Ludwig-Maximilians-Universität München
Department of Statistics

Master's Thesis

Re-Evaluating GermEval 2017: Document-Level and Aspect-Based Sentiment Analysis Using Pre-Trained Language Models

Author:
Alessandra Corvonato

Supervisors:
Prof. Dr. Christian Heumann
M.Sc. Matthias Assenmacher

January 4, 2021

Abstract

The GermEval Shared Task on "Aspect-based Sentiment in Social Media Customer Feedback" was conducted in 2017 using mainly traditional machine learning methods or LSTM-based neural networks. In the scope of the present master's thesis, the GermEval 2017 Task was re-evaluated using state-of-the-art methods. The task represents a complete classification pipeline consisting of four subtasks: Relevance Classification (Subtask A), Document-level Polarity (Subtask B), Aspect-based Polarity (Subtask C) and Opinion Target Expression identification (Subtask D). The whole task was tackled using seven different pre-trained Transformer-based language models, consisting of five BERT and two DistilBERT models which are suitable for the German language. The language models outperformed all the micro-averaged F1 scores achieved in 2017 on a synchronic and a diachronic test dataset by a large margin. Moreover, they achieved similar results for both test datasets, which shows robustness. The model which performed best on all subtasks is the uncased German $\text{BERT}_{\text{BASE}}$ model provided by DBMDZ, advancing the state-of-the-art for this task. For Subtask D, the best scores were achieved by that model using a CRF layer instead of a softmax layer.

Contents

List of Figures

List of Tables

List of Acronyms

`[CLS]` classification token.

`[SEP]` separation token.

ABSA Aspect-Based Sentiment Analysis.

ALBERT A Lite BERT.

BERT Bidirectional Encoder Representations from Transformers.

biLM bidirectional language model.

biLSTM bidirectional Long Short-term Memory.

BIO Beginning, Inner, Outer (annotation scheme).

BOW Bag of Words.

BPE byte-pair encoding.

CBOW Continuous Bag of Words.

CNN Convolutional Neural Network.

CRF Conditional Random Field.

DB "Deutsche Bahn".

DBMDZ "Digitale Bibliothek/Münchener DigitalisierungsZentrum".

DistilBERT distilled version of BERT.

ELMo Embeddings from Language Models.

GLUE General Language Understanding Evaluation.

GPT Generative Pre-trained Transformer.

LSTM Long Short-term Memory.

MLM masked language model.

NER Named Entity Recognition.

NLP Natural Language Processing.

NSP next sentence prediction.

OTE Opinion Target Expression.

PE positional encoding.

POS Part-of-Speech.

RNN Recurrent Neural Network.

RoBERTa Robustly optimized BERT approach.

SOP sentence-order prediction.

SQuAD Stanford Question Answering Dataset.

SRL Semantic Role Labeling.

SST Stanford Sentiment Treebank.

SVM Support Vector Machine.

TF-IDF Term Frequency - Inverse Document Frequency.

Chapter 1

Introduction

Customer feedback can be analyzed to attain an important source for identifying problems affecting the services and the image of a company. To transform gathered data into helpful information on how a product or service is perceived among the customers, sentiment analysis is needed. Although crawled data, e.g. from social media platforms like Twitter and Facebook, is often noisy and does not always cover specific company related topics, it is much cheaper than conducting large studies using questionnaires. However, scraped data from the web must be classified in relevant and irrelevant documents before feedback is extracted automatically. Preferably, the approach should work on multilingual texts as social media feedback provides many non-English documents.

Up to now, only a few works have focused on sentiment related problems for German texts. One of them is the GermEval 2017 Shared Task on "Aspect-based Sentiment in Social Media Customer Feedback" (Wojatzki et al., 2017). The data was crawled from the web and analyzed by several teams using standard machine learning techniques such as Support Vector Machines (SVM) or models based on neural networks like Long Short-term Memory (LSTM). However, there has been major development in the field of NLP in the past few years. In 2017, researchers demonstrated the power of transfer learning for natural language modeling that means pre-training a neural network model on a known task, and then performing fine-tuning, using the trained neural network as the basis of a new purpose-specific model. Consequently, LSTM-based language models were practically replaced with language models relying on the Transformer, the transfer learning technique developed by Vaswani et al. (2017). Afterwards, Google developed a Transformer-based language model called Bidirectional Encoder Representations from Transformers, short BERT (Devlin et al., 2019), achieving state-of-the-art performance on several natural language tasks and becoming a cornerstone in the field of NLP. Today,

pre-trained language models are available for many languages, including German, so that the GermEval 2017 Task can be solved using the new state-of-the-art. Guhr et al. (2020) and Biesialska et al. (2020), for instance, have already re-evaluated some part of the task and have shown that a Transformer-based model outperforms the results for the classification of document-level polarity.

This thesis aims at re-analyzing the complete GermEval 2017 Shared Task using a variety of pre-trained language models which are currently available for German. The language models are expected to outperform the systems trained at that time. A comparison between the pre-trained models will be drawn in order to discuss which model is best suited for this task.

The thesis is outlined as follows. In Chapter 2 an overview on the GermEval 2017 is given. More precisely, the task, the data, the participants' methods and their results are depicted here. In Chapter 3, the theoretical background is explained in order to understand the methods used for re-evaluating the GermEval 2017 Task. In Chapter 4, the results of the re-evaluation are shown in detail and compared to those achieved in 2017. Afterwards, the results are discussed and an outlook to possible future work is given in Chapter 5. The thesis concludes with a summary, recapping the main findings.

Chapter 2

GermEval 2017

Sentiment analysis can be conducted on different levels, depending on what the business is interested in, e.g. document- or aspect-level sentiment. The GermEval 2017 Shared Task by Wojatzki et al. (2017) is a task on automatically analyzing customer reviews about "Deutsche Bahn" (DB) - the German public train operator. The idea is to derive insights from reviews on different granularities, e.g. general evaluation of travel or evaluation of a dedicated aspect of train travel. Therefore, the task is formed of four subtasks, namely

- Relevance Classification,
- Document-level Polarity,
- Aspect-based Polarity,
- Opinion Target Expression (OTE).

The organizers are the first to provide a large public annotated dataset for German sentiment analysis in order to hopefully strengthen the research on German sentiment and social media analysis. Eight teams participated in the shared task using different approaches and achieving different results. The organizers provided a dictionary to the participants, as well as a majority class baseline and a baseline system for comparison. The latter consists of a linear SVM classifier for the first three subtasks and a Conditional Random Field (CRF) classifier for the last subtask.

In Section 2.1 the four subtasks are described in more detail. In Section 2.2 a descriptive analysis of the data is given. The same data will also be used to carry out the re-evaluation outlined in Chapter 4. Afterwards, an overview of the participants' approaches are given in Section 2.3 and in Section 2.4, the results of their evaluation are presented.

2.1 Task Description

The shared task comprises four subtasks that can be approached individually. They aim at implementing a complete classification pipeline when handling web data from different heterogeneous sources. The subtasks are explained separately in the following.

2.1.1 Subtask A: Relevance Classification

Subtask A focuses on determining whether a document is relevant to the topic. In real life scenarios, this task is necessary to filter irrelevant documents that are a by-catch of the data collection.

In German, the term "Bahn" can refer to the trains, the rails, any track or anything that can be straight in lines. It often occurs in compound words such as "U-Bahn" (subway, short for "Untergrundbahn"), "Seilbahn" (cableway) or "Achterbahn" (roller coaster). Thus, it is crucial to first distinguish between documents that refer to the DB and those that do not and to eliminate the latter ones for the next steps. Below is an original example for a relevant and an irrelevant document (Table 2.1).

Relevance	**Example (German)**	**Example (English)**
`true`	Und ich hasse es das die bahn immer jede 10 min erst kommt	And I hate it that the train always comes only every 10 minutes
`false`	Verärgerung bei Pendlern in London: Tiefstehende Sonne sorgt bei Bahn für Verspätungen [...]	Annoyance with commuters in London: Low sun causes train delays [...]

Table 2.1: Example for document relevance.

The relevant feedback is about the train running infrequently and the irrelevant document refers to delays of trains in London.

2.1.2 Subtask B: Document-Level Polarity

In general, Subtask B corresponds to a document-level sentiment analysis. Hereby, the systems should identify if the customer evaluates the service of the company as positive, negative or neutral. Document-level sentiment is determined by the polarities of the individual aspects in the document. If there is a mix between neutral and positive or

negative sentiment, the document is classified as `positive` or `negative`, respectively. If there are two opposite polarities (positive and negative), the document-level polarity is set to `neutral`. Table 2.2 gives an example for a positive, a neutral and a negative document polarity.

Sentiment	Example (German)	Example (English)
`negative`	Hat die Bahn eigentlich jeden Tag mindestens 20 Minuten verspätung oder nur wenn ich fahre?	Is the train actually at least 20 minutes late every day or only when I am taking it?
`neutral`	Huhu, liebe @DB_Bahn Wo kann ich denn meine Erstattungsansprüche wegen der Ereignisse heute morgen geltend machen?	Yoo-hoo, dear @DB_Bahn Where can I claim my reimbursement for the events this morning?
`positive`	Sowohl auf der Hin- als auch auf der Rückreise lief alles perfekt, danke @DB_Bahn [...]	Everything went perfectly on the outward and return journey, thanks @DB_Bahn [...]

Table 2.2: Example for document sentiment.

The first document is annotated as `negative` since it complains about the DB always being late, whereas the second one is a question regarding refunds and therefore annotated as `neutral`. The last document expresses a positive journey with DB.

2.1.3 Subtask C: Aspect-Level Polarity

The Aspect-Based Sentiment Analysis (ABSA) is conducted with Subtask C. Its target is to establish a system that identifies all the aspects with corresponding polarity in a document. The aspects of the provided data were assigned to categories (e.g. Allgemein (engl. general), Ticketkauf (engl. ticket purchase), Connectivity etc.) and subcategories (e.g. Connectivity#WLAN/Internet). Multiple annotations of the same category are possible. Table 2.3 gives an example for two different aspects in one document.

Aspect+Sentiment	**Example (German)**	**Example (English)**
`Atmosphäre:negative`	Wieder sind in der Hitze Klimaanlagen in Zügen ausgefallen.	Once again, air conditioning systems in trains broke down in the heat.
`Allgemein:positive`	Diesmal reagiert die Deutsche Bahn schnell und twitter...	This time, the Deutsche Bahn reacts quickly and twitters...

Table 2.3: Example for a document with different aspects and polarities.

This particular writer complains about the broken air conditioning system, and thus, the aspect `Atmosphäre` (engl. atmosphere) is assigned with negative polarity. However, the writer continues that the DB reacted quickly at that time. Therefore, the document is also labeled with the aspect `Allgemein` and positive polarity.

2.1.4 Subtask D: Opinion Target Expression

For Subtask D, the participants' systems should identify the linguistic phrases in the document which are used to express the aspect-based polarities from Subtask C. This task is called Opinion Target Expression (OTE) identification. An example for a ticket purchase containing a neutral sentiment is given below in XML format.

```
</Document>
<Document id=[...]>
    <Opinions>
        <Opinion category="Ticketkauf#Haupt" from="50" to="63"
        target="Stornierungen" polarity="neutral"/>
    </Opinions>
    <relevance>true</relevance>
    <sentiment>neutral</sentiment>
    <text>Re: DB Bahn Wie sollen sie denn bei dem Haufen an
    Stornierungen SOFORT alle AUF EINMAL erstatten?</text>
</Document>
```

Translation: "DB Bahn How are they supposed to reimburse all the cancellations IMMEDIATELY AT ONCE?", category="ticket purchase#main", target="cancellations".

The target phrase in this example document would be "Stornierungen" (engl. cancellations) which is identified by the string positions within `from` and `to`. The system should

then label the expression accordingly. The label would be the aspect category `Ticketkauf` with the polarity `neutral`. Several targets with different aspects and polarities in one document are possible.

2.2 Data Description

The main data, including the document texts and their URL, was collected by web scraping on a daily basis for a whole year between May 2015 and June 2016. The basic sources were social media platforms like Twitter, Facebook, microblogs, news, and question and answer websites. From each month, approximately 1,500 documents were sampled and manually annotated by two annotators. In case of inconsistencies, a supervisor decided on the correct ones. Relevant documents without clear OTE could also be assigned a document-level aspect annotation. In order to obtain primarily relevant documents, the organizers trained a baseline SVM classifier in advance to perform pre-filtering. Moreover, a second dataset was crawled from November 2016 to January 2017 to test the robustness of the models. This diachronic dataset was pre-processed the same way as the main data. The annotated data was then used by the participants for training, evaluating and testing.

The data is freely available in XML and TSV format[1]. Each data split in TSV format contains the following variables:

- document id (URL)
- document text
- relevance label (`true`/`false`)
- document-level sentiment label (`negative`/`neutral`/`positive`)
- aspects with polarities (e.g. `Ticketkauf#Haupt:negative`)

For documents which are annotated as irrelevant, the sentiment label is set to `neutral` and no aspects are available. Visibly, the TSV formatted data does not contain the target expressions or their associated sequence positions. Consequently, Subtask D can only be conducted using the data in XML format which incorporates the same information as the data in TSV format together with the starting and ending sequence positions of the target phrases. An example document in XML format was given in Section 2.1.4.

[1]The data can be downloaded from http://ltdata1.informatik.uni-hamburg.de/germeval2017/.

The data contains about 26,000 records in total, including the diachronic test dataset with around 1,800 examples. Further, the main data was randomly split by the organizers into a train dataset for training, a development dataset for evaluation and a synchronic test dataset. Table 2.4 displays the number of documents for each split.

train	**dev**	**test syn**	**test dia**
19,432	2,369	2,566	1,842

Table 2.4: Number of documents per dataset. "test syn" stands for the synchronic test dataset and "test dia" for the diachronic test dataset.

Obviously, most of the data is used for training the system, roughly 74%, whereas the development split contains around 9% and the synchronic test split around 10% of the data. The remaining 7% of the data records belong to the diachronic data.

Relevance	**train**	**dev**	**test syn**	**test dia**
true	16,201	1,931	2,095	1,547
false	3,231	438	471	295

Table 2.5: Relevance distribution per dataset for Subtask A.

Table 2.5 shows the relevance distribution for every data split. Hereby, the relevant documents represent the clear majority with over 80% in each dataset.

Sentiment	**train**	**dev**	**test syn**	**test dia**
negative	5,045	589	780	497
neutral	13,208	1,632	1,681	1,237
positive	1,179	148	105	108

Table 2.6: Sentiment distribution per dataset for Subtask B.

Table 2.6 gives the sentiment distribution per dataset. The neutral class appears as the clear majority class with 65-69% of the data in each split. The documents with a negative sentiment form at least 25-31% of the data and the positive class contains only about 4-6% of the documents.

Table 2.7 presents the distribution of the aspect categories. The organizers assigned the aspects to 20 different categories, where multiple annotations are possible[2]. The

[2]A detailed category description can be found here: https://sites.google.com/view/germeval2017-absa/data.

table shows the number of documents containing certain categories without differentiating between how often a given category can be found within the document.

Category	**train**	**dev**	**test syn**	**test dia**
Allgemein	11,454	1,391	1,398	1,024
Zugfahrt	1,687	177	241	184
Sonstige Unregelmäßigkeiten	1,277	139	224	164
Atmosphäre	990	128	148	53
Ticketkauf	540	64	95	48
Service und Kundenbetreuung	447	42	63	27
Sicherheit	405	59	84	42
Informationen	306	28	58	35
Connectivity	250	22	36	73
Auslastung und Platzangebot	231	25	35	20
DB App und Website	175	20	28	18
Komfort und Ausstattung	125	18	24	11
Barrierefreiheit	53	14	9	2
Image	42	6	0	3
Toiletten	41	5	7	4
Gastronomisches Angebot	38	2	3	3
Reisen mit Kindern	35	3	7	2
Design	29	3	4	2
Gepäck	12	2	2	6
QR-Code	0	1	1	0
total	18,137	2,149	2,467	1,721
# documents with aspects	16,200	1,930	2,095	1,547
∅ different aspects/document	1.12	1.11	1.18	1.11

Table 2.7: Aspect category distribution per dataset for Subtask C. Multiple mentions of the same aspect category in a document are not considered.

The relative distribution of the aspect categories is similar between the datasets. On average, the data shows 1.12 different aspects per document. The category `Allgemein` clearly represents the majority class as it is present in 75.8% of the documents with aspects. The second most frequent category is `Zugfahrt` (engl. train ride) appearing in only around 13.8% of the documents. This strong imbalance in the aspect categories

leads to an almost Zipfian distribution[3] (Wojatzki et al., 2017).

Apparently, there exists a clear majority class for every task, leading to a strong baseline and unbalanced data. Two of the participant teams tried to compensate the imbalances in the data. The participants' approaches are summarized in the following section.

2.3 Participants' Approaches

The participating teams applied various approaches across all subtasks. However, some trends and commonalities are identifiable (Wojatzki et al., 2017). For tokenizing, some participants used off-the-shelf tokenizers such as the opennlp maxent tokenizer (Schulz et al., 2017), while others made use of own implementations combined with large sets of rules covering social media specific language phenomena such as emoticons, URLs or repeated punctuation (Sayyed et al., 2017; Sidarenka, 2017; Mishra et al., 2017; Hövelmann and Friedrich, 2017). Hövelmann and Friedrich (2017) also normalized the data by lower-casing the letters and applying an off-the-shelf spell checker and rules to replace dates, numbers, and URLs with a special token. Some teams pre-processed the data utilizing lemmatizers[4], chunkers[5] and Part-of-Speech (POS) taggers[6] (Sidarenka, 2017; Schulz et al., 2017; Naderalvojoud et al., 2017). Sayyed et al. (2017) and UH-HHU-G[7] considered sampling techniques to compensate for imbalances in the data.

Most participants relied on sentiment lexicons like the one published by Waltinger (2010) (Schulz et al., 2017; Mishra et al., 2017) or SentiWS by Remus et al. (2010) (Schulz et al., 2017; Sidarenka, 2017). Some teams also created their own lexicons such as Sidarenka (2017) and Naderalvojoud et al. (2017).

Furthermore, some teams considered word embeddings in order to incorporate distributional semantic word information in their models. For instance, Lee et al. (2017) used Word2Vec (Mikolov et al., 2013a) trained word embeddings. They also trained

[3] a pattern of distribution, by which the frequency of an item is inversely proportional to its ranking by frequency. In such a distribution, frequency declines sharply as rank number increases: A small number of items occur very frequently, and a large number of items occur very rarely.

[4] do full morphological analyses to accurately identify the lemma for each word

[5] analyze a sentence to identify the constituents (noun groups, verbs, etc.)

[6] assign parts of speech to each word, e.g. noun, verb, adjective, etc., or more fine-grained tags such as "noun-singular"

[7] submission withdrawn after reviewing

sentence vectors on the same data and experimented with German-English bilingual embeddings. Mishra et al. (2017) trained dense word vectors on a large corpus of parliament speeches using GloVe (Pennington et al., 2014). Some of the teams such as Hövelmann and Friedrich (2017) relied on fastText (Bojanowski et al., 2017; Joulin et al., 2017) which considers sub-word information to create word embeddings.

The teams used different models to classify the data. In total, three major trends can be observed: traditional (non-neural) classification approaches, neural networks, and ensemble methods that combine neural and non-neural approaches. SVMs, CRFs and threshold based classification strategies belong to the non-neural approaches considered by some teams (Sidarenka, 2017; Mishra et al., 2017; Lee et al., 2017; Schulz et al., 2017). Concerning the neural approaches, one can observe the use of Recurrent Neural Networks (RNN) such as (bidirectional) LSTMs (UH-HHU-G; Sidarenka, 2017; Mishra et al., 2017; Naderalvojoud et al., 2017; Lee et al., 2017). Others considered convolutional layers or multi-layered perceptrons (UH-HHU-G; Mishra et al., 2017). Within the ensemble methods, there are ensembles combining LSTM and SVM (Sidarenka, 2017), fastText (Hövelmann and Friedrich, 2017) and approaches that rely on gradient boosted trees (Hövelmann and Friedrich, 2017; Sayyed et al., 2017) and several neural networks (Lee et al., 2017).

In addition, the organizers shared another self-built framework. Ruppert et al. (2017) implemented SVMs for the document-level classification tasks (Subtask A, B and C). The model features are based on TF-IDF weights for the top 30 tokens, word embeddings obtained from Word2Vec, a background lexicon, an aggregated lexicon and a sentiment lexicon. The latter was enlarged with the polarity lexicon published by Waltinger (2010). Moreover, a CRF classifier was considered for Subtask D. Here, the relative-positional features include the current token, the POS tag, the lemma, the character pre- and suffixes, capitalization, hyphenation, the numeric type (identifies the type when numbers are present e.g. year, digits) and character categories (patterns based on Unicode categories).

The following section presents the participants' results for GermEval 2017. Although the system provided by Ruppert et al. (2017) did not officially compete in the shared task, for the sake of completeness, their results are included in the following. Some of the above-mentioned methods are described in more detail within Section 3.

2.4 Participants' Results

The participants provided their results on the synchronic and diachronic test datasets to the organizers. To adequately compare the results, the micro-averaged F1 score was considered as the evaluation measure in every task. This metric is well-suited for datasets with a clear majority class since each observation is equally weighted, which is especially beneficial for evaluating the multilabel classification (Subtask C). The micro-averaged F1 score is calculated as the harmonic mean of the micro precision and the micro recall:

$$\text{micro F1 score} = 2 \times \frac{\text{micro precision} \times \text{micro recall}}{\text{micro precision} + \text{micro recall}} \in [0, 1],$$

where a score of 1 indicates a perfect classification. The micro precision and recall are defined as

$$\text{micro precision} = \frac{\sum \text{true positives}}{\sum \text{true positives} + \sum \text{false positives}},$$
$$\text{micro recall} = \frac{\sum \text{true positives}}{\sum \text{true positives} + \sum \text{false negatives}}.$$

Table 2.8 presents the results for Subtask A for the two test datasets. Besides the organizers, six teams contributed to this subtask. The best scores are printed in bold.

Team	Method	syn	dia
Sayyed et al. (2017)	xgboost	**0.903**	**0.906**
Hövelmann and Friedrich (2017)	fastText	0.899	0.897
organizers (Ruppert et al., 2017)	SVM	0.895	0.894
Mishra et al. (2017)	biLSTM structured perceptron	0.879	0.870
Lee et al. (2017)	stacked learner CCA SIF embedding	0.873	0.881
Sidarenka (2017)	biLSTM-SVM	0.873	0.869
Sidarenka (2017)	biLSTM	0.865	0.857
Hövelmann and Friedrich (2017)	GBT_BOW	0.863	0.856
organizers	baseline system	0.852	0.868
UH-HHU-G	ridge classifier char fourgram	0.835	0.849
UH-HHU-G	linear SVC 12 char fivegram	0.834	0.859
UH-HHU-G	passive-aggressive char fivegram	0.827	0.850
UH-HHU-G	linear SVC 12 trigram	0.824	0.837
organizers	majority class baseline	0.816	0.839
UH-HHU-G	gru mt	0.816	0.840
UH-HHU-G	cnn gru sent mt	0.810	0.839
Hövelmann and Friedrich (2017)	ensemble	0.734	0.160

Table 2.8: Results for Subtask A per participating team on synchronic and diachronic test dataset (Wojatzki et al., 2017).

With a micro F1 score of 0.903 for the synchronic test data and 0.906 for the diachronic data, Sayyed et al. (2017) provided the model with the best scores. The remaining participants achieved scores below 0.9. Overall, the results across the teams are very similar and most of them obtain slightly better results for the diachronic test dataset than for the synchronic one.

All the teams participated in Subtask B. Table 2.9 shows their results.

Team	**Method**	**syn**	**dia**
organizers (Ruppert et al., 2017)	SVM	**0.767**	0.744
Naderalvojoud et al. (2017)	SWN2-RNN	0.749	0.736
Hövelmann and Friedrich (2017)	fastText	0.748	0.742
Sidarenka (2017)	biLSTM-SVM	0.745	0.718
Naderalvojoud et al. (2017)	SWN1-RNN	0.737	0.736
Sayyed et al. (2017)	xgboost	0.733	**0.750**
Sidarenka (2017)	biLSTM	0.727	0.704
Lee et al. (2017)	stacked learner CCA SIF embedding	0.722	0.724
Hövelmann and Friedrich (2017)	gbt_bow	0.714	0.714
Hövelmann and Friedrich (2017)	ensemble	0.710	0.725
UH-HHU-G	ridge classifier char fourgram	0.692	0.691
Mishra et al. (2017)	biLSTM structured perceptron	0.685	0.675
UH-HHU-G	linear SVC 12 char fivegram	0.680	0.692
organizers	baseline system	0.667	0.694
UH-HHU-G	linear SVC 12 trigram	0.663	0.702
organizers	majority class baseline	0.656	0.672
UH-HHU-G	gru mt	0.656	0.672
UH-HHU-G	cnn gru sent mt	0.644	0.668
Schulz et al. (2017)		0.612	0.616
UH-HHU-G	passive-aggressive char fivegram	0.575	0.676

Table 2.9: Results for Subtask B per participating team on synchronic and diachronic test dataset (Wojatzki et al., 2017).

For the synchronic test dataset, Ruppert et al. (2017) reached the best result, reaching a micro F1 score of 0.767. The highest score for the diachronic test set was 0.750, which was obtained by Sayyed et al. (2017) using the xgboost model. Interestingly, for the synchronic dataset, the xgboost model only ranks sixth. Most of the systems achieved better scores on the diachronic data than on the synchronic data, especially the weaker systems. However, the five best models show higher scores for the synchronic test dataset.

Apart from the organizers, only Lee et al. (2017) and Mishra et al. (2017) participated in Subtask C. The results are displayed in Table 2.10.

Team	Method	Test syn		Test dia	
		aspect	aspect+ sentiment	aspect	aspect+ sentiment
organizers (Ruppert et al., 2017)	SVM	**0.537**	**0.396**	**0.556**	**0.424**
Lee et al. (2017)	LSTM CRF stacked learner correct offsets	0.482	0.354	-	-
organizers	baseline system	0.481	0.322	0.495	0.389
organizers	majority class baseline	0.442	0.315	0.456	0.384
Mishra et al. (2017)	biLSTM structured perceptron	0.421	0.349	0.460	0.401
Lee et al. (2017)	LSTM CRF stacked learner correct offsets 2	0.358	0.308	-	-
Lee et al. (2017)	LSTM CRF only correct offsets	0.095	0.081	-	-

Table 2.10: Results for Subtask C per participating team on synchronic and diachronic test dataset (Wojatzki et al., 2017).

Ruppert et al. (2017) provided the best results for the complete Subtask C. Regarding the aspect classification, their system respectively reached a score of 0.537 and 0.556 for the synchronic and diachronic test data. Concerning the aspect+sentiment classification, they obtained 0.396 and 0.424. Here, the diachronic test set shows overall better results than the synchronic test data. However, one team (Lee et al., 2017) did not provide any results for the diachronic data.

The same teams that participated in Subtask C also tackled Subtask D. Their results are demonstrated in Table 2.11.

Team	Method	Test syn		Test dia	
		exact	overlap	exact	overlap
organizers (Ruppert et al., 2017)	CRF	**0.229**	0.306	**0.301**	**0.365**
Mishra et al. (2017)	biLSTM structured perceptron	0.220	0.221	0.281	0.282
Lee et al. (2017)	LSTM CRF stacked learner correct offsets	0.203	**0.348**	-	-
Lee et al. (2017)	LSTM CRF stacked learner correct offsets 2	0.186	0.267	-	-
organizers	baseline system	0.170	0.237	0.216	0.271
Lee et al. (2017)	LSTM CRF only correct offsets	0.089	0.089	-	-
Lee et al. (2017)	LSTM CRF stacked learner for polarity correct offsets	0.024	0.183	-	-

Table 2.11: Results for Subtask D per participating team on synchronic and diachronic test dataset (Wojatzki et al., 2017)

Concerning the exact target match, Ruppert et al. (2017) provided the best results for both test datasets again. These are 0.229 for the synchronic and 0.301 for the diachronic test data. Regarding the overlap match, the organizers reached the best results with a value of 0.365 for the diachronic test set, while for the synchronic test data, Lee et al. (2017) achieved a score of 0.348, outperforming the others.

All in all, despite the use of traditional classifiers like SVM and CRF, the organizers mostly achieved the best results. Sidarenka (2017) also showed that traditional machine learning methods are able to outperform deep learning methods for these tasks. This confirms that the neural-based NLP approaches used in 2017 still leave room for improvement. A notable downside of these methods was the lack of contextualized word embeddings. Every word was assigned exactly one word embedding albeit most of the words have multiple meanings depending on the context. Meanwhile, the NLP world was expanded by several methods establishing new state-of-the-art results on many language modeling tasks. In Section 3, the recent evolution in the field of NLP are described.

Chapter 3

Methods

Nowadays, German text data can be analyzed using state-of-the-art Transformer-based language models such as BERT. In order to understand the workings of BERT, the underlying principles need to be explored in more depth. In this chapter, the theoretical background of more generic NLP research is described before taking a detailed look at BERT.

First, static text and word representations are sketched (Section 3.1 and 3.2). Next, the theoretical background of ELMo (Peters et al., 2018) is explained in detail as it was the first LSTM-based approach to successfully compute contextualized word embeddings (Section 3.3). Section 3.4 illustrates the Transformer framework published by Vaswani et al. (2017), a language modeling architecture tackling the downsides of LSTM-based models. Finally, Section 3.5 refers to the Transformer-based model BERT, which was initially proposed by Devlin et al. (2019).

3.1 Text Representations

Machine learning models cannot work with raw text directly, the text must first be converted into numbers or numerical vectors reflecting linguistic properties. Directly extractable features are, for instance, the counts and the order of the letters or the words within the text. These count-based text representations initialize the word vectors based on the frequency of word appearances in the document. The following sketches two common techniques for modeling text representations: BOW and TF-IDF.

3.1.1 Bag Of Words (BOW)

An intuitive method to represent text is the Bag of Words (BOW) model. The idea is to take each word count as feature, i.e. to look at the histogram of words (Goldberg, 2017). It is called a "bag" of words because any information about the order or grammar of the words in the document is discarded, where a document can be anything from a sentence to a multi-paragraph text. As a variation, one can also consider other counts that directly derive from words and letters, e.g. bag of n-grams[1], bag of letters, or the length of a sentence regarding the number of words or letters (Goldberg, 2017). However, not every word in a document is usually equally important. TF-IDF addresses this problem.

3.1.2 TF-IDF

To enhance the BOW method, one can consider weighting the terms. The aim is to distinguish words which have a high count because they are commonly used (e.g. "the", "and") from words that have a high count because they relate to the topic of the document (Goldberg, 2017). The TF-IDF weighting was introduced by Manning et al. (2008) and is calculated as

$$\frac{\#_d\ (w)}{\sum_{w' \in d} \#_d\ (w')} \times \log\left(\frac{|D|}{|\{d \in D : w \in d\}|}\right). \tag{3.1}$$

Here, d is the document, D represents a larger corpus, and w stands for a word. The term frequency (TF) refers to the first term in Equation 3.1 and represents the normalized count of words in the document d. The inverse document frequency (IDF), corresponding to the second term in Equation 3.1, refers to the inverse of number of distinct documents in the corpus in which this word occurred, i.e. it highlights words distinctive of the current text (Goldberg, 2017).

A shortcoming of these count-based representations is that neither word meanings nor word similarities are considered (Goldberg, 2017). They usually fail to incorporate the context of the words[2]. Moreover, as only the appearance of a word in a document is counted, these one-hot encoded text representations quickly become high-dimensional, sparse feature vectors.

[1] contiguous sequences of n words, e.g. bigrams (pairs of consecutive words).

[2] To a certain extent, this can be circumvented by including bigrams into the word-document matrix. Basically, a BOW with bigrams is more powerful than a single word BOW. However, the use of bigrams also results in many irrelevant entries (Goldberg, 2017).

3.2 Static Word Representations

Neural-based (also called prediction-based) word representations (Baroni et al., 2014; Levy et al., 2015) apply a supervised approach to obtain word vectors. Mikolov et al. (2013a) introduced two approaches for estimating static neural-based word representations: CBOW and continuous skip-gram. Both approaches create word embeddings that represent the textual data. The Word2Vec model by Google (Mikolov et al., 2013a) uses either of the two model architectures to produce a distributed representation of words, i.e. words which appear in the same context tend to have similar meanings (Harris, 1954). According to Mikolov et al. (2013a), CBOW is faster, while skip-gram performs better for infrequent words.

Word embeddings then became an essential part of any language model. The word representations are tweaked based on a training corpus to directly create dense vectors. They represent each feature as a vector in a low dimensional space, providing increased generalization power to the model using them (Goldberg, 2017). The main objective of Mikolov et al. (2013a) is to tackle the disadvantages from count-based representations, and thus give meaning to words in an efficient way. They use the two approaches to learn high-quality word vectors from huge datasets. The authors state that similar words tend to be close to each other, and that words have multiple degrees of similarity. It turns out that simple algebraic operations can be performed on the word vectors, for example, "$vector$(”King”) − $vector$(”Man”) + $vector$(”Woman”) ≈ $vector$("Queen"). Figure 3.1 visualizes the two approaches sketched below.

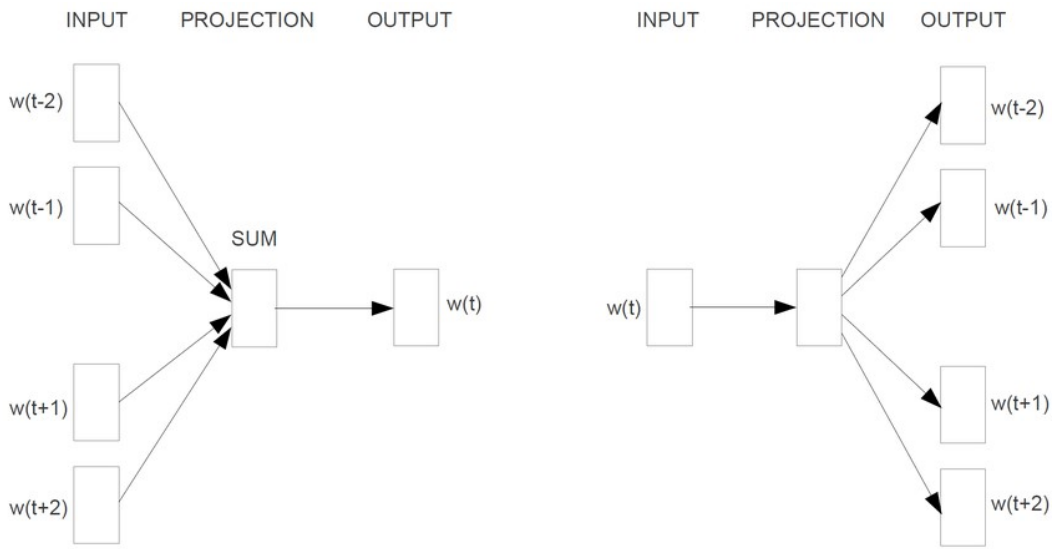

Figure 3.1: Architecture of CBOW and continuous skip-gram by Mikolov et al. (2013a).

3.2.1 CBOW

If the number of features is not known in advance, there is a need to represent an unbounded number of features using a fixed size vector (Goldberg, 2017). As illustrated in Figure 3.1, the Continuous Bag of Words (CBOW) model predicts the word of interest based on its surrounding words within a pre-defined window size. It works by either summing or averaging the context vectors $\mathbf{c}_i$ of the corresponding features (Goldberg, 2017). For the CBOW variant of Word2Vec (Mikolov et al., 2013a), a simple neural network architecture is used, consisting of one input layer describing the context words, one hidden layer, and one output layer predicting the target word. It defines the context vector to be the sum of the embedding vectors of the context components:

$$\mathbf{c} = \sum_{i=1}^{k} \mathbf{c}_i.$$

3.2.2 Skip-gram

In contrast to the CBOW, the continuous skip-gram model predicts the context words based on the word of interest (see Figure 3.1). The skip-gram variant of Word2Vec (Mikolov et al., 2013a) assumes independence between the context vectors $\mathbf{c}_i$, with $i = 1, \ldots, k$. Mikolov et al. (2013b) trained the skip-gram model using a negative-sampling procedure.
Although skip-gram is a commonly used method and effective in practice, like CBOW, it models static word embeddings. This means that the same word always has the same vector value regardless of the context, which is a major limitation.

3.3 Contextualized Word Embeddings (ELMo)

The word embeddings presented above cannot grasp the context in which the word was used. Deep contextualized word vectors are the first to successfully address this issue (Peters et al., 2018). The idea is to assign each token a representation which is a linear function of the entire input sentence. These contextualized word vectors are also called Embeddings from Language Models (ELMo) since they derive from bidirectional LSTMs (biLSTM) trained with a coupled language model objective on a large text corpus. They are called *deep* because they are a function of all internal layers of the bidirectional language model (biLM). The concept behind ELMo developed by AllenNLP (Peters

et al., 2018) is elucidated below.

As mentioned, ELMo is an LSTM-based approach. The Long Short-term Memory (LSTM) was proposed by Hochreiter and Schmidhuber (1997), and is an adaptation of the RNN (Elman, 1990). RNNs allow representing arbitrarily sized sequential inputs in fixed-size vectors while paying attention to the structured properties of the inputs (Goldberg, 2017). An LSTM unit can process entire word sequences and is usually composed of a cell, an input gate, an output gate and a forget gate for every hidden node. The cell remembers values over arbitrary time intervals, while the three gates regulate the flow of information into and out of the cell. The advantage of an LSTM cell is its cell memory unit. As a result, long-range dependencies can be maintained more effectively compared to a classic RNN.

Deep contextualized word vectors are obtained in two steps. First, an L-layer biLSTM with raw word vectors, which are computed using a character-level Convolutional Neural Network (CNN), is modeled. Principally, the biLM consists of a forward pass and a backward pass. Afterwards, the vectors are formed out of the resulting hidden layers.

Firstly, given the raw word vectors, namely a sequence of n tokens, the forward pass calculates the probability of the sequence by modeling the probability of token t_k given its history $(t_1, \ldots, t_{k-1})$. Each LSTM layer outputs a context-dependent representation $\vec{\mathbf{h}}_{k,j}^{LM}$ at each position k with $j = 1, \ldots, L$. The top layer LSTM output $\vec{\mathbf{h}}_{k,j}^{LM}$ predicts the next token t_{k+1} using a softmax layer. The backward pass is computed analogously with the difference that it runs over the sequence in reverse, i.e. it predicts the previous token given the future context. It produces representations $\overleftarrow{\mathbf{h}}_{k,l}^{LM}$ of t_k given $(t_{k+1}, \ldots, t_N)$. The biLM jointly maximizes the log-likelihood of both directions:

$$\sum_{k=1}^{N} \left(\log p \left(t_k \mid t_1, \ldots, t_{k-1}; \Theta_x, \vec{\Theta}_{LSTM}, \Theta_s \right) + \log p \left(t_k \mid t_{k+1}, \ldots, t_n; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s \right) \right).$$

Apparently, the likelihood shares some weights between both directions. The tied parameters Θ_x and Θ_s symbolize the token representation and the softmax layer, respectively, while separate parameters for the LSTMs in each direction ($\vec{\Theta}_{LSTM}$ and $\overleftarrow{\Theta}_{LSTM}$) are being retained.

For the second step, an L-layer biLM calculates $2L + 1$ representations for each token t_k, resulting in

$$R_k = \left\{ \mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L \right\} \quad \text{with} \quad \mathbf{h}_{k,j}^{LM} = \left[\overrightarrow{\mathbf{h}}_{k,j}^{LM}; \overleftarrow{\mathbf{h}}_{k,j}^{LM} \right].$$

For $j = 0$, $\mathbf{h}_{k,j}^{LM}$ represents the token layer $\mathbf{x}_k^{LM}$. The contextualized word embeddings are then computed as a task specific combination of the layer representations in R_k, resulting in a single vector:

$$\mathbf{ELMo}_k^{task} = E\left(R_k; \Theta^{task}\right) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}. \tag{3.2}$$

Here, γ^{task} is a scalar parameter allowing the task model to scale the ELMo vector and s^{task} are softmax-normalized weights.

Given a supervised NLP task and a pre-trained biLM, all the layer representations for each token are stored after running the biLM. The end task model then learns a linear combination of these representations. Hereby, the lowest layers of the supervised model without biLM are regarded first. To include ELMo, the weights of the biLM are being maintained and the ELMo vector from Equation 3.2 is concatenated with $\mathbf{x}_k$. The new vector $\left[\mathbf{x}_k; \mathbf{ELMo}_k^{task}\right]$ is then passed into the task neural network.

The final model by Peters et al. (2018) uses $L = 2$ biLSTM layers with 4,096 units, 512 dimension projections and a residual connection from the first to the second layer. According to the authors, the biLM provides three layers of representations for each input token, including those outside the training set due to the pure character input. The biLM was trained for ten epochs on the One Billion Word Benchmark (Chelba et al., 2014). Once pre-trained, the model can be used as a component for any NLP task. Peters et al. (2018) claim that in some cases, fine-tuning the model on domain specific data causes significant drops in perplexity and an increase in downstream task performance. Thus, in downstream tasks, they mostly prefer using a fine-tuned biLM.

Deep contextualized word embeddings provided a momentous stride towards better language modeling and language understanding. The approach outperformed a variety of NLP tasks that were tackled with static word embeddings before (Peters et al., 2018), such as the sentiment analysis task on the Stanford Sentiment Treebank (SST-5; Socher et al.,

2013). Nevertheless, biLSTMs show a lack of efficiency. The Transformer introduced by Vaswani et al. (2017) addresses the issue.

3.4 The Transformer Architecture

In 2017, LSTM-based language models were practically replaced with Transformer-based language models. The model is more efficient and more parallelizable as there is no need for RNNs or CNNs. Moreover, the method generalizes well to other NLP tasks and outperformed a couple of language tasks by that time (Vaswani et al., 2017).

Figure 3.2 illustrates the model architecture of the vanilla Transformer. Visibly, the backbone of the architecture is an encoder-decoder mechanism (Cho et al., 2014) which was already used in the RNN context before. In such an architecture, one model encodes the input sequence into hidden representations, which are subsequently decoded into the output by a second model. This structure allows to output a sequence with a possibly different length than the input sequence. To further optimize encoder-decoder models, Bahdanau et al. (2016) introduced an attention mechanism, which was commonly referred to as additive attention. The attention mechanism is a method that allows the modeling of dependencies regardless of their distance in the input or output sequences. More specifically, rather than encoding the input into a single summary vector, the decoder uses the output from each hidden state (i.e. the context vectors). Following Bahdanau et al. (2016), the attention mechanism enables the decoder to more effectively select relevant and irrelevant information since individual hidden representations are retained. However, the Transformer solely relies on Self-Attention. This is an attention mechanism which relates different positions of a single sequence to calculate a representation of the sequence. As a result, long-distance dependencies can be captured more efficiently.

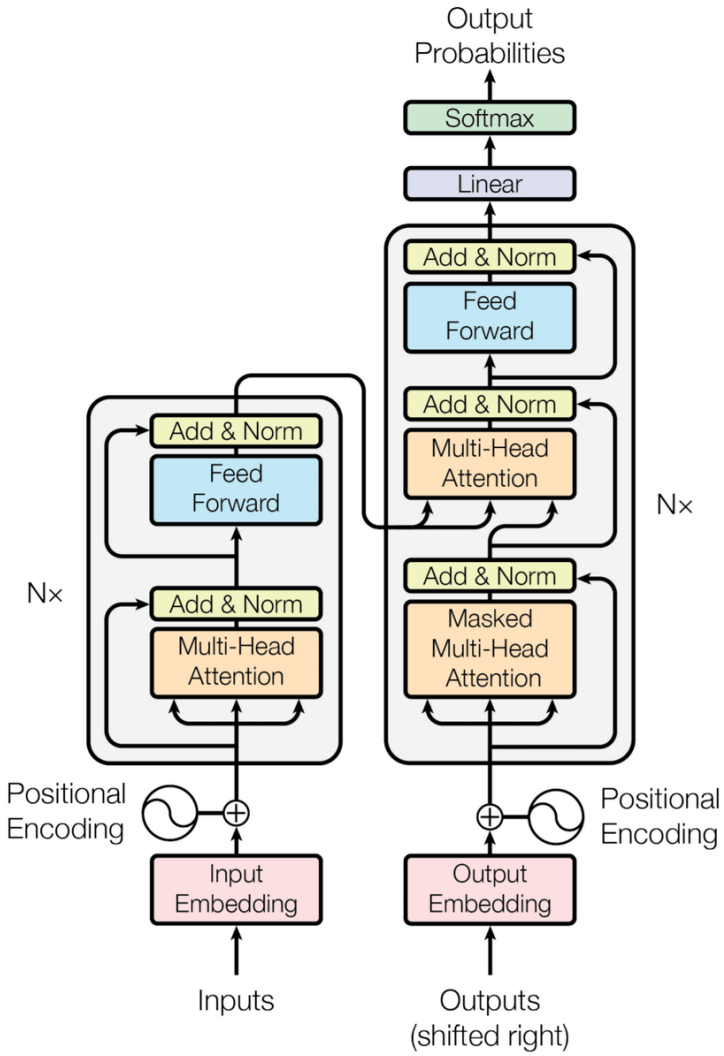

Figure 3.2: The Transformer architecture by Vaswani et al. (2017).

The Transformer architecture works as follows. To convert input and output tokens to vectors of dimension $d_{model} = 512$, pre-trained embeddings are utilized. These are trained on the standard WMT 2014 English-German dataset consisting of about 4.5 million sentence pairs and on the WMT 2014 English-French dataset consisting of about 36 million sentences. The sentences were encoded using byte-pair encoding[3] (BPE; Gage, 1994; Sennrich et al., 2016), which has a shared source-target vocabulary of about 37,000 tokens. The tokens were split into a 32,000 word-piece vocabulary for English-French. Sentence pairs were batched comprising approximately 25,000 source tokens as well as 25,000 target tokens. Then, the learned embeddings serve as inputs to

[3]form of data compression in which the most common pair of consecutive bytes of data is replaced with a byte that does not occur within that data. In addition, a table of the replacements is required to rebuild the original data.

the encoder, while the decoder takes its output embeddings shifted by one position as input. The offset guarantees that the predictions for position i can depend only on the known outputs at positions prior i.

Both the bottom encoder and decoder stacks take the positional encoding (PE) of the corresponding input embeddings into account in order to make use of the order of the sequence. The PEs are calculated as

$$PE_{(pos,2i)} = \sin\left(pos/10000^{2i/d_{model}}\right),$$
$$PE_{(pos,2i+1)} = \cos\left(pos/10000^{2i/d_{model}}\right).$$

Noticeably, each dimension of the PEs represents a sinusoid. Here, i is the dimension of the token embedding and pos corresponds to the position of the token. The PEs show the same dimension (d_{model}) as the embeddings so that the two can be summed. The authors choose this function because they hypothesize, "it would allow the model to easily learn to attend by relative positions, since for any fixed offset k, PE_{pos+k} can be represented as a linear function of PE_{pos}" (Vaswani et al., 2017, p. 6).

Both the encoder and decoder are formed of $N = 6$ layers, each consisting of two and three sublayers, respectively (see Figure 3.2). The sublayers are a multi-head Self-Attention mechanism and a position-wise fully connected feed-forward network. The additional sublayer regarding the decoder computes Multi-Head Attention over the output of the encoder stack. Around each sublayer, residual connection and layer normalization[4] is applied. In the decoder stack, the Self-Attention sublayer is modified to prevent positions from attending to subsequent positions.

Figure 3.3 visualizes the computation of Multi-Head Attention on the right-hand side. Multi-Head Attention takes the linear projection of the queries ($\mathbf{Q}$), the keys ($\mathbf{K}$) and the set of values ($\mathbf{V}$) as input to compute the Scaled Dot-Product Attention (see graph on the left side of Figure 3.3). While the Scaled Dot-Product Attention performs a single attention function with d_{model}-dimensional keys, the Multi-Head Attention consists of h attention layers running in parallel.

[4]The output of each sublayer is layernorm(x+sublayer(x)) where sublayer(x) is a function implemented by the sublayer itself.

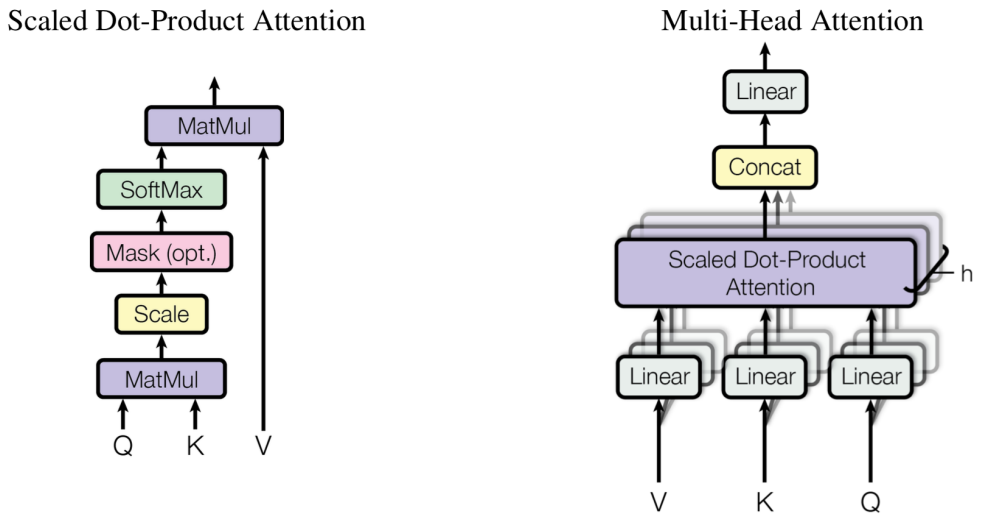

Figure 3.3: The Self-Attention mechanism by Vaswani et al. (2017).

Within the encoder-decoder attention layers, the queries come from the previous decoder layer, whereas the memory keys and values come from the output of the encoder. The encoder contains Self-Attention layers. That means that all the keys, values and queries come from the previous layer in the encoder. The Self-Attention layers in the decoder allow each position in the decoder to attend to all positions in the decoder up to and including that position. Here, the model needs to prevent leftward information flow to preserve the auto-regressive property. The authors implement this inside of the Scaled Dot-Product Attention by masking out (i.e. setting to $-\infty$) all values in the input of the softmax which correspond to illegal connections.
The computation of the Multi-Head Attention yielding d_v-dimensional output values can be formulated as

$$\begin{aligned} & \text{MultiHead}(Q, K, V) = \text{Concat}\left(\text{head}_1, \ldots, \text{head}_A\right) \mathbf{W}^O \\ \text{with} \quad & \text{head}_i = \text{Attention}\left(\mathbf{Q}\mathbf{W}_i^{\mathbf{Q}}, \mathbf{K}\mathbf{W}_i^{\mathbf{K}}, \mathbf{V}\mathbf{W}_i^{\mathbf{V}}\right). \end{aligned}$$

The set of queries $\mathbf{Q}$ and the keys $\mathbf{K}$ have dimension d_k, respectively. The values $\mathbf{V}$ have dimension d_v. The projections are represented by the parameter matrices $\mathbf{W}_i^{\mathbf{Q}} \in \mathbb{R}^{d_{model} \times d_k}$, $\mathbf{W}_i^{\mathbf{K}} \in \mathbb{R}^{d_{model} \times d_k}$, $\mathbf{W}_i^{\mathbf{V}} \in \mathbb{R}^{d_{model} \times d_v}$ and $\mathbf{W}^O \in \mathbb{R}^{hd_v \times d_{model}}$. As visualized in Figure 3.3, Vaswani et al. (2017) define the Scaled Dot-Product Attention function as

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}.$$

The resulting heads are then concatenated and once again projected, yielding the final values. Here, $A = 8$, i.e. the authors implemented 8 heads, with $d_v = d_k = d_{model}/A = 512/8 = 64$.

As already mentioned above, the other sublayer embodies a position-wise fully connected feed-forward network. This comprises two linear transformations with a ReLU function in between and yields different parameters from layer to layer. Finally, the learned linear transformation and softmax function are used to convert the decoder output to the predicted next-token probabilities.

Each attention head can capture a different property in the sentence. This ability makes it possible to grasp the context, which is particularly useful in sequence tagging tasks like Named Entity Recognition (NER), POS tagging or Semantic Role Labeling (SRL) (Raganato and Tiedemann, 2018). Two of the currently most popular applications of the Transformer are the Generative Pre-trained Transformer (GPT), introduced by Radford et al. (2018) (and its successors GPT-2 (Radford et al., 2019) and GPT-3 (Brown et al., 2020)), and BERT, proposed by Google (Devlin et al., 2019).

3.5 BERT

The main difference between the two Transformer-based language models - GPT and BERT - is that GPT is an unidirectional language model, whereas BERT is bidirectional. According to Devlin et al. (2019), unidirectional models are suboptimal for sentence-level tasks and harmful for fine-tuning with token-level tasks as they only regard the context of a token from one direction, e.g. its history. In contrast, BERT grasps the idea of bidirectional context from ELMo and applies it to the Transformer. As a result, BERT cracked transfer learning for NLP in 2018 and advanced the state-of-the-art for several language tasks (Devlin et al., 2019). Therefore, BERT is considered for re-evaluating GermEval 2017.

Basically, BERT applies pre-trained language representations on downstream tasks by fine-tuning the pre-trained weighting parameters. The model architecture fully relies on the multi-layer bidirectional Transformer encoder based on the original implementation by Vaswani et al. (2017) which is described in the section before. Devlin et al. (2019) differentiate between the $\text{BERT}_{\text{BASE}}$ model consisting of $L = 12$ layers, a hidden size of $H = 768$ and $A = 12$ Self-Attention heads, and the BERT_{LARGE} with $L = 24$,

$H = 1024$ and $A = 16$. $\text{BERT}_{\text{BASE}}$ has a total of 110 million parameters and BERT_{LARGE} incorporates 340 million parameters.

The outline of the present section is as follows. First, the input representations of BERT are described (Subsection 3.5.1). The procedure of pre-training BERT is sketched in Subsection 3.5.2. In Subsection 3.5.3, the BERT-CRF model, which refers to BERT with a CRF layer, are explained. This approach is commonly used for sequence labeling tasks and will later be applied to Subtask D of GermEval 2017. Afterwards, in Subsection 3.5.4, some popular variants of BERT are elucidated, including RoBERTa, ALBERT and DistilBERT. The last part of this section displays which pre-trained language models for German documents are available and utilized for the re-evaluation of GermEval 2017.

3.5.1 Input Representations

Within BERT, an input embedding can represent a single sentence as well as a pair of sentences in one token sequence in order to tackle a variety of downstream tasks. As stated by Devlin et al. (2019), a "sentence" is related to an arbitrary span of connected text and not to a sentence in the linguistic sense. A "sequence" stands for the input token sequence to BERT, e.g. a single sentence or two bundled sentences. The input representations are tokenized using WordPiece embeddings (Wu et al., 2016) with a vocabulary of 30,000 tokens. Figure 3.4 gives an example of an input representation and how it is constructed. For a given token, its input representation is composed by summing the corresponding position, segment and token embeddings.

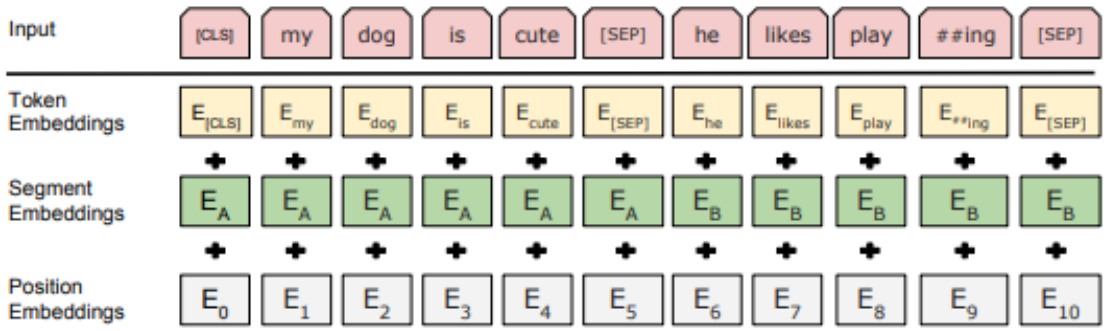

Figure 3.4: Example of a BERT input representation by Devlin et al. (2019). E symbolizes the input embedding.

The first token of each input sequence corresponds to a particular classification token (`[CLS]`). The final hidden vector related to this token is utilized as the aggregated

sequence representation for classification tasks and is denoted as $C \in \mathbb{R}^H$. Furthermore, the sentences which are concatenated into a single sequence are separated with a particular separation token (`[SEP]`). Afterwards, they are enriched with a learned embedding to every token which indicates if it belongs to sentence `A` or `B`. The final hidden vector of the i-th input token is denoted as $T_i \in \mathbb{R}^H$.

3.5.2 Pre-Training

Basically, Devlin et al. (2019) used two unsupervised tasks to pre-train BERT: a masked language model (MLM) and next sentence prediction (NSP). As pre-training corpus, the authors consider the Toronto `BookCorpus` (Zhu et al., 2015), including 800 million words, and English `Wikipedia`, consisting of 2,500 million words.

The MLM accounts for the bidirectionality property of the model. At this point, some percentage - here 15% - of the input tokens are masked at random. As in the standard language model, the final hidden vectors corresponding to the mask tokens are fed into an output softmax over the vocabulary. Thereby, the model should predict the masked words. However, the `[MASK]` token does not appear during fine-tuning. Therefore, a disadvantage of this approach is that it builds a mismatch between pre-training and fine-tuning. The authors suggest not always replacing the masked word with the actual `[MASK]` token. Concretely, 10% of the masked words obtain a random token (e.g. "apple" instead of "dog"), further 10% of them keep the unchanged token and the remaining 80% of the masked words obtain the `[MASK]` token. T_i is then utilized to predict the original token with cross entropy loss.

The aim of the second pre-training task, namely NSP, is to understand the relationship between two sentences. The model is pre-trained for a binarized NSP task. This can be trivially generated from any monolingual corpus. When choosing sentence `A` and sentence `B` for each pre-training example, 50% of the time, sentence `B` is a random sentence from the corpus, which is tagged as `NotNext`. The other 50% of the time, `B` is the actual next sentence which follows `A`, and is labeled as `IsNext`. Vector `C` is then used for NSP.

Compared to pre-training, fine-tuning BERT is relatively inexpensive. It is straightforward as the Self-Attention mechanism allows BERT to tackle many downstream tasks by swapping out the appropriate inputs and outputs. Devlin et al. (2019) showed that

BERT can effectively solve a variety of NLP tasks. The language model advanced the state-of-the-art on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019) and the Stanford Question Answering Dataset (SQuAD) task (Rajpurkar et al., 2016, 2018). They discovered that BERT_{LARGE} significantly outperforms $\text{BERT}_{\text{BASE}}$ across all their experiments. The authors further recommend a range of hyperparameters working well across different tasks. However, they point out that the optimal hyperparameter values are task-specific and that large datasets (at least 100,000 documents) are usually less sensitive to varying hyperparameter choices compared to smaller datasets.

3.5.3 BERT-CRF for Sequence Labeling

For sequence tagging, putting a Conditional Random Field (CRF) as classification layer on top of the model is a popular and often successful strategy to enhance LSTM-based (Huang et al., 2015; Lample et al., 2016; Reimers and Gurevych, 2017; Li et al., 2019a) as well as Transformer-based language models like BERT (Li et al., 2019b; Mao and Liu, 2019; Arkhipov et al., 2019; Souza et al., 2020; Trautmann et al., 2020). CRFs calculate the joint probability for a whole sequence rather than individual label probabilities. So, the idea is to jointly model tagging decisions in order to find the most probable tag sequence. Furthermore, the procedure is especially popular for sequence labeling tasks such as NER because it assures that strict independence between observations is not necessary. For instance, in an NER task, `B-PER` ("Begin-Person Entity") cannot be followed by `I-LOC` ("Inner-Location Entity") since every entity has to start with a `B-` tag for "Beginning". CRFs were repeatedly considered for Subtask D of GermEval 2017 as OTE is also a special sequence labeling task (Mishra et al., 2017; Lee et al., 2017; Ruppert et al., 2017).

CRFs were first introduced by Lafferty et al. (2001). The authors generally define a CRF as follows:

> **Definition.** Let $G = (V, E)$ be a graph such that $\mathbf{Y} = (Y_v)_{v \in V}$, so that $\mathbf{Y}$ is indexed by the vertices of G. Then $(\mathbf{X}, \mathbf{Y})$ is a conditional random field in case, when conditioned on $\mathbf{X}$, the random variables Y_v obey the Markov property with respect to the graph: $p(Y_v \mid \mathbf{X}, Y_w, w \neq v) = p(Y_v \mid \mathbf{X}, Y_w, w \sim v)$, where $w \sim v$ means that w and v are neighbors in G (Lafferty et al., 2001, p. 5).

This means that a CRF is an undirected graphical model whose nodes can be divided into exactly two disjoint sets $\mathbf{X}$ and $\mathbf{Y}$ - the observed and output variables. The

conditional distribution $p(\mathbf{Y} \mid \mathbf{X})$ is then modeled.

In sequence labeling tasks, $\mathbf{X}$ represents a random variable over token sequences to be labeled and $\mathbf{Y}$ is a random variable over corresponding label sequences. The random variables $\mathbf{X}$ and $\mathbf{Y}$ are then jointly distributed such that the conditional probability $p(\mathbf{X} \mid \mathbf{Y})$ can be constructed without explicitly computing the marginal $p(\mathbf{X})$ (Lafferty et al., 2001). Then, the graph of interest is a chain graph $G = (V, E)$ with the vertices $V = \{1, 2, \ldots, m\}$ and the edges $E = \{(i, i+1)\}$. Figure 3.5 illustrates the structure of the chain graph. Consequently, for $\mathbf{X} = (X_1, \ldots, X_n)$, and $\mathbf{Y} = (Y_1, \ldots, Y_n)$, the Y_i are structured to form a linear chain, with an edge between each Y_i and Y_{i+1}.

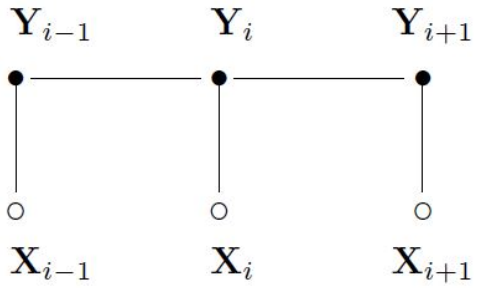

Figure 3.5: Chain graph structure of a CRF for sequences by Lafferty et al. (2001). An open circle indicates that the variable is not generated by the model.

The BERT-CRF model for sequence tagging is then computed as follows. For an input sequence of n tokens, BERT outputs an encoded token representation of hidden dimension H. The language model maps the encoded sequence of each token to the label space, i.e. $\mathbb{R}^H \mapsto \mathbb{R}^K$, where K symbolizes the number of distinct labels in the data. Then, the matrix of output scores $\mathbf{P} \in \mathbb{R}^{n \times K}$ of BERT is fed to the CRF layer (Souza et al., 2020). Following Lample et al. (2016), for a token sequence $\mathbf{x}$ and a sequence of label predictions $\mathbf{y}$, where $y_i \in \{1, \ldots, K\}$, the score of the sequence is calculated as

$$s(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=1}^{n} P_{i, y_i},$$

where $P_{i,j}$ represents the score of the j-th label and the i-th word in a sentence. $\mathbf{A} \in \mathbb{R}^{K+2 \times K+2}$ is a square matrix of tag transition scores, such that $A_{i,j}$ represents the score of a transition from label i to label j. y_0 and y_{n+1} are the start and end tags of a sentence, respectively, that are included in $\mathbf{A}$. A softmax over all the possible tag sequences provides

the probability for the sequence $\mathbf{y}$:

$$p(\mathbf{y} \mid \mathbf{x}) = \text{softmax}(s(\mathbf{x}, \mathbf{y})) = \frac{\exp\left(s(\mathbf{x}, \mathbf{y})\right)}{\sum_{\tilde{\mathbf{y}} \in \mathbf{Y_x}} \exp\left(s(\mathbf{x}, \tilde{\mathbf{y}})\right)}.$$

Here, $\mathbf{Y_x}$ represents all possible label sequences for sentence $\mathbf{x}$. The log-likelihood of the correct tag sequence is then maximized during training. This results in

$$\log(p(\mathbf{y} \mid \mathbf{x})) = s(\mathbf{x}, \mathbf{y}) - \log\left(\sum_{\tilde{\mathbf{y}} \in \mathbf{Y_x}} \exp(s(\mathbf{x}, \tilde{\mathbf{y}}))\right). \tag{3.3}$$

The summation in Equation 3.3 can be computed using dynamic programming. This procedure encourages the network to produce a valid sequence of output labels. While decoding, the label sequence with the highest scores is regarded as output:

$$\mathbf{y}^* = \underset{\tilde{\mathbf{y}} \in \mathbf{Y_x}}{\text{argmax}}\, s(\mathbf{x}, \tilde{\mathbf{x}}).$$

Here, the most likely sequence $\mathbf{y}^*$ is obtained using Viterbi search (Viterbi, 1967).

The original implementation of BERT (Devlin et al., 2019) uses softmax classification, reaching close to state-of-the-art results. Nevertheless, due to the theoretical advantages and the recent evidence that CRFs can benefit token-level language models, we model and compare BERT and BERT-CRF for Subtask D.

3.5.4 Variants of BERT

Since the release of BERT, many variations of BERT were developed, which all aim to improve the model. In this subsection, three popular model variants are portrayed: RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2020) and DistilBERT (Sanh et al., 2020). It is noteworthy that RoBERTa and ALBERT have neither a multilingual nor a German version yet.

3.5.4.1 RoBERTa

Liu et al. (2019) conducted a replication study of BERT's pre-training, measuring the impact of several key hyperparameters as well as training data size. The authors state that BERT is significantly undertrained and provided an improved version for training BERT models called Robustly optimized BERT approach (RoBERTa).
The following modifications were considered for RoBERTa. First, Liu et al. (2019) trained

the model for a longer time, with bigger batches, and over more data. Precisely, the model was trained for 100,000 steps over `BookCorpus` (Zhu et al., 2015) and `Wikipedia` (same corpora as for BERT), `CC-News` (collected from CommonCrawl News dataset; Nagel, 2016), `OpenWebText` (Gokaslan and Cohen, 2019) and `Stories` (Trinh and Le, 2019), resulting in a total of 160GB of uncompressed text. `CC-News` was collected by the authors themselves and is of comparable size to other privately used datasets in order to provide better control for impacts of the training set size. The authors basically pre-trained the model according to the architecture of BERT_{LARGE}. In addition, they used large mini-batches, larger byte-level BPE than Devlin et al. (2019) and 1024 V100 GPUs. Secondly, the authors removed the NSP objective because it did not contribute to any improvement of the model. They performed some experiments and finally received better results without NSP loss. Additionally, RoBERTa considers `Full-Sentences`, i.e. each input is packed with full sentences which are sampled contiguously from one or more documents so that the total length does not exceed 512 tokens. Third, Liu et al. (2019) utilized dynamic masking instead of static masking. That means the model changes masking position in every epoch. Hence, the pre-training model gradually adapts to different masking techniques and learns different representations (Wang et al., 2020). Lastly, RoBERTa was trained on longer sequences with 512 tokens at the most.

These changes resulted in better performances compared to BERT. As a result, RoBERTa advanced the state-of-the-art on a couple of well-known datasets such as GLUE (Wang et al., 2019) and SQuAD (Rajpurkar et al., 2016, 2018).

3.5.4.2 ALBERT

Given that BERT has millions of parameters, it can easily hit memory limits. A Lite BERT (ALBERT) improves the parameter-efficiency of BERT by reducing its number of parameters without significantly hurting performance (Lan et al., 2020). The model training of ALBERT_{LARGE} compared to BERT_{LARGE} is about 1.7 times faster with 18 times fewer parameters. The improving technique also acts as a form of regularization, i.e. it stabilizes the training and helps with generalization. Mainly, Lan et al. (2020) carried out three major changes on BERT, namely a factorized embedding parameterization, a cross-layer parameter sharing and an inter-sentence coherence loss.

BERT's model architecture chooses the WordPiece embedding size E to be equal to the hidden layer size H. Lan et al. (2020) claim that this is suboptimal because the

WordPiece embeddings are designed to learn context-independent representations, but the hidden-layer embeddings are intended to learn context-dependent representations. Consequently, the modeling requirements demand $H \gg E$ since loosening E from H permits the model to use all its parameters more efficiently. In addition, for NLP, it is common to use large vocabularies. For instance, BERT incorporates $V = 30,000$ tokens. The bigger H gets, the more it would increase the size of the embedding matrix of size $V \times E$ if $E \equiv H$, which could rapidly lead to a model with billions of parameters. To counteract this, Lan et al. (2020) split the embedding matrix into two smaller matrices. This means that they project the one-hot vectors first into a lower dimensional embedding space of size E and then into the hidden space, leading to a significant parameter reduction when $H \gg E$. The authors refer to this as factorized embedding parameterization.

The other parameter-reduction technique ALBERT makes use of is cross-layer parameter sharing. The authors decided on sharing all parameters across layers as this prevents the parameters from growing with the depth of the network.

Furthermore, BERT's NSP objective was replaced by a self-supervised loss for sentence-order prediction (SOP) which focuses on inter-sentence coherence and is designed to address the ineffectiveness of NSP. The SOP objective uses the same strategy as BERT to generate positive examples for the `IsNext` prediction. As negative examples, ALBERT considers the same two sentences but with swapped order, arguing this would compel the model to learn more fine-grained distinctions about coherence properties at discourse-level. Accordingly, this benefits downstream task performance of multi-sentence encoding tasks.

Lan et al. (2020) provide four variants of ALBERT: $\text{ALBERT}_{\text{BASE}}$ with 11 million parameters (H=768, 12-layer network), ALBERT_{LARGE} with 17 million parameters (H=1024, 24-layer network), ALBERT_{XLARGE} with 58 million parameters (H=2048, 24-layer network) and $\text{ALBERT}_{XXLARGE}$ with 223 million parameters (H=4096, 12-layer network). For comparison, BERT_{LARGE} has 340 million parameters (H=1024, 24-layer network). $\text{ALBERT}_{XXLARGE}$ advanced the state-of-the-art on GLUE (Wang et al., 2019) and SQuAD (Rajpurkar et al., 2016) benchmark studies, outperforming BERT and RoBERTa.

3.5.4.3 DistilBERT

The distilled version of BERT (DistilBERT) by Sanh et al. (2020) reduces the size of BERT by 40% while retaining 97% of its performance and being 60% faster. The approach uses a triple loss combining language modeling, knowledge distillation and cosine-distance losses.

Knowledge distillation

Knowledge distillation (Bucila et al., 2006) is a compression method in which a small model - the student - is trained to mimic the behavior of a larger model - the teacher - or an ensemble of models. The knowledge is thus characterized by the full output distribution of the teacher network. By using a cross entropy loss over the soft targets, i.e. the probabilities of the teacher, the knowledge is transmitted from the teacher to the student. The distillation loss is then determined by

$$L_{ce} = -\sum_i t_i \times \log(s_i),$$

where t_i (resp. s_i) is a probability estimated by the teacher (resp. the student). Since a single example enforces much more constraint than a single hard target, this loss results in a rich training signal. The authors further considered a softmax temperature (Hinton et al., 2015):

$$p_i = \frac{\exp\left(z_i/T\right)}{\sum_j \exp\left(z_j/T\right)},$$

where z_i is the model score for class i. The temperature T controls the smoothness of the output distribution and the same value of T is applied to the student and the teacher at training time. T is set to 1 at inference to beget a standard softmax function. The final training loss is a linear combination of the distillation loss L_{ce} and the supervised training loss, which corresponds to the MLM loss L_{mlm}. In order to match the directions of the student and teacher hidden states vectors, Sanh et al. (2020) also include a cosine embedding loss L_{cos}.

Model architecture

DistilBERT represents the student, whereas BERT embodies the teacher. Further, DistilBERT has the same model architecture as BERT. What changes is that the number of layers is halved, and toke-type embeddings and pooler are taken off. In order to find the right model initialization for the sub-network to converge, Sanh et al. (2020) initialize

DistilBERT from BERT by taking one layer out of two.

Similar to RoBERTa, DistilBERT is distilled on very large batches leveraging gradient accumulation using dynamic masking and eliminating the NSP loss. Moreover, it uses the same corpus as BERT - `Wikipedia` and the `BookCorpus` (Zhu et al., 2015) - and is trained on 8 16GB V100 GPUs.

3.5.5 Pre-Trained Language Models for German

To re-evaluate the GermEval 2017 Task, language models which are pre-trained on a German corpus are needed. The `transformers` module released by Hugging Face (Wolf et al., 2020) officially includes the pre-trained language models displayed in Table 3.1.

`transformers` language model	corpus	properties
$\text{BERT}_{\text{BASE}}$ German cased	cased German text by Deepset.ai	L=12, H=768, A=12, 110M parameters
$\text{BERT}_{\text{BASE}}$ German dbmdz cased	cased German text by DBMDZ	L=12, H=768, A=12, 110M parameters
$\text{BERT}_{\text{BASE}}$ German dbmdz uncased	uncased German text by DBMDZ	L=12, H=768, A=12, 110M parameters
$\text{BERT}_{\text{BASE}}$ multilingual cased	cased text from the largest Wikipedias in the top 104 languages	L=12, H=768, A=12, 179M parameters
$\text{BERT}_{\text{BASE}}$ multilingual uncased	uncased text from the largest Wikipedias in the top 102 languages	L=12, H=768, A=12, 168M parameters
$\text{DistilBERT}_{\text{BASE}}$ German cased	uncased German text by DBMDZ	L=6, H=768, A=12, 66M parameters
$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	cased text from the largest Wikipedias in the top 104 languages	L=6, H=768, A=12, 134M parameters

Table 3.1: Pre-trained language models provided by Hugging Face's `transformers` (version 4.0.1) which are suitable for German. The complete list of incorporated language models is available at https://huggingface.co/transformers/pretrained_models.html.

The table shows that the `transformers` module includes three German models by DBMDZ[5] ("Digitale Bibliothek/Münchener DigitalisierungsZentrum") and one by Deepset.ai[6]. Deepset.ai use German `Wikipedia` (6GB of raw text files), the `Open Legal Data` dump (2.4GB; Ostendorff et al., 2020) and news articles (3.6GB) as corpus. DBMDZ combine `Wikipedia`, `EU Bookshop` (Skadiņš et al., 2014), `Open Subtitles` (Lison and Tiedemann, 2016), `CommonCrawl` (Ortiz Suárez et al., 2019), `ParaCrawl` (Esplà-Gomis et al., 2019) and `News Crawl` (Haddow, 2018) to a corpus, resulting in a dataset with a total size of 16GB and 2,350 million tokens. Besides, Hugging Face provides three multilingual models which are also suitable for German. All in all, there are five BERT and two DistilBERT models at disposal, among which there are two language models that were trained on a lower-cased ("uncased") corpus. The rest was trained on the true-cased ("cased") corpus. The upcoming chapter presents the results of the re-evaluation of the GermEval 2017 Shared Task considering these seven pre-trained models.

[5]MDZ Digital Library team at the Bavarian State Library. Visit https://www.digitale-sammlungen.de for details and https://github.com/dbmdz/berts for their repository on pre-trained BERT models.

[6]Visit https://deepset.ai/german-bert for details.

Chapter 4

Results of Re-Evaluating GermEval 2017

For the re-evaluation, the GermEval 2017 data examples provided by the organizers in XML format were re-analyzed. The text data contains some duplicates, but these were not removed to make the results as comparable as possible. The data was hardly changed: The documents were tokenized and single spelling mistakes in the labels were fixed[1]. For Subtask D, the BIO-tags were added based on the provided sequence positions.

The models were trained on one Tesla V100 PCIe 16GB GPU and Python, version 3.8.7. Moreover, the `transformers` module, version 4.0.1, by Hugging Face and `torch`, version 1.7.1, by PyTorch were used[2]. The hyperparameters for fine-tuning were chosen following the recommendations of Devlin et al. (2019). These are

- batch size $\in \{16, 32\}$,
- learning rate (for Adam optimizer) $\in \{5 \times 10^{-5}, 3 \times 10^{-5}, 2 \times 10^{-5}\}$,
- epochs $\in \{2, 3, 4\}$.

Due to memory limitations, not every hyperparameter combination was applicable. According to this fact, a hyperparameter combination that was feasible for all pre-trained models on all subtasks was chosen. Finally, all the language models were fine-tuned

[1] Regarding the polarity labels, "positve" in the training data was replaced with "positive" and " negative" in the diachronic test data was replaced with "negative".

[2] The source code is available on GitHub: https://github.com/ac74/masterthesis_germeval2017. See also Appendix C for details on the electronic annex. The results are fully reproducible for Subtasks A, B and C. For Subtask D, the results should also be reproducible but for some reason they are not, and we could not fix the issue. However, the micro F1 scores only fluctuate between +/-0.01.

with a learning rate of 5×10^{-5} and four epochs. The maximum sequence length was set to 256 and a batch size of 32 was chosen. We further experimented with other hyperparameter combinations for Subtask D. The experimental results are presented in Appendix B.

This chapter presents the results of the re-evaluation and compares them with the best results from 2017. All scores are rounded to the third decimal place.

4.1 Subtask A

The Relevance Classification is a binary document classification task with classes `true` and `false`. Table 4.1 displays the micro F1 score obtained by each language model on each test dataset, along with the rank. The best result per dataset is printed in bold.

	Test syn		Test dia	
Language model	**Score**	**Rank**	**Score**	**Rank**
Best model 2017	0.903	8	0.906	8
$\text{BERT}_{\text{BASE}}$ German cased	0.950	3	0.939	4
$\text{BERT}_{\text{BASE}}$ German dbmdz cased	0.951	2	0.946	2
$\text{BERT}_{\text{BASE}}$ German dbmdz uncased	**0.957**	1	**0.948**	1
$\text{BERT}_{\text{BASE}}$ multilingual cased	0.942	6	0.933	6
$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.944	4	0.939	4
$\text{DistilBERT}_{\text{BASE}}$ German cased	0.944	4	0.939	3
$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.941	7	0.932	7

Table 4.1: Results for Subtask A on synchronic (syn) and diachronic (dia) test datasets. The score refers to the micro-averaged F1 score. The best model 2017 is the xgboost model by Sayyed et al. (2017).

All the models outperform the best result achieved in 2017 for both test datasets. For the synchronic test dataset, the previous result is surpassed by 3.8-5.4 percent points. For the diachronic test set, the absolute difference to the best contender of 2017 varies between 2.6 and 4.2 percent points. With a micro F1 score of 0.957 and 0.948, respectively, the best scoring pre-trained language model is the uncased German $\text{BERT}_{\text{BASE}}$ by DBMDZ, followed by the cased version of the model. All the pre-trained models perform slightly better on the synchronic test data than on the diachronic data.

4.2 Subtask B

Subtask B refers to the Document-level Polarity, which is a document classification task with three classes: `negative`, `neutral` and `positive`. Table 4.2 demonstrates the performances on each test dataset.

	Test syn		Test dia	
Language model	**Score**	**Rank**	**Score**	**Rank**
Best model 2017	0.767	8	0.750	8
$\text{BERT}_{\text{BASE}}$ German cased	0.798	3	0.793	2
$\text{BERT}_{\text{BASE}}$ German dbmdz cased	0.799	2	0.785	3
$\text{BERT}_{\text{BASE}}$ German dbmdz uncased	**0.807**	1	**0.800**	1
$\text{BERT}_{\text{BASE}}$ multilingual cased	0.790	5	0.780	4
$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.784	6	0.766	7
$\text{DistilBERT}_{\text{BASE}}$ German cased	0.798	3	0.776	5
$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.777	7	0.770	6

Table 4.2: Results for Subtask B on synchronic (syn) and diachronic (dia) test datasets. The score refers to the micro-averaged F1 score. The best model 2017 is the system by the organizers (Ruppert et al., 2017).

All the models outperform the best result from 2017 by 1.0-4.0 percent points for the synchronic test data, and 1.6-5.0 percent points for the diachronic test dataset. Again, the best model is the uncased German $\text{BERT}_{\text{BASE}}$ model by DBMDZ with a score of 0.807, which is followed by the cased variant with 0.799. For the diachronic test data, the uncased German $\text{BERT}_{\text{BASE}}$ model exceeds the other models with a score of 0.800, followed by the cased German $\text{BERT}_{\text{BASE}}$ model reaching a score of 0.793. Thus, the uncased German $\text{BERT}_{\text{BASE}}$ model by DBMDZ is the only model surpassing a score of 0.8 on both test datasets. All in all, the three multilingual pre-trained models perform worse than the German-only models on Subtask B. Besides, all the models perform slightly better on the synchronic dataset than on the diachronic one.

4.3 Subtask C

Subtask C is split into aspect (Subtask C1) and aspect+sentiment classification (Subtask C2), each being a multilabel classification task[3]. As the organizers provide 20 aspect

[3]As common for multilabel classification, a sigmoid was considered instead of the softmax layer, combined with a binary cross entropy loss.

categories, Subtask C1 includes 20 labels, whereas Subtask C2 deems 60 labels since each aspect category can be combined with each of the three sentiments. Same as with Lee et al. (2017) and Mishra et al. (2017), multiple mentions of the same label are not taken into account. Table 4.3 refers to the results for Subtask C1.

	Test syn		Test dia	
Language model	**Score**	**Rank**	**Score**	**Rank**
Best model 2017	0.537	8	0.556	8
$\text{BERT}_{\text{BASE}}$ German cased	0.756	2	0.762	4
$\text{BERT}_{\text{BASE}}$ German dbmdz cased	0.756	3	0.781	2
$\text{BERT}_{\text{BASE}}$ German dbmdz uncased	**0.761**	1	**0.791**	1
$\text{BERT}_{\text{BASE}}$ multilingual cased	0.706	7	0.734	7
$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.723	5	0.752	5
$\text{DistilBERT}_{\text{BASE}}$ German cased	0.738	4	0.768	3
$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.716	6	0.744	6

Table 4.3: Results for Subtask C1 (only aspect) on synchronic (syn) and diachronic (dia) test datasets. The score refers to the micro-averaged F1 score. The best model 2017 is the system by the organizers (Ruppert et al., 2017).

All the pre-trained models clearly surpass the best performance from 2017. Regarding the synchronic test set, the absolute difference ranges between 16.9 and 22.4 percent points, while for the diachronic test data, the models outperform the previous results by 17.8-23.5 percent points. The best model is again the uncased German $\text{BERT}_{\text{BASE}}$ model by DBMDZ, reaching a score of 0.761 and 0.791, respectively, followed by the two cased German $\text{BERT}_{\text{BASE}}$ models. One more time, the multilingual models show the lowest performances amongst the pre-trained language models. Next, the results for Subtask C2 are illustrated in Table 4.4.

	Test syn		Test dia	
Language model	**Score**	**Rank**	**Score**	**Rank**
Best model 2017	0.396	8	0.424	8
$\text{BERT}_{\text{BASE}}$ German cased	0.634	2	0.663	4
$\text{BERT}_{\text{BASE}}$ German dbmdz cased	0.628	4	0.663	2
$\text{BERT}_{\text{BASE}}$ German dbmdz uncased	**0.655**	1	**0.689**	1
$\text{BERT}_{\text{BASE}}$ multilingual cased	0.571	6	0.634	6
$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.553	7	0.631	7
$\text{DistilBERT}_{\text{BASE}}$ German cased	0.629	3	0.663	2
$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.589	5	0.642	5

Table 4.4: Results for Subtask C2 (aspect+sentiment) on synchronic (syn) and diachronic (dia) test datasets. The score refers to the micro-averaged F1 score. The best model 2017 is the system by the organizers Ruppert et al. (2017).

Here, the pre-trained models surpass the best model from 2017 by 15.7-25.9 percent points and 20.7-26.5 percent points, respectively, for the synchronic and diachronic test datasets. Again, the best model is the uncased German $\text{BERT}_{\text{BASE}}$ model reaching a score of 0.655 and 0.689, respectively. For both Subtask C1 and C2, all the displayed models perform better on the diachronic than on the synchronic test data.

It may be interesting to have a more detailed look at the model performance for this subtask because of the high number of classes and their skewed distribution by investigating the performance on category-level. Table 4.5 shows the performance of the uncased German $\text{BERT}_{\text{BASE}}$ model by DBMDZ per test dataset for Subtask C1. The support indicates the number of true labels which are also displayed in Table 2.7 in this case. Seven categories are summarized in *Rest* because they have an F1 score of 0 for both test datasets, i.e. the model is not able to correctly identify any of these seven aspects appearing in the test data. The table is sorted by the score on the synchronic test dataset.

	Test syn		Test dia	
Aspect Category	**Score**	**Support**	**Score**	**Support**
Allgemein	0.854	1,398	0.877	1,024
Sonstige Unregelmäßigkeiten	0.782	224	0.785	164
Connectivity	0.750	36	0.838	73
Zugfahrt	0.678	241	0.687	184
Auslastung und Platzangebot	0.645	35	0.667	20
Sicherheit	0.602	84	0.639	42
Atmosphäre	0.600	148	0.532	53
Barrierefreiheit	0.500	9	0	2
Ticketkauf	0.481	95	0.506	48
Service und Kundenbetreuung	0.476	63	0.417	27
DB App und Website	0.455	28	0.563	18
Informationen	0.329	58	0.464	35
Komfort und Ausstattung	0.286	24	0	11
Rest	*0*	*24*	*0*	*20*

Table 4.5: F1 score and support by aspect category (Subtask C1). Seven categories are summarized in *Rest* and show each a score of 0.

The scores for `Allgemein` (engl. general), `Sonstige Unregelmäßigkeiten` (engl. other irregularities) and `Connectivity` are the highest. 13 categories show a positive F1 score on at least one of the two test datasets, which are mostly similar between the two test datasets. By re-taking a look at Table 2.7, it can be deduced that the categories in *Rest* could not be assigned due to data sparsity problems. Therefore, the model was not able to learn how to correctly identify these categories.

The same issue is expected to occur in Subtask C2 because the relative distribution of the true labels is similar here, with the aspect+sentiment category `Allgemein:neutral` as majority class. Over 50% of the true labels belong to this class. Table 4.6 confirms the assumption and shows that only 12 out of 60 labels can be detected by the model (see Table 4.6).

	Test syn		Test dia	
Aspect+Sentiment Category	**Score**	**Support**	**Score**	**Support**
Allgemein:neutral	0.804	1,108	0.832	913
Sonstige Unregelmäßigkeiten:negative	0.782	221	0.793	159
Zugfahrt:negative	0.645	197	0.725	149
Sicherheit:negative	0.640	78	0.585	39
Allgemein:negative	0.582	258	0.333	80
Atmosphäre:negative	0.569	126	0.447	39
Connectivity:negative	0.400	20	0.291	46
Ticketkauf:negative	0.364	42	0.298	34
Auslastung und Platzangebot:negative	0.350	31	0.211	17
Allgemein:positive	0.214	41	0.690	33
Zugfahrt:positive	0.154	34	0	34
Service und Kundenbetreuung:negative	0.146	36	0.174	21
Rest	*0*	*343*	*0*	*180*

Table 4.6: F1 score and support by aspect+sentiment category (Subtask C2). 48 categories are summarized in *Rest* and show each a score of 0.

All the aspect categories displayed in Table 4.6 are also visible in Table 4.5 and most of them have negative sentiment. `Allgemein:neutral` and `Sonstige Unregelmäßigkeiten:negative` show the highest scores. Again, we assume that here, 48 categories could not be identified due to data sparsity. However, having this in mind, the model achieves a relatively high overall performance for both, Subtask C1 and C2 (see Table 4.3 and Table 4.4). This is mainly owed to the high score of the majority classes `Allgemein` and `Allgemein:neutral`, respectively, because the micro F1 score strongly weights these two labels. It might be interesting whether the classification of the rare categories can be improved by balancing the data. We experimented with removing general categories such as `Allgemein`, `Allgemein:neutral` or documents with sentiment `neutral` since these are usually less interesting for a company. These experiments are depicted in Appendix A. Here, one can observe a large drop in the overall score which is attributed to the absence of the strong majority class and the resulting data loss. Indeed, the classification for some single categories could be improved, but the rare categories could still not be identified by the language model.

4.4 Subtask D

Subtask D refers to the OTE identification and is thus a token classification task. As this is a rather difficult task, the organizers distinguish between exact (Subtask D1) and overlapping match (Subtask D2). The overlap tolerates a deviation of +/− one token. Here, "entities" are identified by their BIO-tags, i.e. one entity corresponds to at least one token tag starting with `B-` for "Beginning" and continuing with `I-` for "Inner". If a token does not belong to any entity, the tag `O` for "Outer" is assigned. For instance, the sequence "fährt nicht" (engl. "does not run") consists of two tokens and would receive the entity `Zugfahrt:negative` and the token tags [`B-Zugfahrt:negative I-Zugfahrt:negative`] if it refers to a DB train which is not running. It is noteworthy that there are less entities here than for Subtask C because document-level aspects or sentiments could not always be assigned to a certain sequence in the document. As a result, there are less documents at disposal for this task, namely 9,193. The remaining data has 1.86 opinions per document on average. The majority class is now `Sonstige Unregelmäßigkeiten:negative` with around 15.4% of the true entities (16,650 in total), leading to more balanced data than in Subtask C.

As already mentioned in Subsection 3.5.3, the pre-trained language models with softmax layer will be compared to the models with CRF loss. Table 4.7 shows the results for exact matching for each pre-trained model, divided by usage of CRF. Thus, 14 language models are present for Subtask D1.

		Test syn		Test dia	
Model type	**Language model**	**Score**	**Rank**	**Score**	**Rank**
	Best model 2017	0.229	15	0.301	15
BERT	$\text{BERT}_{\text{BASE}}$ German cased	0.460	7	0.455	6
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.480	3	0.466	3
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	0.492	2	0.501	2
	$\text{BERT}_{\text{BASE}}$ multilingual cased	0.447	8	0.457	5
	$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.429	12	0.404	12
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.347	14	0.357	14
	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.430	11	0.419	10
BERT+CRF	$\text{BERT}_{\text{BASE}}$ German cased	0.446	9	0.443	9
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.466	6	0.444	8
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	**0.515**	1	**0.518**	1
	$\text{BERT}_{\text{BASE}}$ multilingual cased	0.472	5	0.466	3
	$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.477	4	0.452	7
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.424	13	0.403	13
+CRF	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.436	10	0.418	11

Table 4.7: Results for Subtask D1 (exact match) on synchronic and diachronic test datasets. The score refers to the micro-averaged F1 score on entity level. The best model 2017 is the system by the organizers (Ruppert et al., 2017).

The best performing model is the uncased German $\text{BERT}_{\text{BASE}}$ model by DBMDZ with CRF layer on both test datasets, with a score of 0.515 and 0.518, respectively. Overall, the results from 2017 are outperformed by 11.8-28.6 percent points on the synchronic dataset and 5.6-21.7 percent points on the diachronic dataset.

Model type	Language model	Test syn		Test dia	
		Score	**Rank**	**Score**	**Rank**
	Best model 2017	0.348	15	0.365	15
BERT	$\text{BERT}_{\text{BASE}}$ German cased	0.471	7	0.474	4
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.491	3	0.488	3
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	0.501	2	0.518	2
	$\text{BERT}_{\text{BASE}}$ multilingual cased	0.457	8	0.473	6
	$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.435	11	0.417	13
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.397	14	0.407	14
	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.433	12	0.429	10
BERT+CRF	$\text{BERT}_{\text{BASE}}$ German cased	0.455	9	0.457	9
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.476	5	0.469	7
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	**0.523**	1	**0.533**	1
	$\text{BERT}_{\text{BASE}}$ multilingual cased	0.476	5	0.474	4
	$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.484	4	0.464	8
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.433	12	0.423	12
+CRF	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.442	10	0.427	11

Table 4.8: Results for Subtask D2 (overlapping match) on synchronic and diachronic test datasets. The score refers to the micro-averaged F1 score. The best model 2017 is the LSTM CRF stacked learner correct offsets by Lee et al. (2017) for the synchronic test dataset and the system by the organizers (Ruppert et al., 2017) for the diachronic test dataset.

As visible in Table 4.8, for the overlapping match, the models outperform the best system from 2017 by 4.9-17.5 percent points on the synchronic test dataset and by 4.2-16.8 percent points on the diachronic test dataset. The best pre-trained model is the same as for the exact match which achieves a score of 0.523 on the synchronic and 0.533 on the diachronic dataset.

Similar as for Subtask C, the results for the best model are investigated in more detail. Table 4.9 gives the detailed classification report for the uncased German $\text{BERT}_{\text{BASE}}$ model with CRF layer on Subtask D1. Only entities that were correctly detected at least once are displayed. The table is sorted by the score on the synchronic test dataset. The classification report for Subtask D2 is displayed analogously in Table 4.10.

	Test syn		Test dia	
Category	**Score**	**Support**	**Score**	**Support**
Zugfahrt:negative	0.702	622	0.729	495
Sonstige Unregelmäßigkeiten:negative	0.681	693	0.581	484
Sicherheit:negative	0.604	337	0.457	122
Connectivity:negative	0.598	56	0.620	109
Barrierefreiheit:negative	0.595	14	0	3
Auslastung und Platzangebot:negative	0.579	66	0.447	31
Connectivity:positive	0.571	26	0.555	60
Allgemein:negative	0.545	807	0.343	139
Atmosphäre:negative	0.500	403	0.337	164
Ticketkauf:negative	0.383	96	0.583	74
Ticketkauf:positive	0.368	59	0	13
Komfort und Ausstattung:negative	0.357	24	0	16
Atmosphäre:neutral	0.348	40	0.111	14
Service und Kundenbetreuung:negative	0.323	74	0.286	31
Informationen:negative	0.301	68	0.505	46
Zugfahrt:positive	0.276	62	0.343	83
DB App und Website:negative	0.232	39	0.375	33
DB App und Website:neutral	0.188	23	0	11
Sonstige Unregelmäßigkeiten:neutral	0.179	13	0.222	2
Allgemein:positive	0.157	86	0.586	92
Service und Kundenbetreuung:positive	0.115	23	0	5
Atmosphäre:positive	0.105	26	0	15
Ticketkauf:neutral	0.040	144	0.222	25
Connectivity:neutral	0	11	0.211	15
Toiletten:negative	0	15	0.160	23
Rest	*0*	*355*	*0*	*115*

Table 4.9: F1 score and support by aspect+sentiment entity with exact match (Subtask D1). 35 categories are summarized in *Rest* and show each a score of 0.

For Subtask D1, the model returns a positive score on 25 entity categories on at least one of the two test datasets. The category `Zugfahrt:negative` can be classified best on both test datasets, followed by `Sonstige Unregelmäßigkeiten:negative` and `Sicherheit:negative` for the synchronic test dataset and by `Connectivity:negative`

and `Allgemein:positive` for the diachronic dataset. Visibly, the scores between the two test datasets differ more here than in the classification report of the previous task.

	Test syn		Test dia	
Category	**Score**	**Support**	**Score**	**Support**
Zugfahrt:negative	0.708	622	0.739	495
Sonstige Unregelmäßigkeiten:negative	0.697	693	0.617	484
Sicherheit:negative	0.607	337	0.475	122
Connectivity:negative	0.598	56	0.620	109
Barrierefreiheit:negative	0.595	14	0	3
Auslastung und Platzangebot:negative	0.579	66	0.447	31
Connectivity:positive	0.571	26	0.555	60
Allgemein:negative	0.561	807	0.363	139
Atmosphäre:negative	0.505	403	0.358	164
Ticketkauf:negative	0.383	96	0.583	74
Ticketkauf:positive	0.368	59	0	13
Komfort und Ausstattung:negative	0.357	24	0	16
Atmosphäre:neutral	0.348	40	0.111	14
Service und Kundenbetreuung:negative	0.323	74	0.286	31
Informationen:negative	0.301	68	0.505	46
Zugfahrt:positive	0.276	62	0.343	83
DB App und Website:negative	0.261	39	0.406	33
DB App und Website:neutral	0.188	23	0	11
Sonstige Unregelmäßigkeiten:neutral	0.179	13	0.222	2
Allgemein:positive	0.157	86	0.586	92
Service und Kundenbetreuung:positive	0.115	23	0	5
Atmosphäre:positive	0.105	26	0	15
Ticketkauf:neutral	0.040	144	0.222	25
Connectivity:neutral	0	11	0.211	15
Toiletten:negative	0	15	0.160	23
Rest	*0*	*355*	*0*	*112*

Table 4.10: F1 score and support by aspect+sentiment entity with overlapping match (Subtask D2). 35 categories are summarized in *Rest* and show each a score of 0.

The report for the overlapping match (Table 4.10) shows slightly better results on some categories than for the exact match. The third-best score on the diachronic test

data is now `Sonstige Unregelmäßigkeiten:negative`. Besides this, the top three categories per test dataset remain the same.

Apart from the fact that this is a different kind of task than before, one can notice that even though the overall micro F1 scores are lower for Subtask D than for Subtask C, the model manages to successfully identify a larger variety of categories, i.e. it achieves a positive score for more categories. This is probably due to the more balanced data for Subtask D than for Subtask C2, resulting in a lower overall score and mostly higher category-level scores. Thus, it is not surprising that, compared to the experiments on Subtask C2 (see Appendix A), the best model reaches similar overall scores for Subtask D. Interestingly, the best model achieves a positive score on a larger variety of categories on Subtask D than on the balance experiments on Subtask C2 presented in Appendix A.

Chapter 5

Discussion and Outlook

As expected, all the pre-trained language models clearly outperform all the models from 2017, proving once more the power of transfer learning. Throughout the presented analyses, the models always achieve similar results between the synchronic and the diachronic test datasets, which indicates temporal robustness for the models. Nonetheless, the diachronic dataset was collected only half a year after the main data. It would be interesting to see whether the trained models would return similar predictions on data collected a couple of years later. If the models would then achieve similar results as for the synchronic test dataset, it would prove robustness.

The uncased German $\text{BERT}_{\text{BASE}}$ model by DBMDZ achieves the best results across all subtasks. One may have already expected that a German BERT model ranks best across the regarded language models.
As stated in Subsection 3.5.4.3, DistilBERT retains 97% of BERT's model understandings. Thus, DistilBERT was expected to achieve lower scores. It is, however, more efficient and for the document-level tasks, mostly a competitive alternative to BERT. For instance, the model reaches the podium for Subtask C2, namely the second rank for the diachronic data and the third rank for the synchronic test data, although for Subtask D, it performed worst among the pre-trained language models. Besides, it was already proven that monolingual BERT models often outperform the multilingual models for a variety of tasks (Rönnqvist et al., 2019). The uncased multilingual $\text{BERT}_{\text{BASE}}$ model was trained on 102 languages, and its cased version was trained on 104 languages. Especially when words are tokenized into small parts, one may assume that the multilingual models have more difficulties making sense of the individual tokens. It is hence not recommended re-using a multilingual pre-trained model on the GermEval 2017 Task. On document-level, the three multilingual models performed worst. An exception is the

uncased multilingual $\text{BERT}_{\text{BASE}}$ model on Subtask A attaining the fourth rank on both test datasets.
It may not seem evident that an uncased language model results as the best performing model although in sentiment analysis, capitalized letters may be a polarity indicator. In addition, since nouns and beginnings of sentences always start with a capital letter in German, one may assume that lower-casing the whole text may change the meaning of many words and thus confuse the language model. Nevertheless, the GermEval 2017 documents are very noisy since they were retrieved from social media. That means that the data contains many misspellings, grammar and expression mistakes, dialect, and colloquial language. The examples in Section 2.2, which were taken from the GermEval 2017 data, already give an impression. For this reason, some participating teams pursued an elaborate pre-processing on the text data in order to eliminate some noise (Hövelmann and Friedrich, 2017; Sayyed et al., 2017; Sidarenka, 2017). Among other things, Hövelmann and Friedrich (2017) transformed the text to lower-case and replaced, for example, "S-Bahn" and "S Bahn" with "sbahn". We suppose that in this case, lower-casing the texts improves the data quality by taking some noise out. It acts as a sort of regularization and as a result, the model generalizes better than the cased models. The findings from Mayhew et al. (2019) corroborate this hypothesis. Here, the authors compared cased with uncased pre-trained language models on social media data for an NER task.

Specifically for Subtask D, a CRF layer was compared to the usual softmax layer. Indeed, the CRF classifier outperforms the results of the models using softmax. This is not surprising when considering the benefits stated in Subsection 3.5.3. The models with a CRF layer surpassed the results of the models with a softmax layer in nine out of 14 cases (seven language models per two test datasets) by 0.6-7.7 percent points for Subtask D1 and by 0.1-4.9 percent points for Subtask D2. We would hence prefer the CRF layer although it slows down the fine-tuning and inference of the model.

Similar to the results from 2017, the scores get worse as the task progresses. This was foreseeable since the difficulty of the task increases while the data shrinks. While in Subtask A and B there are only two to three classes, a multilabel classification with 20 and 60 labels, respectively, is conducted within Subtask C. Surprisingly, the overall results for Subtask C1 are almost as high as for Subtask B since the high overall scores in Subtask C1 are mainly owed to the strong majority class `Allgemein` (see Section 4.3). For Subtask D, the same 60 labels are present and in addition, the models try to assign the correct target

sequence. Further, the data examples which could not be assigned any target expression get lost. When taking this into account, the models achieve highly competitive results. Nevertheless, we took a deep look at the documents which were wrongly classified by the models. The results for Subtask C and D were already interpreted in Section 4.3 and 4.4, claiming that wrong annotations result mainly from data sparsity and skewness issues. Basically, the wrongly classified documents in Subtask A refer to the DB, but did not contain any feedback about the company, or they did contain feedback, but not about the DB. Moreover, the models seem to have difficulties with classifying long texts in which DB is mentioned besides several other topics. Regarding Subtask B, one can observe that the models struggle with the correct document-level sentiment annotation in the presence of multiple polarities. A common problem - even for humans - is the correct polarity classification in presence of irony. In some single cases, we do not even agree with the annotated label for both subtasks. For instance, the following original document from the diachronic test dataset was annotated as relevant, but all the models classified it as irrelevant.

```
Gemütliche Altbauwohnung mit Charakter WG-Zimmer: Gemütliche
Altbauwohnung mit Charakter 01.01.2017 Grabenweg 4, 5600 Lenzburg Ich
(24) suche eine/n Wochenaufenthalter/in, der/die mit mir meine 3.5
Zimmerwohnung im gemütlichen Lenzburg teilt. Das zu vermietende
Zimmer ist 16.5m2 gross
```

Translation: "Cozy old building apartment with character: cozy old building apartment with character 01.01.2017 Grabenweg 4, 5600 Lenzburg. I (24) am looking for a weekly resident to share my 3.5 rooms apartment in cozy Lenzburg. The room for rent is 16.5 square meters in size"

This example should be annotated as irrelevant since it does not contain any feedback about DB.

Even though the models reach competitive results, they can still be improved. One technical limitation we faced was a lack of memory space. For this reason, the possible hyperparameter choices were limited. Devlin et al. (2019) state that the hyperparameter choice can make a remarkable difference when the training dataset size is less than 100,000. Since there are far less data examples at disposal, it is highly probable that experimenting with other hyperparameters, especially with a different maximum sequence length or batch size, or hyperparameter tuning would improve the results noticeably. Thus, some experiments on Subtask D were conducted to investigate the impact of the

hyperparameter choice as long as the memory space allowed us to. Appendix B sketches the results of the language models for a maximum sequence length of 512 and a batch size of 8 and 16, respectively. However, running the latter setting was not possible for the multilingual $\text{BERT}_{\text{BASE}}$ models. These findings were compared to the results from Section 4.4 where a maximum sequence length of 256 and a batch size of 32 were considered. Herewith, it is verified that the hyperparameter choice has a substantial impact on the model performance. The models achieve the best results with a maximum sequence length of 512 and a batch size of 8.

Nonetheless, the problems concerning data sparsity and skewness remain and are harder to resolve. Theoretically, one solution is to collect and annotate more data, but in practice, this is highly time-consuming and thus usually not realizable. Moreover, it could even increase the imbalance in the data as the web crawling technique does not specifically watch out for scarce aspect categories. An idea would be to consider sampling techniques as Sayyed et al. (2017) did. However, if a company is really interested in gaining more feedback on the rarely addressed aspects, a specific questioning could be an option.

It is expected that NLP will continue to grow in the future, hoping that more language models will be available, especially for languages other than English. For instance, advanced pre-trained language models such as ALBERT or RoBERTa are still not available for German text data. Hopefully, the research on German documents gains popularity so that the need for superior German pre-trained language models increases. As a result, German companies could benefit from this by being able to obtain a more reliable pattern on how their product or service is perceived among the customers.

Chapter 6

Conclusion

In the scope of the present thesis, the GermEval 2017 Shared Task on "Aspect-based Sentiment in Social Media Customer Feedback" was re-analyzed using Transformer-based pre-trained language models. Five different BERT and two DistilBERT models, which are suitable for German documents, were considered. All the models surpass the scores achieved in 2017 and show robust results between the synchronic and the diachronic test datasets. The best performing model is clearly the uncased German $\text{BERT}_{\text{BASE}}$ model by DBMDZ, advancing the state-of-the-art on all four subtasks. We would thus recommend considering this model on German sentiment analysis tasks on social media data. The cased German $\text{DistilBERT}_{\text{BASE}}$ model which was distilled from the cased German $\text{BERT}_{\text{BASE}}$ model by DBMDZ is faster than the $\text{BERT}_{\text{BASE}}$ models and represents a proper alternative to the BERT models.

For Subtask A, the best model achieves a micro-averaged F1 score of 0.957 on the synchronic data and 0.948 on the diachronic test dataset, surpassing the previous results by 5.4 and 4.2 percent points. For Subtask B, the best model surpasses the former results by 4.0 and 5.0 percent points, respectively, resulting in a score of 0.807 and 0.800. Furthermore, the best model reaches a high score of 0.761 and 0.791 on Subtask C1, which outperforms the former results by 22.4 and 23.5 percent points. For Subtask C2, the model reaches scores up to 0.655 and 0.689, and hence outperforms the results from 2017 by 25.9 and 26.5 percent points. The best model for Subtask D includes a CRF instead of the usual softmax layer. Concerning Subtask D1, it achieves a score of 0.515 and 0.518, respectively, outperforming the former scores by 28.6 and 21.7 percent points. The scores on Subtask D1 were however improved using another hyperparameter setting by 4.4 and 3.9 percent points, respectively. With a maximum sequence length of 512 tokens and a batch size of 8, the model thus reaches a high score of 0.556 on

synchronic test data and 0.541 on diachronic dataset, outperforming the results from 2017 by a total of 32.7 and 24.0 percent points. For Subtask D2, the models score up to 0.523 and 0.533. The best model thus surpasses the results from 2017 by 17.5 and 16.8 percent points, respectively. These results were also outperformed using a maximum sequence length of 512 and a batch size of 8, achieving a score of 0.569 and 0.561, respectively. The model hereby surpasses the scores from 2017 by 22.1 and 19.6 percent points.

As a result, an option to further improve the performance would be to consider other hyperparameter combinations as long as the available memory space suffices. Nevertheless, despite the clearly higher scores in comparison to the previous results, the models still struggle with data sparsity and imbalances, irony, and unintuitive annotations. Finally, it would be interesting to investigate whether future pre-trained language models, which are also suitable for German, will outperform these results.

Bibliography

Arkhipov, M., Trofimova, M., Kuratov, Y. and Sorokin, A. (2019). Tuning Multilingual Transformers for Language-Specific Named Entity Recognition, *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, Association for Computational Linguistics, Florence, Italy, pp. 89–93.

Bahdanau, D., Cho, K. and Bengio, Y. (2016). Neural Machine Translation by Jointly Learning to Align and Translate, *3rd International Conference on Learning Representations, ICLR 2015*.

Baroni, M., Dinu, G. and Kruszewski, G. (2014). Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors, *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Baltimore, Maryland, pp. 238–247.

Biesialska, K., Biesialska, M. and Rybinski, H. (2020). Sentiment Analysis with Contextual Embeddings and Self-Attention.

Bojanowski, P., Grave, E., Joulin, A. and Mikolov, T. (2017). Enriching Word Vectors with Subword Information, *Transactions of the Association for Computational Linguistics* **5**: 135–146.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I. and Amodei, D. (2020). Language Models are Few-Shot Learners.

Bucila, C., Caruana, R. and Niculescu-Mizil, A. (2006). Model Compression, *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data*

Mining (KDD 2006), Association for Computing Machinery, New York, USA, pp. 535–541.

Chelba, C., Mikolov, T., Schuster, M., Ge, Q., Brants, T., Koehn, P. and Robinson, T. (2014). One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling, *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, pp. 1724–1734.

Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Association for Computational Linguistics, Minneapolis, Minnesota, pp. 4171–4186.

Elman, J. L. (1990). Finding Structure in Time, *Cognitive Science* **14**(2): 179–211.

Esplà-Gomis, M., Forcada, M., Ramírez-Sánchez, G. and Hoang, H. T. (2019). ParaCrawl: Web-scale parallel corpora for the languages of the EU, *MTSummit*.

Gage, P. (1994). A New Algorithm for Data Compression, **12**(2): 23–38.

Gokaslan, A. and Cohen, V. (2019). OpenWebText Corpus.
URL: *http://Skylion007.github.io/OpenWebTextCorpus*

Goldberg, Y. (2017). *Neural Network Methods for Natural Language Processing*, Vol. #37 of *Synthesis Lectures on Human Language Technologies*, Morgan & Claypool, San Rafael, California.

Guhr, O., Schumann, A.-K., Bahrmann, F. and Böhme, H.-J. (2020). Training a Broad-Coverage German Sentiment Classification Model for Dialog Systems, *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, Marseille, France, pp. 1627–1632.

Haddow, B. (2018). News Crawl Corpus.
URL: *http://metashare.dfki.de/repository/browse/wmt-2018-news-crawl/2f7d7c8c1d5211e8ba4e842b2b6a04d7d9aab8315e5a44e887fab203ffefc3ec/*

Harris, Z. S. (1954). Distributional Structure, *WORD* **10**(2-3): 146–162.

Hinton, G., Vinyals, O. and Dean, J. (2015). Distilling the Knowledge in a Neural Network.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-term Memory, *Neural computation* **9**(8): 1735–1780.

Huang, Z., Xu, W. and Yu, K. (2015). Bidirectional LSTM-CRF Models for Sequence Tagging.

Hövelmann, L. and Friedrich, C. M. (2017). Fasttext and Gradient Boosted Trees at GermEval-2017 Tasks on Relevance Classification and Document-level Polarity, *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*, Berlin, Germany.

Joulin, A., Grave, E., Bojanowski, P. and Mikolov, T. (2017). Bag of Tricks for Efficient Text Classification, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, Association for Computational Linguistics, Valencia, Spain, pp. 427–431.

Lafferty, J. D., McCallum, A. and Pereira, F. C. N. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data, *Proceedings of the 18th International Conference on Machine Learning (ICML 2001)*, Morgan Kaufmann Publishers Inc., San Francisco, California, USA, pp. 282–289.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K. and Dyer, C. (2016). Neural Architectures for Named Entity Recognition.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P. and Soricut, R. (2020). ALBERT: A Lite BERT for Self-supervised Learning of Language Representations, *ICLR 2020* .

Lee, J.-U., Eger, S., Daxenberger, J. and Gurevych, I. (2017). UKP TU-DA at GermEval 2017: Deep Learning for Aspect Based Sentiment Detection, *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*, Berlin, Germany.

Levy, O., Goldberg, Y. and Dagan, I. (2015). Improving Distributional Similarity with Lessons Learned from Word Embeddings, *Transactions of the Association for Computational Linguistics* **3**: 211–225.

Li, X., Bing, L., Li, P. and Lam, W. (2019a). A Unified Model for Opinion Target Extraction and Target Sentiment Prediction.

Li, X., Bing, L., Zhang, W. and Lam, W. (2019b). Exploiting BERT for End-to-End Aspect-based Sentiment Analysis, *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pp. 34–41.

Lison, P. and Tiedemann, J. (2016). OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles, *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach, *ICLR 2020* .

Manning, C. D., Raghavan, P. and Schütze, H. (2008). *Introduction to Information Retrieval*, Cambridge University Press, Cambridge, UK.

Mao, J. and Liu, W. (2019). Hadoken: a BERT-CRF Model for Medical Document Anonymization, *IberLEF@SEPLN*, Bilbao, Spain.

Mayhew, S., Tsygankova, T. and Roth, D. (2019). ner and pos when nothing is capitalized, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processingand the 9th International Joint Conference on Natural Language Processing*, Association for Computational Linguistics, Hong Kong, China, pp. 6256–6261.

Mikolov, T., Chen, K., Corrado, G. and Dean, J. (2013a). Efficient Estimation of Word Representations in Vector Space, *ICLR (Workshop Poster)*, pp. 1–12.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. and Dean, J. (2013b). Distributed Representations of Words and Phrases and their Compositionality, *Advances in Neural Information Processing Systems (NIPS 2013)*, Vol. 26, Curran Associates, Inc., pp. 3111–3119.

Mishra, P., Mujadia, V. and Lanka, S. (2017). GermEval 2017: Sequence based Models for Customer Feedback Analysis, *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*, Berlin, Germany.

Naderalvojoud, B., Qasemizadeh, B. and Kallmeyer, L. (2017). HU-HHU at GermEval-2017 Sub-task B: Lexicon-Based Deep Learning for Contextual Sentiment Analysis, *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*, Berlin, Germany.

Nagel, S. (2016). CC-News.
URL: *http://web.archive.org/save/http://commoncrawl.org/2016/10/newsdataset-available*

Ortiz Suárez, P. J., Sagot, B. and Romary, L. (2019). Asynchronous Pipeline for Processing Huge Corpora on Medium to Low Resource Infrastructures, *in* P. Bański, A. Barbaresi, H. Biber, E. Breiteneder, S. Clematide, M. Kupietz, H. Lüngen and C. Iliadi (eds), *7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7)*, Leibniz-Institut für Deutsche Sprache, Cardiff, United Kingdom.

Ostendorff, M., Blume, T. and Ostendorff, S. (2020). Towards an Open Platform for Legal Information, *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*, JCDL '20, Association for Computing Machinery, New York, NY, USA, pp. 385—-388.

Pennington, J., Socher, R. and Manning, C. (2014). GloVe: Global Vectors for Word Representation, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, pp. 1532–1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. and Zettlemoyer, L. (2018). Deep Contextualized Word Representations, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Association for Computational Linguistics, New Orleans, Louisiana, pp. 2227–2237.

Radford, A., Narasimhan, K., Salimans, T. and Sutskever, I. (2018). Improving Language Understanding by Generative Pre-Training.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. and Sutskever, I. (2019). Language Models Are Unsupervised Multitask Learners.

Raganato, A. and Tiedemann, J. (2018). An Analysis of Encoder Representations in Transformer-Based Machine Translation, *Proceedings of the 2018 EMNLP Workshop*

BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, Association for Computational Linguistics, Brussels, Belgium, pp. 287–297.

Rajpurkar, P., Jia, R. and Liang, P. (2018). Know What You Don't Know: Unanswerable Questions for SQuAD, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Association for Computational Linguistics, Melbourne, Australia, pp. 784–789.

Rajpurkar, P., Zhang, J., Lopyrev, K. and Liang, P. (2016). SQuAD: 100,000+ Questions for Machine Comprehension of Text, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Austin, Texas, pp. 2383–2392.

Reimers, N. and Gurevych, I. (2017). Optimal Hyperparameters for Deep LSTM-Networks for Sequence Labeling Tasks.

Remus, R., Quasthoff, U. and Heyer, G. (2010). SentiWS - A Publicly Available German-language Resource for Sentiment Analysis, *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2010)*, European Language Resources Association (ELRA), Valletta, Malta, pp. 1168–1171.

Rönnqvist, S., Kanerva, J., Salakoski, T. and Ginter, F. (2019). Is Multilingual BERT Fluent in Language Generation?, *Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing*, Linköping University Electronic Press, Turku, Finland, pp. 29–36.

Ruppert, E., Kumar, A. and Biemann, C. (2017). LT-ABSA: An Extensible Open-Source System for Document-Level and Aspect-Based Sentiment Analysis, *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*, Berlin, Germany.

Sanh, V., Debut, L., Chaumond, J. and Wolf, T. (2020). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, *EMC2: 5th Edition Co-located with NeurIPS'19* .

Sayyed, Z. A., Dakota, D. and Kübler, S. (2017). IDS-IUCL: Investigating Feature Selection and Oversampling for GermEval 2017, *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*, Berlin, Germany.

Schulz, K., Mieskes, M. and Becker, C. (2017). h-da Participation at GermEval Subtask B: Document-level Polarity, *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*, Berlin, Germany.

Sennrich, R., Haddow, B. and Birch, A. (2016). Neural Machine Translation of Rare Words with Subword Units, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Berlin, Germany, pp. 1715–1725.

Sidarenka, U. (2017). PotTS at GermEval-2017 Task B: Document-Level Polarity Detection Using Hand-Crafted SVM and Deep Bidirectional LSTM Network, *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*, Berlin, Germany.

Skadiņš, R., Tiedemann, J., Rozis, R. and Deksne, D. (2014). Billions of Parallel Words for Free: Building and Using the EU Bookshop Corpus, *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC 2014)*, European Language Resources Association (ELRA), Reykjavik, Iceland, pp. 1850–1855.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. and Potts, C. (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Seattle, Washington, USA, pp. 1631–1642.

Souza, F., Nogueira, R. and Lotufo, R. (2020). Portuguese Named Entity Recognition using BERT-CRF.

Trautmann, D., Daxenberger, J., Stab, C., Schütze, H. and Gurevych, I. (2020). Fine-Grained Argument Unit Recognition and Classification, *Proceedings of the AAAI Conference on Artificial Intelligence* **34**(05): 9048–9056.

Trinh, T. H. and Le, Q. V. (2019). A Simple Method for Commonsense Reasoning.

Van Rossum, G. and Drake, F. L. (2009). *Python 3 Reference Manual*, CreateSpace, Scotts Valley, California, USA.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. and Polosukhin, I. (2017). Attention Is All You Need, *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, California, USA.

Viterbi, A. (1967). Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm, *IEEE Transactions on Information Theory* **13**(2): 260–269.

Waltinger, U. (2010). Sentiment Analysis Reloaded - A Comparative Study on Sentiment Polarity Identification Combining Machine Learning and Subjectivity Features, *Proceedings of the 6th International Conference on Web Information Systems and Technologies (WEBIST 2010)*, Vol. 1, INSTICC Press, Valencia, Spain, pp. 203–210.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O. and Bowman, S. R. (2019). GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding, *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Association for Computational Linguistics, Brussels, Belgium, pp. 353–355.

Wang, Y., Sun, Y., Ma, Z., Gao, L., Xu, Y. and Sun, T. (2020). Application of Pre-training Models in Named Entity Recognition, *2020 12th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, IEEE.

Wojatzki, M., Ruppert, E., Holschneider, S., Zesch, T. and Biemann, C. (2017). GermEval 2017: Shared Task on Aspect-based Sentiment in Social Media Customer Feedback, *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*, Berlin, Germany, pp. 1–12.
URL: *https://sites.google.com/view/germeval2017-absa/*

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q. and Rush, A. M. (2020). Transformers: State-of-the-Art Natural Language Processing, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Association for Computational Linguistics, Online, pp. 38–45.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Łukasz Kaiser, Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M. and Dean, J. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation.

Zhu, Y., Kiros, R., Zemel, R. S., Salakhutdinov, R., Urtasun, R., Torralba, A. and Fidler, S. (2015). Aligning Books and Movies: Towards Story-like Visual Explanations by

Watching Movies and Reading Books, *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV 2015)*, IEEE Computer Society, Santiago, Chile, pp. 19–27.

Appendix A

Experiments on Subtask C2

In order to improve the results for less frequent classes, we experimented with excluding "less interesting" categories, which are `Allgemein` (engl. general), `neutral` and `Allgemein:neutral` and fine-tuned the language models on the remaining documents and labels. Since these categories are majority classes, leaving them out leads to more balanced data. The following presents the experimental results for the uncased German $\text{BERT}_{\text{BASE}}$ model on Subtask C2. Table A.1 displays the results after excluding the category `Allgemein`. Table A.2 shows the results after deleting the sentiment `neutral` and Table A.3 refers to the results after excluding `Allgemein:neutral`. The tables are sorted by the F1 score on the synchronic test dataset. The experiments were conducted using the same hyperparameter setting as stated in Chapter 4.

	Test syn		Test dia	
Aspect+Sentiment Category	**Score**	**Support**	**Score**	**Support**
Sonstige Unregelmäßigkeiten:negative	0.778	221	0.775	159
Zugfahrt:negative	0.689	197	0.701	149
Connectivity:negative	0.645	20	0.476	46
Sicherheit:negative	0.615	78	0.576	39
Atmosphäre:negative	0.464	126	0.405	39
Service und Kundenbetreuung:negative	0.233	36	0	21
Auslastung und Platzangebot:negative	0.176	31	0.111	17
Ticketkauf:negative	0.154	42	0.190	34
Zugfahrt:positive	0.054	34	0	34
Rest	*0*	*343*	*0*	*180*

Table A.1: F1 score and support by aspect+sentiment category (Subtask C2) without category `Allgemein`. 48 categories are summarized in *Rest* and show each a score of 0. The overall micro-averaged F1 score is 0.481 on synchronic test data and 0.501 on diachronic test data.

After deleting the category `Allgemein`, the model achieves a noticeably lower overall micro F1 score on both test datasets (0.481 and 0.501) as it uses much less and more balanced data. The model reaches positive scores for nine aspect+sentiment categories. The best results are attained for `Sonstige Unregelmäßigkeiten:negative` with scores above 0.77. Moreover, the model achieves positive results for the same categories as with the category `Allgemein`. Compared to Table 4.6, the score increases for `Zugfahrt:negative` and `Connectivity:negative` on both test datasets, and for `Service und Kundenbetreuung:negative` (engl. service and customer support) on the synchronic test datasets. The remaining categories show a lower score.

	Test syn		Test dia	
Aspect+Sentiment Category	**Score**	**Support**	**Score**	**Support**
Sonstige Unregelmäßigkeiten:negative	0.791	221	0.789	159
Zugfahrt:negative	0.684	197	0.732	149
Sicherheit:negative	0.602	78	0.571	39
Allgemein:negative	0.589	258	0.400	80
Connectivity:negative	0.588	20	0.639	46
Atmosphäre:negative	0.545	126	0.458	39
Auslastung und Platzangebot:negative	0.455	31	0.455	17
Ticketkauf:negative	0.436	42	0.407	34
Service und Kundenbetreuung:negative	0.415	36	0.231	21
Zugfahrt:positive	0.279	34	0.383	34
Service und Kundenbetreuung:positive	0.250	14	0	4
Allgemein:positive	0.241	41	0.696	33
Informationen:negative	0.129	29	0.143	26
Rest	*0*	*123*	*0.133*	*105*

Table A.2: F1 score and support by aspect+sentiment category (Subtask C2) without sentiment `neutral`. 27 categories are summarized in *Rest* and show each a score of 0. The overall micro-averaged F1 score is 0.568 on synchronic test data and 0.562 on diachronic test data.

Similar findings can be observed when looking at the results after excluding the sentiment `neutral`. The overall micro F1 scores, namely 0.568 and 0.562, are again much lower than for the full model (see Table 4.4). The class the model can classify best is again `Sonstige Unregelmäßigkeiten:negative` which achieves a result of 0.791 on the synchronic test dataset and 0.789 on diachronic test data. All in all, the model shows positive scores for 13 out of 40 categories. Here, compared to Table 4.6, it achieves positive results for `Service und Kundenbetreuung:positive` and `Informationen:negative` (engl. information). Furthermore, the results improve for `Zugfahrt:negative`, `Allgemein:negative`, `Auslastung und Platzangebot:negative`, `Ticketkauf:negative`, `Allgemein:positive` and `Zugfahrt:positive`. We assume that the higher scores for these aspect+sentiment categories are also related to the fact that the differentiation between the polarities `positive` and `negative` is sharper than between `positive`, `neutral` and `negative`.

	Test syn		Test dia	
Aspect+Sentiment Category	**Score**	**Support**	**Score**	**Support**
Sonstige Unregelmäßigkeiten:negative	0.783	221	0.800	159
Zugfahrt:negative	0.693	197	0.720	149
Sicherheit:negative	0.608	78	0.615	39
Allgemein:negative	0.574	258	0.317	80
Atmosphäre:negative	0.558	126	0.429	39
Connectivity:negative	0.552	20	0.551	46
Ticketkauf:negative	0.471	42	0.318	34
Allgemein:positive	0.230	41	0.542	33
Auslastung und Platzangebot:negative	0.222	31	0.200	17
Service und Kundenbetreuung:negative	0.054	36	0	21
Rest	*0*	*377*	*0*	*214*

Table A.3: F1 score and support by aspect+sentiment category (Subtask C2) without `Allgemein:neutral`. 49 categories are summarized in *Rest* and show each a score of 0. The overall micro-averaged F1 score is 0.504 on synchronic test data and 0.504 on diachronic test data.

When deleting only the class `Allgemein:neutral`, the model shows again much lower results for the overall micro F1 scores, namely 0.504 on both datasets, compared to the full model. It returns positive results for the ten aspect+sentiment categories already observed within the full model. The score increments for `Zugfahrt:negative` and `Allgemein:positive` on synchronic test data, and for `Connectivity:negative` and `Ticketkauf:negative` on both test datasets.

In summary, it can be observed that balancing the data by previously excluding "less interesting" categories helps improving the results for some single categories. Nonetheless, the model shows a maximum overall score of 0.568 and 0.562, respectively, and scarcer categories can still not be classified as the data sparsity problem remains.

Appendix B

Hyperparameter Experiments on Subtask D

For Subtask D, our memory space allowed us to run the language models using a maximum sequence length of 512 with a batch size of 8 and 16, respectively. Yet, the latter could not be run for the multilingual $\text{BERT}_{\text{BASE}}$ models. Tables B.1 and B.2 display the results of the hyperparameter experiments on Subtask D1. The remaining hyperparameters were set as described in the beginning of Section 4.

		Test syn		Test dia	
Model type	**Language model**	**Score**	**Rank**	**Score**	**Rank**
	Best model 2017	0.229	15	0.301	15
BERT	$\text{BERT}_{\text{BASE}}$ German cased	0.489	9	0.465	10
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.524	3	0.496	3
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	0.549	2	0.534	2
	$\text{BERT}_{\text{BASE}}$ multilingual cased	0.496	7	0.481	6
	$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.502	6	0.490	5
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.415	14	0.412	14
	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.467	12	0.475	9
BERT+CRF	$\text{BERT}_{\text{BASE}}$ German cased	0.483	10	0.462	11
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.519	4	0.481	6
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	**0.556**	1	**0.541**	1
	$\text{BERT}_{\text{BASE}}$ multilingual cased	0.508	5	0.478	8
	$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.494	8	0.492	4
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.459	13	0.437	13
+CRF	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.468	11	0.459	12

Table B.1: Results for Subtask D1 (exact match) on synchronic and diachronic test datasets. The language models were fine-tuned with a maximum sequence length of 512 and a batch size of 8. The score refers to the F1 score on entity-level. The best model 2017 is the system by the organizers (Ruppert et al., 2017).

Model type	Language model	Test syn		Test dia	
		Score	**Rank**	**Score**	**Rank**
	Best model 2017	0.229	11	0.301	11
BERT	$\text{BERT}_{\text{BASE}}$ German cased	0.479	5	0.452	4
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.492	4	0.450	5
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	0.524	2	0.520	2
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.381	10	0.396	10
	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.448	8	0.429	8
BERT+CRF	$\text{BERT}_{\text{BASE}}$ German cased	0.462	6	0.431	7
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.514	3	0.478	3
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	**0.539**	1	**0.532**	1
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.428	9	0.406	9
+CRF	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.456	7	0.436	6

Table B.2: Results for Subtask D1 (exact match) on synchronic and diachronic test datasets. The language models were fine-tuned with a maximum sequence length of 512 and a batch size of 16. The score refers to the F1 score on entity level. The best model 2017 is the system by the organizers (Ruppert et al., 2017).

For the exact match, within the same language model, the model always achieves higher performance values with a maximum sequence length of 512 and a batch size of 8 (Table B.1) than with a batch size of 16 (Table B.2), and also compared to a maximum sequence length of 256 with a batch size of 32 (Table 4.7). Thus, the overall best performing model is the uncased German $\text{BERT}_{\text{BASE}}$ by DBMDZ with CRF loss, a maximum sequence length of 512 and a batch size of 8, reaching a score of 0.556 on the synchronic test data and 0.541 on the diachronic test data. These scores outperform the results from 2017 by 32.7 and 24.0 percent points, respectively.

		Test syn		Test dia	
Model type	**Language model**	**Score**	**Rank**	**Score**	**Rank**
	Best model 2017	0.348	15	0.365	15
BERT	$\text{BERT}_{\text{BASE}}$ German cased	0.502	7	0.485	9
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.537	3	0.518	3
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	0.558	2	0.554	2
	$\text{BERT}_{\text{BASE}}$ multilingual cased	0.501	8	0.491	7
	$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.512	6	0.504	5
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.427	14	0.428	14
	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.452	13	0.460	12
BERT+CRF	$\text{BERT}_{\text{BASE}}$ German cased	0.496	10	0.480	10
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.530	4	0.502	6
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	**0.569**	1	**0.561**	1
	$\text{BERT}_{\text{BASE}}$ multilingual cased	0.515	5	0.490	8
	$\text{BERT}_{\text{BASE}}$ multilingual uncased	0.501	9	0.505	4
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.470	12	0.451	13
+CRF	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.475	11	0.473	11

Table B.3: Results for Subtask D2 (overlapping match) on synchronic and diachronic test datasets. The language models were fine-tuned with a maximum sequence length of 512 and a batch size of 8. The score refers to the F1 score on entity level. The best model 2017 is the LSTM CRF stacked learner correct offsets by Lee et al. (2017) for the synchronic test dataset and the system by the organizers (Ruppert et al., 2017) for the diachronic test dataset.

Model type	Language model	Test syn		Test dia	
		Score	**Rank**	**Score**	**Rank**
	Best model 2017	0.348	11	0.365	11
BERT	$\text{BERT}_{\text{BASE}}$ German cased	0.491	5	0.468	5
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.505	4	0.472	4
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	0.537	2	0.535	2
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.389	10	0.411	10
	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.455	8	0.436	8
BERT+CRF	$\text{BERT}_{\text{BASE}}$ German cased	0.473	6	0.447	6
	$\text{BERT}_{\text{BASE}}$ German DBMDZ cased	0.526	3	0.501	3
	$\text{BERT}_{\text{BASE}}$ German DBMDZ uncased	**0.552**	1	**0.547**	1
DistilBERT	$\text{DistilBERT}_{\text{BASE}}$ German cased	0.437	9	0.428	9
+CRF	$\text{DistilBERT}_{\text{BASE}}$ multilingual cased	0.462	7	0.442	7

Table B.4: Results for Subtask D2 (overlapping match) on synchronic and diachronic test datasets. The language models were fine-tuned with a maximum sequence length of 512 and a batch size of 16. The score refers to the F1 score on entity-level. The best model 2017 is the LSTM CRF stacked learner correct offsets by Lee et al. (2017) for the synchronic test dataset and the system by the organizers (Ruppert et al., 2017) for the diachronic test dataset.

The same occurs with the overlap match, as can be observed when comparing Table B.3 with Table 4.8 and Table B.4. The overall best performing model is again the uncased German $\text{BERT}_{\text{BASE}}$ by DBMDZ with CRF loss, a maximum sequence length of 512 and a batch size of 8, reaching a score of 0.569 on the synchronic test data and 0.561 on diachronic test data. Here, the model outperforms the results from 2017 by 22.1 and 19.6 percent points, respectively.

These experiments confirm the statement by Devlin et al. (2019) which says that the hyperparameter setting can have a decisive impact on the performance when training on less than 100,000 data examples, as it is the case in Subtask D. Here, we would recommend using a maximum sequence length of 512 tokens instead of 256 if possible as this has noticeably improved the overall performance for all the pre-trained language model. This is not surprising in a token classification task. Interestingly, a batch size of 16 instead of 8 results in worse performance when keeping the maximum sequence length at 512.

Appendix C

Electronic Annex

The enclosed electronic storage device contains the following files:

- the present master's thesis as PDF file,
- the source code written on Python, version 3.8.7 64-bit (Van Rossum and Drake, 2009), and Visual Studio Code, version 1.52.1,
- the original GermEval 2017 data in XML and the pre-processed data in TSV format,
- the result tables as Excel files.

The source code is also available in a repository on GitHub: https://github.com/ac74/masterthesis_germeval2017. It includes a README file explaining how to run the codes and which Python module versions were utilized.

Statutory Declaration

I herewith declare that I have composed the present thesis myself and without use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such. Other references regarding the statement and scope are indicated by full details of the publications concerned. The thesis in the same or similar form has not been submitted to any examination body and has not been published. This thesis was not, even in part, used in another examination or as a course performance. Furthermore, I declare that the submitted written bound copies of the present thesis and the version submitted on a data carrier are consistent with each other in contents.

Munich, January 4, 2021

(Alessandra Corvonato)