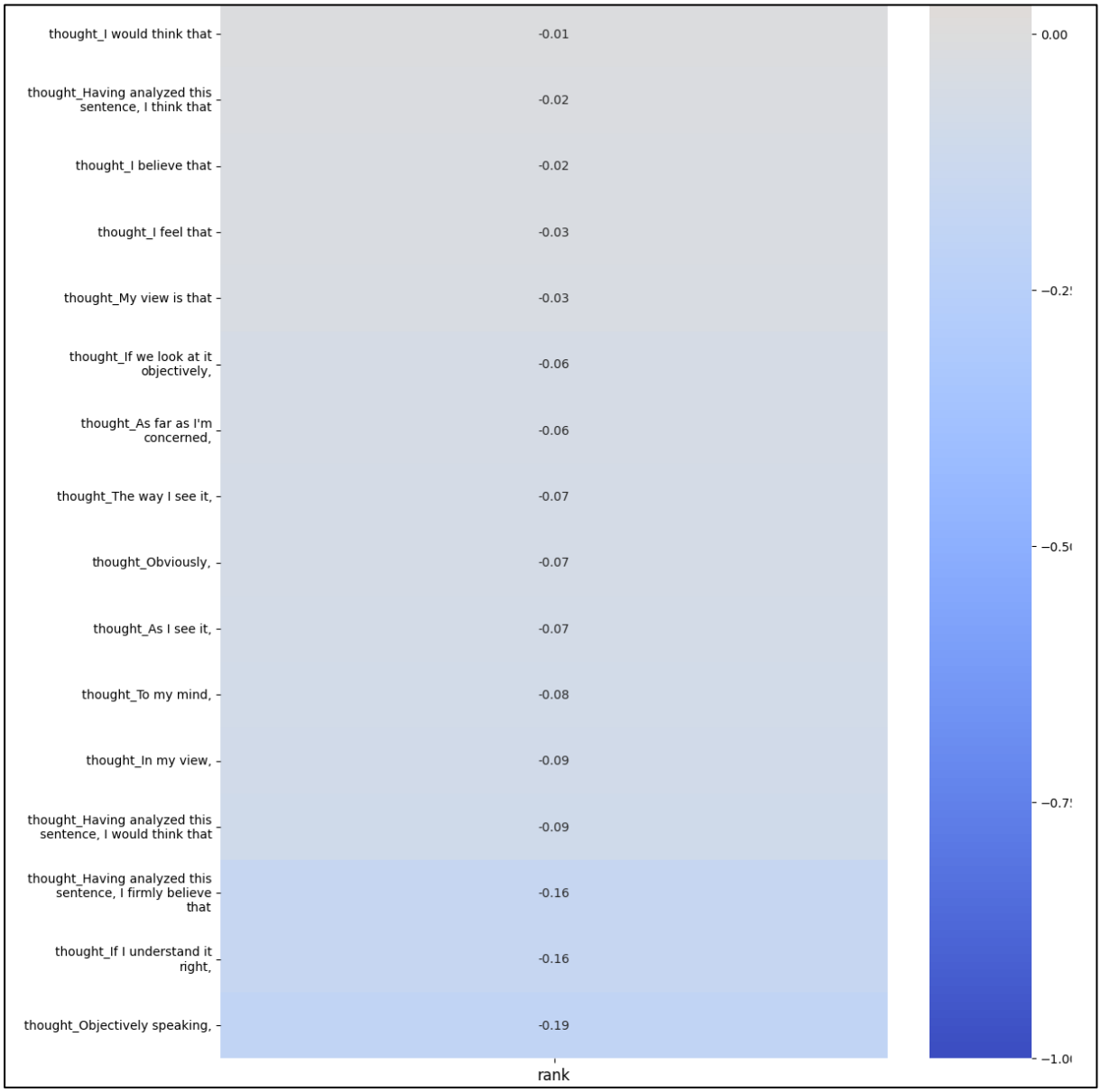


O.9 Tasks

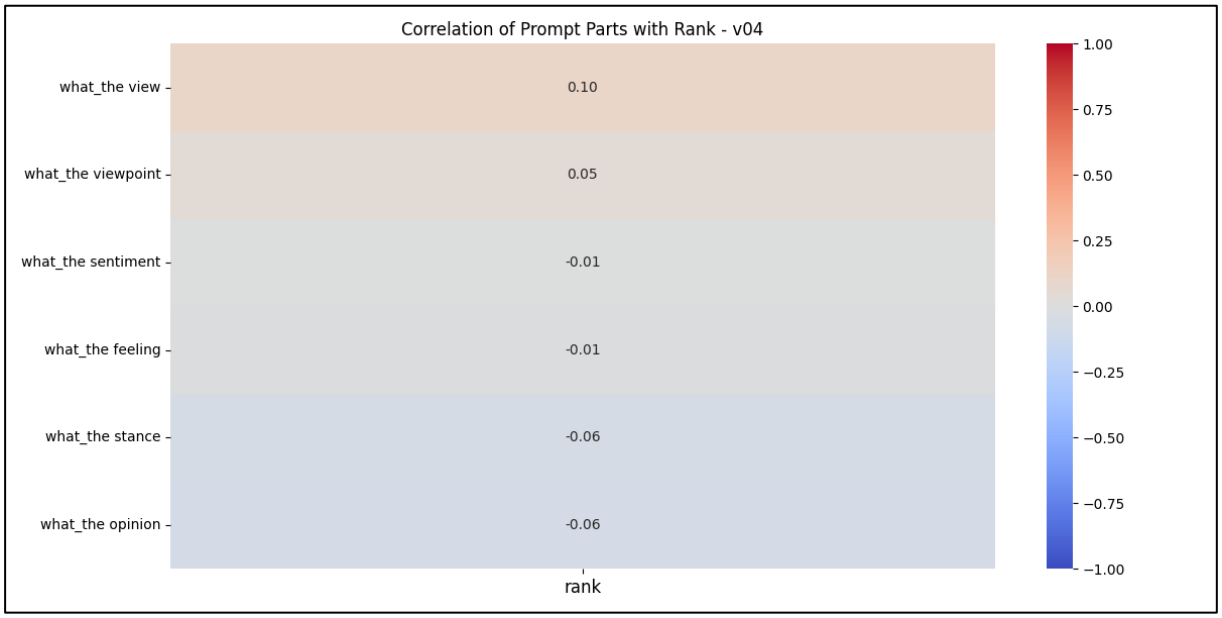


O.10 Thoughts




thought_I would think that
thought_Having analyzed this sentence, I think that
thought_I believe that
thought_I feel that
thought_My view is that
thought_If we look at it objectively,
thought_As far as I'm concerned,
thought_The way I see it,
thought_Obviously,
thought_As I see it,
thought_To my mind,
thought_In my view,
thought_Having analyzed this sentence, I would think that
thought_Having analyzed this sentence, I firmly believe that
thought_If I understand it right,
thought_Objectively speaking,
-0.01
-0.02
-0.02
-0.03
-0.03
-0.06
-0.06
-0.07
-0.07
-0.07
-0.08
-0.09
-0.09
-0.16
-0.16
-0.19
rank
0.00
−0.2
−0.5
−0.7
−1.0

O.11 Whats



O.12 Wheres