University of Zurich[UZH]

University of Zurich
Faculty of Business, Economics and Informatics
Department of Informatics

Sentence-like Segmentation of Swiss German Audio Transcripts for Dependency Parsing

Melvin S. Steiger
13-762-117

Submitted in part fulfilment of the requirements for the degree of
Master of Science in Informatics
01.04.2023

Examiner: Prof. Dr. Martin Volk
Supervisor: Dr. Tanja Samardžić

Abstract

Dependency parsers tend to struggle with parsing transcribed spoken language as they are trained on properly structured, written text. Spoken language lacks the structure of properly written text and exhibits typical phenomena like disfluency, repetition, and truncation of words and sentences. This research examines the problem of parsing spoken language for Swiss German audio transcripts from ArchiMob corpus. Swiss German, an umbrella term for the German (Alemannic) dialects spoken in Switzerland, lacks orthographic and grammatical standardization, shows a high degree of variation among the various dialects and differs substantially from Standard German. The lack of standardization is due to the situation of diglossia in Switzerland. As Swiss German is mainly an oral language or restricted to informal writing, many resources lack structure and exhibit a high variability in terms of morphology, spelling and vocabulary. The combination of variation in Swiss German, its lack of standardization and the unstructuredness of spoken language render parsing transcribed Swiss German challenging. Accordingly, pre-trained (German) dependency parsers struggle with Swiss German audio transcripts and little data is available to train them.

This research tackles the problem of parsing spoken language by re-segmenting Swiss German audio transcripts into sentence-like units (SLUs) and examines the impact of re-segmentation on dependency parser performance. Therefore, our experiment setup includes two evaluation steps, one for re-segmentation and one for dependency parsing. We frame the re-segmentation as a binary classification task aiming to predict tokens marking an SLU-boundary. For this purpose, we fine-tune a pre-trained German Bert model to predict such boundaries. These predicted SLU-boundaries are used to re-shape the input for the dependency parser. We show that the re-segmentation into SLUs leads to an improvement of the Labeled Attachment Score (LAS) over a baseline. Moreover, we demonstrate that the performance in the SLU-boundary classification task correlates with the parser performance. To engage in such a supervised learning setting, a test set composed out of roughly 200 SLUs was manually created and annotated with dependency labels for the two folded evaluation. With our work, we contribute to processing spoken Swiss German by showing a way of inducing more structure.

Acknowledgements

I would like to express my gratitude to my thesis advisor, Dr. Tanja Samardžić, for her continuous support, her patience and dedication. I thank the Department of Computational Linguistics from the University of Zurich for access to infrastructure and the possibility to write this thesis under the supervision of Prof. Dr. Martin Volk. Finally, I'd like to express my gratitude to my parents without whom none of this would have been possible.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Dependency parsing, one variant of syntactical parsing, refers to the process of assigning a head to each token of a sentence and labeling the dependency relation with its head according to a predefined dependency grammar. It is a common step in NLP-processing together with POS-tagging, tokenization and others. The syntactical information from dependency parsing helps to identify semantic and syntactic relations between words (tokens), and provides a base that can be used in other NLP tasks like Named Entity Recognition (NER), Co-reference Resolution, Information Retrieval and more. Dependency parsers are trained on properly structured, written text and therefore tend to encounter problems if input does not exhibit the same structure. Accordingly, dependency parsing is challenging for transcribed spoken language. Spoken language lacks the structure of properly written text and exhibits typical phenomena like disfluency, repetition, and truncation of words and sentences.

In the following, this research examines the effect of re-segmenting unstructured audio transcripts into sentence-like units (SLU) on the performance of dependency parsing. This is motivated by the fact that SLUs are closer to properly written text than transcribed spoken language. A pre-trained neural network is fine-tuned in a token classification setting to identify SLU-boundaries. The predicted SLU-boundaries are then used to re-shape the input for the dependency parser. Besides investigating the impact on parser performance, we examine the connection between performance in SLU-boundary-detection and improvement in parser per-

formance. The experiment setup, therefore, includes two evaluation steps, one for the token classification and one for dependency parsing. For the experiments, Universal Dependencies (UD, de Marneffe et al. (2021)) annotation, a pre-trained UDPipe (Straka et al., 2016) parser and a uncased German Bert model (Devlin et al., 2019) are used. We performed this examination for transcribed spoken Swiss German coming from ArchiMob corpus (Scherrer et al., 2019). ArchiMob consists of transcribed interviews held with Swiss native speakers born between 1910 and 1930. This data is in highly unstructured form, does neither contain SLUs nor dependency annotation, and exhibits a high degree of disfluency, repetition and truncation. Accordingly, no test data is available for ArchiMob. Therefore, another important contribution is the manual creation of a two folded test set composed of roughly 200 SLUs in order to evaluate the SLU-boundary detection and the parser performance. To the author's best knowledge, the following work is the first to tackle SLU-boundary-detection for Swiss German audio transcripts. Thus, we contribute to parsing and processing spoken Swiss German with our approach.

Swiss German is a low-resource language (or dialect) that, even if not the official language of Switzerland, is the variety of the German language most commonly spoken in Switzerland by roughly 60% of the population (Bundesamt für Statistik, 2022b). Swiss German dialects, the German variants spoken across Switzerland, are the daily language of choice in speaking and writing, but standard Swiss German is the language used in official and formal settings. Little data is available in Swiss German, though this began to change in recent years. This data, however, exists in a highly unstructured form; on the one hand, Swiss German itself exhibits a lot of variation in vocabulary, morphology, and syntax and also differs strongly from standard German, and, on the other hand, Swiss German lacks standardization. As Swiss German is mainly an oral language, there is especially a demand for systems processing spoken Swiss German for other NLP tasks (NER, Machine Translation[1], automatic subtitle generation etc.). As mentioned, this research contributes to processing transcribed spoken Swiss German and shows successfully a way of inducing more structure into the transcripts.

[1]Usually Machine translation is performed with data properly structured into sentences

1.1 Outline

Chapter 2 provides the necessary background on Swiss German, sentence segmentation in spoken language and dependency parsing. The linguistic situation in Switzerland and differences among dialects and with standard German are presented and exemplified. Furthermore, the chapter provides references to and summaries of related work for all three background topics. Following this, Chapter 3 introduces the methodology in detail. We argue that fine-tuning a Bert model is a promising approach to detect SLU-boundaries. Fine-tuning a model is a supervised task. Accordingly, we elaborate on the requirements for data, discuss the creation of the manual test set in detail and introduce suitable training data. The deep-dive into the process of manual segmentation and dependency annotation exemplifies the complexity of the task further. Chapter 3 also introduces the data processing workflow and the statistics for the data used in the experiments. Chapter 4, then, guides through the experiment setup, specifies technical details of the libraries, models and training settings; and elaborates on the two evaluation steps. The chapter ends with a detailed presentation of the results and shows the impact of re-segmentation on parser performance. Chapter 5, finally, summarizes the achievement of this thesis and introduces possibilities for future work.

Chapter 2

Background and Literature Overview

For our approach to re-segment spoken Swiss German into SLUs and to study the impact of this re-segmentation on dependency parsing performance, background for Swiss German, sentence segmentation in spoken language and dependency parsing is required. Section 2.1 provides context of the linguistic situation in Switzerland and introduces distinctive. The related work section elaborates on approaches for dependency parsing, POS-tagging and mentions recent developments in machine translation. More importantly, the ArchiMob corpus, the data this thesis aims to re-segment, is introduced, together with SwissDial, another Swiss German text corpus.

A short overview of sentence segmentation is provided in Section 2.2 followed by a brief description of difficulties when working with transcripts of spoken language. Furthermore, the two articles most similar to the approach undertaken in the experiments section are discussed in detail. Lastly, Section 2.3 presents dependency parsing and annotation tag sets.

2.1 Swiss German

The linguistic situation in Switzerland is rather complicated. Although the country has four official languages, German, French, Italian and Romansh; none of these is the language spoken by most people in daily life. 62,3% reported German as one of their main languages in 2020

(Bundesamt für Statistik, 2022b). The fact that most of them will not speak German in their daily life is due to the phenomenon of diglossia (see Wyler and Siebenhaar (1997)). Diglossia means that two languages or dialects are used by the same language group restricted to different situations. This leads to the first important distinction in terminology, while Swiss German (Schwiizerdütsch[1]) refers to the German dialects spoken in Switzerland, Swiss Standard German (Schweizer Hochdeutsch) is the term for the standardized variant of the German language of Switzerland, which is the official language of Switzerland. Throughout the following text, *Swiss* will often be used to refer to German speaking part of Switzerland and does not refer to the country as a whole. While Swiss Standard German and Standard German (from Germany) are mutually intelligible, Swiss German dialects tend not to be understood by speakers of other German variants. Differences in both Standard German variants are rather small and concern mainly vocabulary, orthography, pronunciation, and to a small extent syntax, and morphology. Such differences are called Helvetisms. Common orthographic examples are the spelling of Umlaut (*Ae* (CH) vs. *Ä* (DE)) at the beginning of (geographical) names or the absence of the letter *ß* in Swiss Standard German. An example from vocabulary is the verb to park which is *parkieren* in Switzerland and *parken* in Germany. Swiss Standard German is the language used in written contexts, the education system, news and television, politics and other official contexts. Swiss German, however, is the default language in daily life and informal situations. With the rise of the internet and smartphones Swiss German is, especially by younger generations, also the written language of choice in daily life. There is, of course, also music and literature in Swiss German .

Linguistically, (almost) all Swiss German dialects belong to the Alemannic dialect group marking the southern end of the German dialect continuum. The dialects are further subdivided into Low, High and Highest Alemannic with most Swiss dialects belonging to High Alemannic. Most native Swiss German speaker tend to understand (almost) all dialects. Although historically the different dialects corresponded roughly to the different regions of Switzerland and the dialects are still named after their region, given the mobility and technology of modern times, Swiss German native speakers grow up exposed to a variety of dialects. The Swiss German

[1]If not specified differently, spelling of Swiss German in this section follows the individual writing style of the author.

dialects differ between them in pronunciation, vocabulary, morphology and syntax. For example, the word coffee has a male gender for some dialects but is neuter for others. Other typical examples are the word order for complex or composed predicates. *To let go* is rendered by some speakers as *gah lah* and as *lah gah* by others while in Standard German the only correct way is *gehen lassen* (all verbs are in infinitive). Similarly the past in Swiss German of *had (has had)* is formed by some speakers with *gha hät* and *hät gha* by others and will also vary depending on the context. Hollenstein and Aepli (2014) presented a good example showing differences in dialects around Bern and Zurich, depicted in Figure 2.1. Besides demonstrating the different word order, it also gives a few more examples of different vocabulary. *Because* has three different forms *weil - wil - wüu*, or *him* has two forms *ihn - ne*.

Swiss German completely lacks standardization of any kind. "Language standardization is traditionally defined as the process that creates and promotes a consistent norm for language usage and typically involves the reduction or loss of optional variability."Ayres-Bennett and Bellamy (2021) For Swiss German, this means that there is no norm for grammar (morphology and syntax), orthography, and pronunciation. Attempts to provide a system to represent phonetic differences across the dialects have been made. The most famous one is the Dieth-Schreibung (Dieth and Schmid-Cadalbert, 1986). However, this system was never really popular and never integrated into daily usage as it requires a lot of linguistic knowledge to write correctly and writing Swiss German is never taught in Switzerland. Nowadays, most people tend to have their own way of writing which is influenced by the provenance of the speaker and personal taste. Typical examples are the rendering of long vowels, e.g., *ii* or *ie*; *aa*, *ah* or *a*, or whether to represent a certain sound with *ä* or *e*. This phenomenon complicates the situation for Swiss German. Although there is more text written in Swiss German (SMS, Mails, Social Media) orthography can differ significantly from dialect to dialect and speaker to speaker. Thus, dealing with standardizing writing is always an important step when working with Swiss German data. Finally, Swiss German also exhibits differences with Standard German. As there are differences between dialects, the following facts are not necessarily true for all dialects. There are, of course, phonetic differences like the complete absence of *ç*, the voiceless palatal fricative. Figure 2.1

Dialect around Bern:	Si **het** ne **la ga**, wüu er ne gnue Gäu **het gha**, **für** es Billet **z'**löse.
Dialect around Zurich:	Si **hät** ihn **gah lah**, wil er nöd gnueg Gäld **gha het**, **zum** es Billet löse.
Standard German:	Sie **liess** ihn **gehen**, weil er nicht genug Geld **hatte**, **um** ein Billet **zu** kaufen.
English:	She **let** him **go** because he **did** not **have** enough money **to** buy a ticket.

Figure 2.1: Differences between Dialects and Standard German (Hollenstein and Aepli, 2014, 88)

displays some differences in word order. A very characteristic difference is the usage of clitic pronouns in Swiss German. For example, *we have* is *mir hä**nd*** but if the word order is inverted for a question it will become *hä**mmer***, while in Standard German it is *haben wir*. This also demonstrates that Swiss German exhibits Sandhi (see Moulton (1986)). It is common in Swiss German that words appear in different forms depending on emphasis and their surrounding words. For example, *I* could be *i* or *ich*[2]. If *ich* is placed after the relative pronoun *wo* (literally where but used as a general relative pronoun) it can merge with it forming *wo**n**i*, another example would be *I have* which is *ich han* but *hani* when inverted. Another peculiarity is that the subject pronoun tends to be dropped for certain verb forms, in Standard German the subject pronoun is mandatory and can never be dropped outside of very colloquial daily speech. *You have* is *häsch* in Swiss German but *du hast* in Standard German. These two phenomena also appear combined; *did you do it?* is realized as *häsch s gmacht*. If we compare that to Standard German *Hast **du es** gemacht*, we see that the *du* is dropped completely and the *es* becomes a clitic. It also serves to illustrate the difficulty of writing Swiss German; for a native speaker it does not matter if one writes *häsch**s***, *häsch s*, *häsch's*, or *häsch äs/häsch es* (with non clitic pronoun); and the probability is high that the same speaker would write the same form differently. Furthermore, indefinite articles tend to merge with prepositions in Swiss German, which in Standard German happens only with the definite article. *at a place* is *amä Platz* in Swiss German and *an einem Platz* in Standard German, showing clearly that Swiss German merges *an* and *einem* which also leads to phonetic changes.

Swiss German uses some words differently than Standard German. The verbs *tun* (to do) and *gehen* (to go) can be used as auxiliary verbs. *Gah* (gehen) for example is often used together with another verb to convey the meaning of going to; *tue* (tun) can be used to express an

[2]Variants used by the author

imperative instead of forming the proper imperative among other uses. In order to is expressed with *um ... zu* in Standard German and as *zum* or *für* in Swiss German.

From a more high level perspective, Swiss German lacks a bunch of tenses that are present in Standard German, namely the preterite (Präteritum) and the future tenses (Future 1 and 2). As a consequence, Swiss German does not have a true preterite perfect (Plusquamperfekt), but has a reduplicated form of the perfect instead: *The train had parted* would be *Dä zug isch ab**g**fahre **g**sii* in Swiss German with two past participles but *Der Zug **war** ab**ge**fahren* in Standard German (*g* or *ge* usually marks a past participle). Usage of the reduplicated perfect is not consistent across dialects and speakers. Swiss German also does not know the genitive case (except in some fixed collocations) and uses constructions with dative instead. Furthermore, nominative and accusative are no longer distinct for nouns in Swiss German, they are, as for English, still different for (personal) pronouns. More information about syntactical peculiarities of Swiss German and how to treat them when annotating can be found in the article of Scherrer (2011).

These examples illustrate why normalizing Swiss German is important. In the context of Swiss German, normalization refers to the process of mapping variation in writing, pronunciation and morphology arising from inter-speaker and inter-dialectal differences to one lexical unit (see Scherrer et al. (2019), Honnet et al. (2018), and Section 2.1.2). Normalization is usually used in the context of some form of data processing, while standardization is rather tied to a high-level (linguistic) view of a language. Without normalization - besides having a lot of variation - alignment of Swiss German with Standard German (or any other language) is utterly difficult. Alignment means mapping sentences and tokens from one language to their counterpart from a second language, thus creating parallel data.[3] Such corpora of aligned sentences between languages are called parallel corpora. Aligned Swiss German - Standard German text does, however, normally not match in terms of token number and word order.

[3]Usually exact token-to-token alignment is not possible.

2.1.1 Swiss German - Related Work

NOAH's corpus[4] (73'616 tokens) was manually annotated with POS-tags and is composed out of articles from the Alemannic Wikipedia, some newspaper articles written in Swiss German, and parts of the Swiss German dialect version of the official annual report of Swatch. The corpus lacks normalization and thus exhibits a variety of different spellings. Hollenstein and Aepli (2014) introduced NOAH's corpus and tackled Part-Of-Speech-tagging (POS-tagging) for Swiss German. They trained various state-of-the part POS-taggers (e.g. TreeTagger, Wapiti CRF Tagger) with the corpus and evaluated them over the corpus with 10-fold cross validation. The best result was achieved by a BTagger scoring an accuracy of 90.62%. Their work builds on the Stuttgart-Tubingen-TagSet (STTS) with some modification for Swiss German peculiarities, e.g., the indefinite article that merges with prepositions.

Aepli and Clematide (2018) (see also Aepli (2018)) examined two approaches to Swiss German dependency parsing. As a solid statistical dependency parser needs a lot of training data - which do not exist for Swiss German - they opted for "cross-lingual parsing strategies (...), making use of Standard German resources."(Aepli and Clematide, 2018, 6) In order to do so, they needed a corpus of parallel sentences Standard German - Swiss German. Leveraging crowd-sourcing, Swiss German sentences from NOAH's corpus and from Swiss German literature were translated to Standard German. Doing so, they created a corpus consisting of 26'015 parallel sentences. Aepli and Clematide (2018) engaged in a *model transfer* approach where a dependency parser is trained on Standard German data only passing the POS-tags as input and later applied to Swiss German by providing the POS-tags of the Swiss German input. A second, more complicated approach is *annotation projection*. "The parse of the Standard German translation is projected along the word alignment to its Swiss German correspondent. The input consists of the Standard German parse and the alignment between the Standard German sentence and its Swiss German version." Aepli and Clematide (2018) This approach needs word-alignment and has to take into account cases where a one-to-one token alignment is not possible. They scored a Labelled Attachment Score (LAS) of 60% with the transfer model approach evaluated on a manually curated test set composed of 100 sentences.

[4]Available here: `https://github.com/noe-eva/NOAH-Corpus`[28.02.2023]

Especially in recent years, a lot of research was conducted revolving around spoken Swiss German or translating Swiss German. Hufe and Avramidis (2022) trained a sequence-to-sequence model trying to translate Swiss German sign-language to Standard German text. To represent the sign-language, they used 3D-augmentation of body key points. Lambrecht et al. (2022) engaged in a machine translation task from Standard German to Alemannic dialects (which includes Swiss German) leveraging data from the Alemannic Wikipedia[5]. They tried a multi-dialectal approach, meaning translating with one model from Standard German to different Alemannic dialects. They showed that "using back-translation, a significant gain of +4.5 over the strong transformer baseline of 37.3 BLEU points is accomplished" (Lambrecht et al., 2022, 129). Furthermore, they concluded that "[d]ifferentiating between several Alemannic dialects instead of treating Alemannic as one dialect leads to substantial improvements" (Lambrecht et al., 2022, 129) enhancing the aforementioned gain of 4.5 to "7.5 to 10.6 BLEU points over the baseline depending on the dialect".

2.1.2 ArchiMob

As stated in the introduction, we present an approach to re-segment Swiss German into SLUs. This is performed on Swiss German data from ArchiMob corpus, which is presented in the following.

ArchiMob[6] corpus, as the authors put it, "is a result of a long design process, intensive manual work and specially adapted computational processing"(Scherrer et al., 2019, 735) The corpus contains just one part of the broader Archimob project[7] that was founded in 1998. The creation started in 2004 "when a collection of 52 VHS tapes was obtained from the Archimob association"(Scherrer et al., 2019, 738). Archimob stands for *Archives de la mobilisation* (Mobilisation archive) as the tapes contain interviews held with native Swiss German speakers narrating their

[5] `https://als.wikipedia.org/wiki/Wikipedia:Houptsyte`[28.02.2023]
[6] Following Scherrer et al. (2019) the corpus is called ArchiMob while the project is labeled Archimob.
[7] Available at: urlhttp://www.archimob.ch/[06.02.2023]

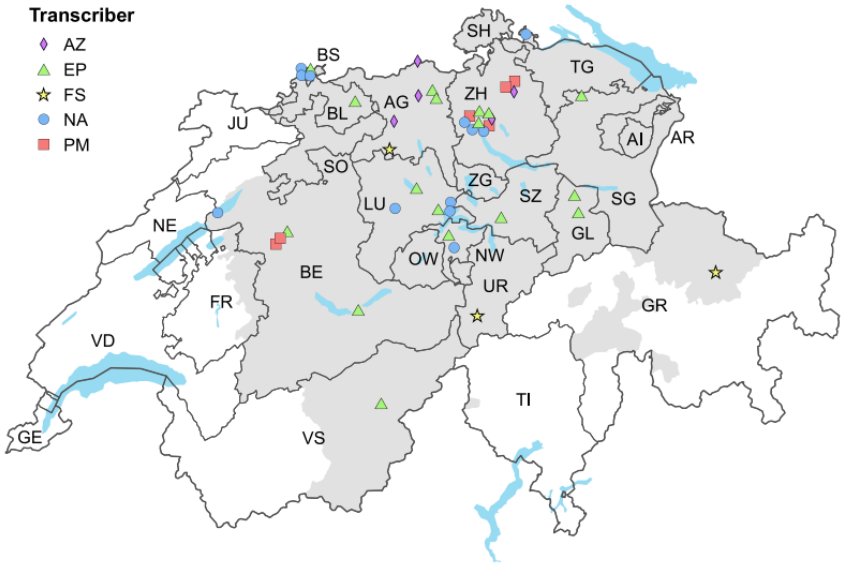

Figure 2.2: Locations of the ArchiMob Recordings (Scherrer et al., 2019, 738)

childhood memories of Switzerland before and during the Second World War[8]. Out of this 52 recordings, 43 were used to create the corpus while nine had to be excluded due to poor quality or linguistic reasons. The origins of the speakers of these 43 recordings is shown in Figure 2.2. Each symbol on the map indicates the origin of a speaker, while the shape and the color of the symbols represent the different transcribers. The grey area, furthermore, shows the German speaking areas of Switzerland. The audios were manually transcribed, accordingly ArchiMob does not only contain the audio files but also the manual transcripts. The corpus is provided in an XML-format[9] consisting of content files (the transcripts in XML), media files (alignment transcripts to audio), and speaker files (socio-demographic speaker information). A content file contains not only the transcribed text but encodes further information.

These content files contain the transcripts which are segmented into *utterances*. An utterance corresponds to transcription unit of an approximate average length of 4-8 seconds. The XML-representation of an utterance consists of the sequence of tokens (words) spoken during this time frame. Each token is rendered as an XML-element containing a normalized version of the token, a POS-tag, and the token itself (token is element, rest are attributes). For POS-tagging the ap-

[8]As Switzerland has only a defense army (Verteidigungsarmee) mobilization is only ordered if Switzerland needs to defend its sovereignty which happened for the last time during the Second World war.

[9]Available at: `https://www.spur.uzh.ch/en/departments/research/textgroup/ArchiMob.html`[27.02.2023]

proach of Hollenstein and Aepli (2014) was adapted. Furthermore, utterances contain "pauses (vocalised or not), repeated speech, and unclear (or untranscribable) passages"(Scherrer et al., 2019, 742), which are also encoded as XML-elements. Each utterance has a media reference pointing to the audio file and a reference to the speaker. As it is common for transcripts, there is neither punctuation in ArchiMob, nor do the utterances correspond to SLUs (or proper sentences).

As elaborated before, Swiss German has no consistent orthography. In ArchiMob we find: "First, dialectal variation causes lexical units to be pronounced, and therefore also written, in a different way in different regions. Second, a lexical unit that can be considered phonetically invariant (within a region) is written in a different way on different occasions (...)"(Scherrer et al., 2019, 744) Therefore, normalization is a key step in processing Swiss German "[i]n order to establish lexical identity of all writing variants that can be identified as 'the same word'"(Scherrer et al., 2019, 744). For ArchiMob, detailed normalization guidelines were established and, based on them, training data normalized by humans was created. This data was used to train a character-level statistical machine translation (CSMT) system which automatically normalized the rest of the corpus (Samardžić et al. (2015)). Normalization in ArchiMob does not only concern orthography[10]. The mentioned clitic pronouns were mapped to several tokens during the normalization. For example, *would we have it*: "hettemers is normalised as hätten wir es (...)".(Scherrer et al., 2019, 745) "Swiss German word forms that do not have etymologically related standard German counterparts are normalised using a reconstructed common Swiss German form. For example, *öpper* 'someone' is normalised as *etwer* instead of the semantic standard German equivalent *jemand* (...)".(Scherrer et al., 2019, 745) ArchiMob's normalization makes Swiss German more similar to written Standard German on the world level. The article itself provides more details on the normalization process.

ArchiMob consists of spontaneous, one-sided interviews and therefore the transcripts are more unstructured than properly written Swiss German. Besides already mentioned problems

[10]Also different transcribers might induce variation in writing.

arising from having Swiss German text with all its variation, the interviewees for ArchiMob were born between 1910 and 1930 and thus all roughly between 70 or 90 years old when giving the interviews. This is important as age (among other factors) has been shown to affect speech fluency (Leeper and Culatta (1995), Andrade and Martins (2010)). This leads to speech disruptions and a decrease in speech rate. Ageing might affect speech in other aspects as well. There are indications "that maintaining coherence in speech becomes more challenging as people age"(Hoffman et al., 2018, 1) Furthermore, repetition is more common in speech of elderly people. All this aspects together make the ArchiMob data highly unstructured and difficult to process. Although normalization standardizes orthography (and vocabulary) to some degree, the unstructuredness that is a consequence of spoken language remains part of the transcripts. The ArchiMob transcripts are full of repetitions, interrupted sentences followed by a reformulation attempt, incomplete sentences and sometimes the meaning is even hard to access for native speakers.

In summary, ArchiMob, given its nature, is probably more unstructured than other Swiss German resources. Normalization reduced the unstructuredness to some degree but much of it remains. Besides mentioned aspects of elderly speech, it is important to be aware that the provided utterance structure does not correspond to SLUs or sentences. One SLU might be composed out of several utterances; one utterance might be a single SLU or contain several SLUs. Or in the words of the authors: "The utterances are mostly fragments spanning over one or more sentence constituents. We do not mark sentence boundaries."(Scherrer et al., 2019, 743)

All these factors make ArchiMob a highly unstructured corpus in dire need of re-segmentation to induce structure and render the resource available for further processing.

2.1.3 SwissDial

SwissDial is another resource for Swiss German data besides the two already mentioned corpora ArchiMob and NOAH's corpus. Dogan-Schönberger et al. (2021) introduced, as they put it, "[t]he first annotated parallel corpus of spoken Swiss German across 8 major dialects

(...)."(Dogan-Schönberger et al., 2021) Although it is by far not the first parallel corpus for Swiss German and Standard German, it is indeed the first parallel corpus that aligns eight Swiss German dialects and Standard German. Not every Standard German sentence is aligned with all eight dialects, some are only mapped with a single dialect or a subset out of these eight dialects. They do not report the statistics of the corpus explicitly but only a table showing number of sentences per topic and dialect. This table claims that the corpus contains a total of 23'195 sentences. Their reported numbers seem, however, not to match with the corpus retrieved from the official download link[11][12]. The downloaded corpus contains 11'212 sentences in Standard German labeled with an ID. SwissDial was created by selecting Standard German sentences from different domains like medicine, sports, economics, Swiss politics and other which were manually translated to eight different Swiss German dialects. Annotators for each dialect were asked to translate the sentences into their native dialect and, in a second step, to read them aloud while being recorded. The corpus, thus, not only consists of aligned text but also includes recordings.

2.2 Sentence Segmentation in Spoken Language

Before diving deeper into sentence segmentation in spoken language, some general considerations concerning sentence segmentation might be a good starting point. Sentence Segmentation is a sub-task of text segmentation, and, even though it might seem a little contradictory, as spoken language is usually rendered as text before further processing, sentence segmentation in spoken language could be considered a sub-task as well.

Pak and Teh (2018) provide an overview over different approaches in text segmentation. They summarize 50 scientific papers covering a period of ten years, from January 2007 to January 2017. They report that 11% of the examined papers engage in sentence segmentation and language-wise only 2% perform some kind of text segmentation task for German. The most frequent task with 47% is segmentation into single words, and not surprisingly at all English

[11] Available at: `https://mtc.ethz.ch/publications/open-source/swiss-dial.html`[06.02.2023]

[12] For example, they state that the corpus contains 2'749 sentences from the dialect of Graubünden but the downloaded corpus contains 10'475 sentences labeled as this dialect.

with 38% is the most studied language, closely followed by Chinese with 33%. This yields the insight, that sentence segmentation is not really a prominent task. Furthermore, sentence segmentation is often considered a solved problem. However, this is not true at all and related to the fact that most sentence-segmentation algorithms (and papers) work with properly structured text. If the data in question is properly formatted and contains punctuation, the task of sentence segmentation is reduced to punctuation disambiguation, i.e., it remains only to decide if a full stops marks the end of a sentence or not. An example of such an established system is the Punkt Sentence Tokenizer from nltk[13]. It is a unsupervised algorithm that builds a model for abbreviation words, collocations, and words that start sentences. An example of a CRF-based (conditional random fields) sentence segmentation algorithm that also leverages punctuation for German can be found in the article from Sugisaki (2018). Another interesting read might be the article of Evang et al. (2013),where they engaged in a sequence labeling task simultaneously for tokenization and sentence segmentation also using punctuation.

In this paper, however, we aim to perform sentence segmentation for highly unstructured transcripts of spoken language that lack punctuation and are not properly structured according to grammar. This matter is further complicated by the peculiarities of Swiss German elaborated in the previous section and the nature of ArchiMob data (elderly speech). Segmenting spoken language transcripts also poses difficulties to human annotators. The paper of Stevenson and Gaizauskas (2000) "explores the problem of identifying sentence boundaries in the transcriptions produced by automatic speech recognition systems. An experiment which determines the level of human performance for this task is described (...)"(Stevenson and Gaizauskas, 2000, 84). The experiments were conducted for English with six different human annotators. They stripped the data from punctuation, presented different versions (upper cased, lower cased and mixed) to the humans and ask them to perform sentence boundary detection. The data consists of automatic transcripts from BBC's Nine O'Clock News and is therefore rather structured for spoken language when compared to spontaneous daily speech. The human annotated sentence boundaries were evaluated against the original form with punctuation. In terms of F1-

[13] `https://www.nltk.org/api/nltk.tokenize.punkt.html`[27.02.2023]

score, the best human achieved 94% and the lowest score ranged at 79%[14]. This paper clearly demonstrates that human annotators do not re-segment the transcripts identically and that no annotator managed to reproduce the original segmentation, which of course does not mean that the human segmentation is not valid. Different, valid segmentations are always possible. It does, however, prove that segmenting spoken language transcripts is a difficult task, even for humans. The next section elaborates in detail on two highly interesting articles that engage in the same task, SLU-detection in spoken language transcripts.

2.2.1 Sentence Segmentation in Spoken Language - Related Work

Zribi et al. (2016) "study the problem of detecting sentence boundary in transcribed spoken Tunisian Arabic." (Zribi et al., 2016, 323) As for most languages, also transcribed Tunisian Arabic lacks punctuation and spontaneous speech is often ill-formed according to standard grammar. Tunisian Arabic also differs in several aspects from modern standard Arabic, and thus also a low-resource language. Such differences concern the syntactical, lexical, morphological and phonological level. Similar to Swiss German, Tunisian Arabic is not codified nor standardized nor the official language of Tunisia. It is, however, considered to be the mother tongue of most Tunisians and the language, respectively dialectal varieties of it, spoken in daily life. Another, interesting feature of Tunisian Arabic is code switching. Not only did European languages influence the lexical level of Tunisian Arabic but it is still common today to include words or whole expression from French in the middle of a Tunisian Arabic sentence.

In general, sentence boundary detection "is a challenging task for Arabic language that is characterized by the absence of capital letters and the boundaries of sentences are not generally marked with punctuation marks." (Zribi et al., 2016, 325) Often, sentence boundaries are marked with conjunctions or other lexical expression, but these same words do also have other functions not marking a boundary. This already difficult task is further complicated by characteristics of Tunisian Arabic mentioned above, the lack of punctuation in the transcripts, and by phenomena related to spoken Tunisian Arabic like incomplete sentences, disfluency,

[14]They did not report *kappa*, refer to the article for further explanation

	Recall	Precision	F-measure
Baseline	40.98	82.45	54.75
CR	68.31	90.841	77.98
PART	**72.5**	**94.8**	**82.1**
Hyb1	72.42	89.15	79.92
Hyb2	66.00	73.91	69.73

Figure 2.3: Evaluation of Sentence Boundary Detection (Zribi et al., 2016, 329)

truncated word, repetitions and other. Zribi et al. (2016) present three different approaches to tackle described problem working with a corpus of transcribed Tunisian Arabic consisting of 42'388 words. The corpus is already segmented into sentences and a sentence was considered to constitute "a semantically meaningful unit"(Zribi et al., 2016, 326). The corpus was divided into a training set (32,012 words and 6,133 sentences), a development set (3,175 words and 440 sentences), and a test set (7,201 words and 1,215 sentences).

Firstly, they presented a rule based approach leveraging lexical markers like conjunctions and two prosodic features (silent and filled pauses). They created a set of 23 rules. Secondly, they engaged in a statistical approach based on a partial decision tree algorithm (PART). For this purpose, the tokens were labeled *B-S* (Beginning of sentence), *I-S* (Inside of sentence), *E-S* (End of sentence), and *S* (Single word sentence). Such algorithms rely on predefined features, "[they] have used two simple prosodic features that are silent and filled pauses. In the design of our features, [they] rely on linguistic features like adverbs, adjectives, verbs, etc."(Zribi et al., 2016, 328) Based on these features, the algorithm creates rules that serve as the decision points. Lastly, they experimented with two (three in theory, but only two evaluated) different hybrid approaches, combining aspects from the rule-based and the statistical approach. Table 2.3 displays the results of the evaluation on the test set, CR stands for the rule-based approach, PART for the statistical one (partial decision tree), and finally the two reported hybrid approaches. As a baseline, they used STAr, a system for sentence boundary detection in modern standard Arabic based on a set of contextual rules.

Following the evaluation of sentence boundary detection, they examined the effect of the seg-

		Ripper	PART	SVM
NoSeg		62.53	**71.88**	61.87
HandSeg		63.92	70.55	63.02
AutSeg	**PART**	61.69	66.58	61.04
	CR	**64.84**	70.65	63.04
	Hyb1	64.20	70.21	63.39
	Hyb2	63.92	68.22	**63.66**

Figure 2.4: Accuracy of the POS-tagger Evaluation (Zribi et al., 2016, 330)

mentation on a POS-tagger. For this purpose, they trained a POS-tagger with three different methods on different segmentations of the corpus. For the training method, they chose a statistical approach (SVM), and two rule-based methods (Ripper and PART). As an input format, they used no segmentation at all (NoSeg), the manually segmented corpus (HandSeg), and the sentence segmentation based on the four approaches presented (AutSeg). For the reported results, ten-fold-cross-validation was used. Although utterly interesting, Zribi et al. (2016) fail to interpret their results properly. Firstly, it is not clear on which data they evaluated the tagger. Secondly, they state "We remark that the SBD [sentence boundary detection] system helps the TA [Tunisian Arabic] POS tagger to improve its accuracy. We note that SVM and RIPPER performed better when the SBD system detects short sentences. The value of accuracy of our POS tagger trained on SVM has decreased [sic!] from 61.78% (non-segmented corpus) to 63.66% (corpus segmented with the second method of hybridization). Likewise [sic!], the accuracy increases from 62.53% to 64.84% when Ripper is used for training the tagger."(Zribi et al., 2016, 330) The accuracy for the SVM-POS-tagger *increased* for all approaches except PART when compared to the approach with non-segmented input. Furthermore, the best result was achieved using the non-segmented corpus with training based on PART (rule-based). If sentence boundary detection truly improved the performance of the POS-tagger, we would expect that 1) the manually segmentation is always better than no segmentation, and that 2) there is a general trend that inducing segmentation into the data improves the accuracy. 1) clearly does not hold as demonstrated by the superior performance of PART on NoSeg. Also regarding 2), we see contradicting results in Table 2.4. For the PART-based tagger, any segmentation performs worse than no segmentation. Segmentation induced by PART (AutSeg)

yields worse results for all three parser than no segmentation. This is especially puzzling as Table 2.3 reports the sentence boundary detection performance of the PART approach as the best. For example, it is indeed confusing that the CR approach (contextual rules) ranks on the second last place in terms of sentence segmentation in Table 2.3 but on the second place for PART and SVM, and on the first place for Ripper in terms of POS-tagger-accuracy. They even seem to contradict themselves within a few lines of text: "We remark that the SBD system helps the TA POS tagger to improve its accuracy." and "We show that the best value is given by using non-segmented corpus."(Zribi et al., 2016, 330) Lastly and most importantly, varying the training algorithms for the parser and the input data is not a good idea as it impedes to conclude if changes in results are due to changes in the algorithm or in the data.
Zribi et al. (2016) were confronted with an almost identical problem as for the task presented here. They aimed to re-segment transcribed data from a not standardized, low-resource language into SLUs. Similar to Swiss German, Tunesian Arabic exhibits a lot of variation. Their data already contained segmentation into SLUs, thus allowed to extract linguistic rules by studying patterns of the SLU-boundaries and engage in statistical approaches.

Rehbein et al. (2020) engaged in "Improving Sentence Boundary Detection for Spoken Language Transcripts" for German as the title states. The paper describes different approaches to detect SLU-boundaries in the SegCor corpus[15]. SegCor "consists of 33 documents with more than 54,000 lexical tokens (...) that were divided into sentence-like units by the SegCor project."(Rehbein et al., 2020, 7103) The original data originates in the FOLK corpus that consists of "conversational speech with two or more speakers that was recorded in non-laboratory settings"(Rehbein et al., 2020, 7103). Topics of the conversations include, to only name a few, children taking in kindergarten, teacher giving feedback, couples talking, presentations held by experts. While the original segmentation of the data is based on 'dialogue contribution', i.e., one consecutive unit of speech from one speaker, they worked with a version that was manually divided into SLUs (see Westpfahl and Schmidt (2016) for the manual segmentation). A contribution might contain various SLUs and one SLU might stretch across more than one contribution. There is, lastly, a mixed case, where a contribution contains various SLUs and

[15] `https://segcor.cnrs.fr/`[06.02.2023]

	# tokens		# SLU	
	SegCor	**KiDKo**	**SegCor**	**KiDKo**
train	38,293	230,166	7,756	61,524
dev	5,578	33,265	1,213	8,985
test	10,841	65,845	1,771	16,634
total	**54,712**	**329,276**	**10,740**	**87,143**

Figure 2.5: Statistics for Train, Dev, Test Data from SegCor and KiDKo (Rehbein et al., 2020, 7105)

one of them might continue in the next contribution.

Figure 2.5 shows the number of tokens and SLU across the train, development and test set from SegCor corpus. Besides relying on the SegCor corpus, they engaged in training data expansion using another corpus KiDKo, also shown in Figure 2.5. KiDKo comes with punctuation, part-of-speech-tags and already divided into SLUs. The corpus "contains spontaneous peer-group dialogues of adolescents from multiethnic Berlin-Kreuzberg (around 266,000 tokens) and a supplementary corpus with adolescent speakers from monoethnic Berlin-Hellersdorf (around 111,000 tokens, excluding punctuation)."(Rehbein et al., 2020, 1704) They fine-tuned 3 types of pre-trained Bert models, all of them based on the pre-trained *dbmdz/bert-base-german-uncased*[16]. Before training models, they examined with a CRF model if adding more and more data from KiDKo corpus step-wise to the training data from SegCor would improve the results of the model for SLU detection. "[They] were surprised that the large KiDKo training set did not help at all to improve results for SLU detection in SegCor."(Rehbein et al., 2020, 7106) In a previous article, Ruppenhofer and Rehbein (2019) had already examined and shown that CRF models can be outperformed by neural models with contextual string embeddings (e.g. Bert), thus the CRF model was only used to study the potential impact of adding data from KiDKo on the SegCor evaluation.

Firstly, they fine-tuned the pre-trained Bert model as a sequence tagging task, meaning that each token was assigned a label and the model was trained to predict these correctly. They used B for boundary and O for non-boundary. With this model, they already achieved rather impressive results with an accuracy of 95.1% reported in Table 2.1. Secondly, they experi-

[16]Available at: `https://huggingface.co/dbmdz/bert-base-german-uncased`[06.02.2023]

Model	Accuracy	F1
Sequence tagging	95.1	89.7
Sentence pair	96.0	91.5
Transfer learning	96.3	92.0

Table 2.1: Average Results Reported in Rehbein et al. (2020)

mented with modeling SLU-detection as a sentence pair classification task. The model receives two inputs S1 and S2 and has to decide if they form an SLU together or not. In order to create these input pairs, they iterated over all tokens. For each token t, they set t to be the last token of S1 and take at least ten previous tokens of t for S1 and at least ten following tokens for S2. Doing so, the model will decide for (almost) all tokens t if an SLU-boundary follows the token or not. Both of these model were trained on the SegCor training data. The results improved slightly when modelling the task as a sentence pair classification. Thirdly, although adding KiDKo to the training data did not improve the results of the CRF models, they tried to leverage this data. In their final experiment, transfer learning, instead of mixing both training data sets, they fine-tuned the pre-trained Bert model first on the KiDKo data and followed by a second fine-tuning on the SegCor data. Besides improving results further by almost 1%, they, more importantly, found "that this procedure results in a more robust classifier, with hardly any variation between the results for the different runs. In contrast, our previous Bert models are highly sensitive to initialisation (...)."(Rehbein et al., 2020, 7109)

For further reading regarding other languages, some references include: Wang et al. (2019) (sentence segmentation embedded in a English-Chinese translation task), Downey et al. (2021) (sentence segmentation for low resource Mayan language). For further reading about German, the papers from Glaser et al. (2021) (chunking for legal German) and Ortmann (2021) (chunking for historical German) might be a good starting point.

2.3 Dependency Parsing

Dependency parsing, especially with the rise of neural networks, might not be the most prominent or prestigious task in NLP, it remains, however, an important one and provides the basis for many different further processing steps in NLP (e.g. grammar or spellcheckers). Dependency parsing also explicitly learns and annotates the structure of a language. It follows the developments in NLP, meaning that statistical approaches were used at the beginning, while now most state-of-the-art parsers are based on neural networks. An important distinction when talking about syntactical parsing is the difference between dependency and constituency parsing. Constituency parsing aims to group words forming a syntactical unit together, e.g., an article and a noun form a noun phrase, while dependency parsing annotates the relationship between single tokens, meaning that the article would depend on the noun and the relationship is marked as a determiner (det). In dependency parsing, a head is assigned to each token of a sentence and the syntactic relation between the head and the child is labeled according to a predefined dependency grammar. Both parsing approaches require a pre-defined set of labels to annotate the syntactic relationships of a sentence. These tag-sets can be language specific. There is, however, a widely known approach to create a tag-set aiming to annotate all languages of the world.

"*Universal dependencies* (UD) is at the same time a framework for crosslinguistically consistent morphosyntactic annotation, an open community effort to create morphosyntactically annotated corpora for many languages"(de Marneffe et al., 2021, 255) As the name suggests, UD is a way of annotating *dependency* parsing. Universal Dependencies provides currently data, guidelines and treebanks (data corpora containing annotated syntax trees) for 138 languages[17]. The current UD is the updated version, see Nivre et al. (2020) for information about the change from UD1 to UD2. For each language, there might be some specific rules how to annotate certain cases, language-specific labels or more fine-grained label-supplements. The official annotation guidelines with examples for German can be found on the web-page[18].

The common data format used for dependency parsing is *CoNLLU*[19]. CoNLLU-files are tab-

[17] `https://universaldependencies.org/`[27.02.2023]
[18] `https://universaldependencies.org/de/`[27.02.2023]
[19] `https://universaldependencies.org/format.html`[27.02.2023]

separeted uft8 files composed out of ten columns. For dependency parsing, the relevant columns are: 4. UPOS - Universal part-of-speech tag; 5. XPOS - Language-specific part-of-speech tag; 7. HEAD - Head of the current word; 8. DEPREL - Universal dependency relation to the HEAD. For German, *Stuttgart-Tübingen-TagSet* (STTS)[20] is usually used as the fine-grained language-specific tagset.

Many languages have their own annotation scheme for syntactic relations. In the case of German, a famous tag-set was introduced with the *TiGer-dependency-bank* (Forst et al., 2004) (see Uszkoreit et al. (2003) for constituency parsing). It is more fine-grained than the UD-tag-set and more tailored to the German language. For example, the TiGer dependency tag set has its own label for a genitive object, which does not exist in UD and needs to be annotated differently (it may also vary with the language in discussion). Usually, different tag-sets for a language are not compatible with one another, and neither are dependency and constituency sets. There is, however, a conversion from the tag-set applied in the TiGer tree-bank (constituency) to German UD (dependency). Çöltekin et al. (2017) introduced this conversion for the TiGer treebank and others (e.g. TuBa-D/Z[21]). Generally, a conversion from UD to a more fine-grained tag-set is not possible.

In recent years, UD was used more frequently to annotate spoken language. Dobrovoljc (2022) compared different UD-treebanks for spoken language and concluded "that the spoken language treebanks differ considerably with respect to the inventory and the format of transcribed phenomena, as well as the principles adopted in their morphosyntactic annotation. This is particularly true for the dependency annotation of speech disfluencies, where conflicting data annotations suggest an underspecification of the guidelines pertaining to speech repairs[22] (...)."(Dobrovoljc, 2022, 1798). This does not come as a surprise, UD encodes the standardized dependencies for a language but spoken language usually does not comply with this standardization and is accordingly challenging to annotate. Speech repairs here means correcting a previously uttered mistake while speaking. A common speak repair in spoken German would

[20] https://www.ims.uni-stuttgart.de/forschung/ressourcen/lexika/germantagsets/[27.02.2023]

[21] https://uni-tuebingen.de/fakultaeten/philosophische-fakultaet/fachbereiche/neuphilologie/seminar-fuer-sprachwissenschaft/arbeitsbereiche/allg-sprachwissenschaft-computerlinguisti ressourcen/corpora/tueba-dz/[27.02.2023]

[22]

be a correction of the definite article; a speaker might start with *der*, then opts for a different noun and has to correct to *die* or *das*. A frequent example from ArchiMob is the repair of the auxiliary verb: *man hat man ist gan baden* (one had one 'went' swimming). The speaker probably did not intend to use the word *gan* as *baden* alone requires *hat* but in combination with *gan ist* must be used.

2.3.1 Dependency Parser

Dependency parsers need to be trained on annotated data[23]. There are statistical and neutral network based approaches (among others) for training parsers. Alternatively, a pre-trained dependency parser can be used. *MaltParser*[24] and *Wapiti-Parser*[25] are two examples of statistical parsers which were also applied in Hollenstein and Aepli (2014). More recent parsers tend to be based on neural networks. *Parsito* is "a neural network classifier for prediction and requires no feature engineering"(Straka et al., 2016, 4293). Parsito serves as the dependency parsing component of *UDPipe*, a trainable pipeline that "performs tokenization, morphological analysis, part-of-speech tagging, lemmatization and dependency parsing for nearly all treebanks of Universal Dependencies (...)"(Straka et al., 2016, 4290). Straka et al. (2016) achieved a Labeled Attachment Score (LAS)[26] of 78.6 for all avalaible UD German treebanks (15'894 sentences, 2016, UD version 1.2). This result was achieved when providing manual gold POS-tags with the data, with UDPipe generated POS-tags the parser scored a LAS of 71.8.

de Kok and Pütz (2020) engaged in a student-teacher-model approach aiming to improve the performance of a bidirectional LSTM parser for German and Dutch. They tackled dependency parsing as a sequence labeling task. With their approach, they managed to improve the performance of their bidirectional LSTM parser from 92.23 to 94.33 LAS for German. Results for Dutch were slightly lower. More interestingly, however, they elaborate on common errors

[23]There are approaches which try to avoid this as for example the model transfer approach presented in Aepli and Clematide (2018).

[24]`https://www.maltparser.org/userguide.html`[27.02.2023]

[25]`https://github.com/wireghoul/Wapiti-Parser`[27.02.2023]

[26]Explanation of LAS see Section 4.2

induced by statistical parsers when engaging in UD dependency parsing for German and Dutch. For their baseline parser (also bidirectional LSTM) they reported the ten most frequent labels causing problems: obl, nmod, conj, parataxis, nsubj, amod, obj, appos, root, advmod. "These relations are particularly difficult to parse, because the correct attachment is not only determined by syntactic constraints, but also by semantic preferences (...)"(de Kok and Pütz, 2020, 91) They fine-tuned various models and engaged in different strategies to reduce the errors for the error-prone labels. They managed to improve LAS for all the labels, where improvements ranged from 0.71 up to 7.47 LAS-points.

Chapter 3

Data and Methods

The present work investigates the impact of re-segmenting unstructured Swiss German audio transcripts into SLUs on dependency parser performance. Accordingly, the previous Chapter 2 provided background knowledge on Swiss German, sentence re-segmentation in spoken language and dependency parsing. The variation within the Swiss German dialects and the differences to Standard German were exemplified in detail (Section 2.1), furthermore, ArchiMob and other Swiss German resources were introduced. Section 2.3 provided the necessary background for dependency parsing and emphasized challenges. This chapter elaborates on the approach undertaken to tackle re-segmentation into SLUs for spoken Swiss German (ArchiMob) and the examination of the impact on dependency parsing.

Section 3.1 argues that a neural network approach seems most promising and introduces the choice for the neural network (NN).

Swiss German is a low-resource language lacking standardization. Therefore, the lack of available data and variation in data is a pressing problem, then Section 2.2 revealed most existing approaches for SLU-segmentation are supervised. A supervised learning setting requires test and training data. To engage in a supervised setting, we need data that (1) contains SLU-boundaries in any form; (2) has word-level normalization, and for the test set additionally (3) dependency parsing annotation. ArchiMob, however, has no alignment between utterances and SLUs (1) and no dependency labels (3). Hence, a test set had to be manually created, adding

(1) and (3); and moreover training data from a different source was required.
Section 3.2.1 elaborates extensively on the manual segmentation process of the test set while concurrently serving the purpose of further exemplifying differences between Swiss German and Standard German, and the unstructuredness of the ArchiMob data. The manual dependency annotation, the chosen tag-set and the dependency parser are presented in Section 3.2.2.
For the training data, normalization (2) is the most crucial point, given the differences among the dialects, especially in orthography, as explained in Section 2.1. The SwissDial corpus (see Section 2.1.3) was chosen as out-of-domain training data and was automatically processed to resemble the structure of ArchiMob. The purpose of this argumentation is elaborated in Section 3.3. The following Section 3.4 presents mentioned data processing steps and introduces terminology for the different data sets, followed by the data statistics in Section 3.5.

3.1 The Approach to the Problem of Re-Segmentation and its Impact on Dependency Parsing

For structured text, SLU-boundary detection tends to be a simple task and mainly relies on disambiguating full stops. The previous chapter left no doubt that the same task for spoken language transcripts is not trivial and disambiguating punctuation is not an option. We, therefore, require an algorithm that catches the patterns marking SLU-boundaries to re-segment ArchiMob into SLUs. Section 3.2.1 will exemplify how complex such patterns are for unstructured data as it is the case for spoken Swiss German.
The situation of the presented related work differs mainly in the availability of segmented data. SegCor corpus used by Rehbein et al. (2020) and the Tunisian Arabic one presented by Zribi et al. (2016) already contained SLUs, since both corpora were manually split into SLUs. Zribi et al. (2016) relied on a semantic concept for the notion of an SLU, while for SegCor prosodic features played an important role (see Westpfahl and Schmidt (2016) and Rehbein et al. (2020)). Leveraging grammatical features to create rules, as performed by Zribi et al. (2016), might be a promising approach for SLU detection in Swiss German transcripts. However, for ArchiMob,

we do not have the data segmented into SLUs to study and extract rules. Moreover, Section 2.1 elaborated on the differences between Standard German and within the Swiss German dialect continuum. This demonstrates that finding rules governing SLU-boundaries is not trivial and highly dependent on the dialect and the speaker.

Rehbein et al. (2020) used the provided segmentation to engage in a supervised machine learning setting fine-tuning pre-trained Bert models. They motivated training neural networks by referring to previous work showing the superior performance of such models over CRF-models. This research follows an approach similar to Rehbein et al. (2020) and engage in a binary token classification task fine-tuning a Standard German Bert model. Tokens are labeled with **I**nside or **E**nd of SLU and we aim to predict tokens that mark the SLU-boundary (**E**). Recent research about linguistic structure captured by NNs further supports the choice of methods. Dependency parsing is a way of explicitly annotating (linguistic) structure of a language. Manning et al. (2020) "(...) demonstrate that modern deep contextual language models learn major aspects of this structure, without any explicit supervision."(Manning et al., 2020, 30046) By supervision they mean that the models are not specifically trained to do so. The Bert model we fine-tune is such language model for German that should be able to capture parts of the structure we explicitly annotate with dependency annotation.

Bert is a transformer-based neural network architecture. Bert stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers and was first introduced in Devlin et al. (2019). Transformers make extensive use of different attention mechanisms and include positional embeddings. Non-positional embeddings follow a bag-of-word approach, mapping each token to one value in the embedding space completely ignoring surrounding tokens and the position within the sentence. Positional embeddings take word order and surrounding tokens into account. SLU-boundaries in spoken language are not strictly governed by standardized grammar but are rather highly dependent on the context. For example, placing part of the predicate at the last position of subordinated or main clauses is often disturbed by placing other constituents in this position, by repeating parts of the utterance, with interruption of the current utterance, with absence or omission of the part of the predicate that should be placed last in general, or with other phenomena. Hence, only the context can help to recognize such SLU boundaries and

Bert's positional embeddings might be able to capture the complex, irregular patterns governing SLU-boundaries for transcribed Swiss German. "Generally, if words appear close to each other in a text (i.e., their positions are nearby), they are more likely to determine the (local) semantics together, than if they occurred far apart. Hence, *positional proximity* of words x and y should result in proximity of their embedded representations $\vec{x}$ and $\vec{y}$." (Wang et al., 2021, 2) The uncased German Bert mode used for the experiments in this research leverages absolute positional embeddings (Gehring et al., 2017). In simple terms, the absolute positional embedding of a input token x_i is a combination of a 'normal' word embedding w_i and the embedding p_i of the position of token x_i within the input (see Gehring et al. (2017)). Wang et al. (2021) conduct an empirical examination of seven positional embeddings (and their combination) of Bert models for classification (and span prediction).

For the second part of the research question, the impact of SLU re-segmentation on dependency parsing, it suffices to select a pre-trained dependency parser and an annotation schema. We examine the impact and are interested in the improvement we can achieve with SLU-re-segmentation. Accordingly, the absolute performance of the dependency parser is secondary. We will use the UD tag set for German. On the one hand, we want to contribute to and support the open UD-community. On the other hand, results based on UD are better comparable to other languages. As the task tackled here might be interesting for other low-resource languages, UD based results provide a better point of reference. As a dependency parser, UDPipe was chosen (see Section 2.3; Straka et al. (2016)).

3.2 Manual Creation of Test Set

Framing re-segmentation into SLUs as a token classification with a neural network is a supervised learning setting. As stated at the beginning of this chapter, this setting requires a test set that (1) contains SLU-boundaries in any form; (2) has word-level normalization; and (3) dependency parsing annotation. The manual segmentation into SLUs is presented first, followed

by the manual dependency annotation. The following two sections presupposes familiarity with linguistic terminology and a rather profound understanding of Standard German grammar.

3.2.1 Manual Test Set Creation

The test set contains 203 SLUs. This slightly odd number of test sentence-like units is due to the fact that originally 200 SLUs were selected. Later, following the rules in table 3.1 rigorously and iteratively re-evaluating the splits made, a few sentences had to be split up further, thus, yielding a total of 203 sentences.

For the test set, two interviews from different dialectal areas were chosen and a little more than 100 utterances extracted from each one. One interview was held with a male speaker from Bern born in 1920, the second one comes from a female interviewee from the dialectal area of Zurich born in 1928. These two dialect areas were selected out of several aspects. They are quite different from each other in terms of pronunciation, vocabulary and some grammatical aspects. Moreover, variants of both dialects are the most commonly spoken dialects in Switzerland. Zurich is the most populous city in Switzerland, and so is the region (Kanton) of Zurich. Bern as a city ranks on the fifth place, respectively on the third if only the German speaking part of Switzerland is considered, and is the second most populous region (Kanton). (Bundesamt für Statistik, 2022a, 40) Bern is also the seat of the government, and the dialect from Bern is considered by many native speakers (of any Swiss German dialect) the most beautiful and popular one (Gasser, 2020). The motivation for this selection is clearly that those two dialects are the dialects (or variants of it) most commonly used, written and spoken, accordingly it would be a huge step to improve NLP techniques for these two dialects. Whereas the dialect from Zurich is closer to modern Standard High German, the dialects from Bern conserve more dialectal features as it is most apparent in the vocabulary (e.g. Anke[1] (butter), Nidlä [2] (a type of cream))[3].

[1] `https://www.idiotikon.ch/wortgeschichten/anken`[16.01.2023]

[2] `https://digital.idiotikon.ch/idtkn/id4.htm#!page/40671/mode/2up`[16.01.2023]

[3] Although, the traces of the word Anke go back to the 12th century and it used to be the standard word for butter in the Alemannic dialect areas of Switzerland and South Baden (Südbaden), its usage is disappearing. The Idiotikon explains that "since we [people in the German speaking part of Switzerland] started buying 'edible fat' [Speisefett] packaged in supermarkets, the Standard German word Butter is penetrating [the dialectal areas]

Number	Rule
R1	Split into shortest SLUs possible
R2	Split parataxis if topic changes
R3.1	Split on coordinating conjunctions if the topic changes or they are used as 'next', 'and then' etc.
R3.2	Coordinating conjunctions not used as such go to the beginning of the next SLU
R3.3	'oder' not used as a conjunction goes to the end of an SLU
R4	Do not split if the second SLU shares a constituent with the first one
R5	If two consecutive SLUs do not share a constituent, but one of them is syntactically incomplete, split only if they are not semantically related

Table 3.1: Guidlines for Manual Segmentation

Although test data is usually chosen randomly from the whole data set, this might not be the best approach for ArchiMob. The data is segmented into utterances which do not follow semantic nor syntactic criteria at all. Some utterances contain various sentence-like units others none at all. Even for a native speaker it is often not possible to decide on the boundaries of sentence-like units if the context is not provided. As the interviews are one-sided, such 'context window' of SLUs belonging together are rather large. For that matter, the utterances were not selected at random, but the original order of the utterances within the interviews was preserved. From both interviews, a bit more than the first 100 utterances were selected for further manual editing. For the manual re-segmentation process, a few guidelines had to be established, which are displayed in Table 3.1. Usually, a language knows no grammatical restriction for how many main clauses can be stacked consecutively. Especially in spoken language, this is a common phenomenon. German as a language is prone to construct long, complex sentences, even in structured formal writing. Therefore, the guidelines revolve around such chains of SLUs (or sentences) and specify when and where to split. Split refers here to inserting an SLU-boundary. The notion of SLU applied here aims at a compromise between grammar and semantics. We aim to capture the smallest (possibly) syntactically complete (!) SLUs, which is expressed by **R1**.

on a broad front." - Original: "Seit wir das Speisefett fertig abgepackt bei den Grossverteilern einkaufen, dringt auf breiter Front das schriftdeutsche (aber immer noch mit dem männlichen Geschlecht von Anke und Schmalz verbundene) Wort Butter vor." (Landolt, 2015)

R2 - Parataxis

In German grammar, parataxis (*Parataxe*) refers to the coordination of two or more equal (*gleichrangig*)[4] clauses by using conjunctions (und, oder, wie etc.) or punctuation (, and ; and -) (Averintseva-Klisch, 2018, 25). A parataxis can occur between main clauses or subordinated clauses but as a subordinated clauses can never form a proper syntactical unit according to Standard German grammar, R2 only refers to parataxis with main clauses. Parataxis with subordinated clauses is (almost) never split into SLUs. A parataxis without conjunctions can always be split into single main clauses by replacing coordinating punctuation with full stops. Following, R3 specifies how to treat constructions with coordinating conjunctions.

R3 - Coordinating Conjunctions

Coordinating conjunctions are very common in spoken languages and appear in many places where they do not show up in written language, further rules on how to treat them had to be established. In written language, however, it is considered a sign of poor stylistics if a main clause starts with a coordinating conjunctions, although grammar rules do not forbid it. Rules R3.1-R3.3 specify when to introduce a sentence split in the presence of a coordinating conjunction, and where to place the conjunction. Even for a native speaker it is not a trivial task to decide which coordinating conjunctions are indeed used as such and which are tied to spoken language. Therefore, the splits made implementing R3.2 are subject to the author's own interpretation of the semantics of these conjunctions, and might be debatable. The *oder* (or) specified in R3.3 is a highly characteristic feature of Swiss German. It is the standard question tag for many Swiss German varieties, and is referenced often in popular culture. For example, a humorous five-day language course for Swiss German lists for day one to use *oder*[5] after every sentence (Schweiz für Dummies). As a final remark when talking about conjunctions, German grammar knows a subcategory of adverbs (in the sense of German grammar) that might look like a conjunction on first sight and that is often used very similarly. This category is called

[4]For German grammar, the distinction between main and subordinated clauses is highly important.
[5]Actually pronounced as *odr*

Konjunktionaladverb or *Konnektoradverb* meaning 'adverb used as a conjunction or connector'. The difference is that a *Konjunktionaladverb* may appear in the position of a conjunction but is, opposed to a real conjunction, free to alter its position in the sentence (see (Eisenberg, 2009, 584)). Typical examples include *dann, deswegen, jedoch* (then, therefore, however) and many more. For the sake of simplicity, this distinction is not maintained throughout this thesis. All *Konjunktionaladverbien* will be labeled and treated as conjunctions[6].

R4 and R5 - Contracted Sentences

In German grammar, a contracted sentence is defined as at least two coordinated (main) clauses which share a constituent that is omitted in the latter clauses (Speyer, 2016). Commonly shared constituents are subject and auxiliary verbs. R5, lastly, serves as a guideline on how to treat syntactically incomplete constructions of any kind. This might be a parataxis of ellipsis or other phenomena coordinating incomplete main clauses. As the data set consists of spontaneous interviews, i.e., none of the interviewee prepared answers, it is full of sequences of syntactically incomplete (main) clauses which lack grammatical features inducing sentence splits. According to R5, we rely on semantics to segment such sequences into SLUs.

Examples

The manual SLU-creation is illustrated with examples[7] from the data for each 'rule'. $< x >$ refers to the utterance number within the test set. Utterances were extracted and simply enumerated. $< x >$ marks the utterance boundaries, whereas | denotes a manually induced SLU-boundary; this illustrates that utterance and SLU-boundaries do not coincide. Examples come as triplets of *Original Utterance*, *Standard German*, *Translation*. Although labeled 'translation' it, in fact, does not represent a real translation from Swiss German to English. While the *Standard German* version is correct according to German grammar, orthography

[6]To the best knowledge of the author, the UD guidelines do not distinguish the two categories. Given that they fulfill more or less the same function, it is reasonable not to distinguish them for dependency labels.

[7]Most examples do not correspond to one single rule but rather to various. Especially R1 - R3.1 tend to overlap.

and punctuation, this does not hold true for English. The English translation does not only convey the meaning but stays as close as possible to the Swiss German version in an attempt to illustrate the unstructuredness of the original data. This, of course, comes at a cost. Often the English 'translation' is a bit odd in terms of vocabulary and word order, some sentence might even be ungrammatical.

Table 3.2 shows an example utterance that was split into three SLUs. Although the first split might correspond to R3.1 and the second one to R5, the example still shows how one utterance was divided into as many and as short SLUs as possible.
The next example, shown in Table 3.3, contains a typical parataxis of the form *main clause, main clause coordinating conjunction main clause* rendered as such in the Standard German translation. The different main clauses were split into SLUs.

Original Utterances	< 22 > das sind ein haufen italiener gewesen | aber das hat man nicht gewusst | das sind einfach
Standard German	Das sind ein haufen Italiener gewesen, aber das hat man nicht gewusst. Das sind einfach.
Translation	and that was a bunch of italians | but one didn't know that | those were simply

Table 3.2: Example of R1

Original Utterances	< 3 > wir sind da bei ihr daheim | es ist der einundzwanzigste märz zweitausend und eins | und ich bin die tanja wirz
Standard German	Wir sind da bei ihr zuhause, es ist der einundzwanzigste März zweitausendundeins, und ich bin Tanja Wirz.
Translation	we are there at her home | it is the twenty-first of March two thousand and one | and I am Tanja Wirz

Table 3.3: Example of R2

For R3.1 an example can be found in Table 3.4. The split before *aber* (but) is based on a change in topic. The ellipsis before does not share a topic with the second, complete sentence, nor does it belong to the previous utterance. In addition, this example shows a difference in word order, whereas for Swiss German the order is *modal verb main verb* it is inverted for Standard German.

Original Utterances	< 92 > und das ist | aber eine stunde haben wir müssen laufen bis wir in dem wald gewesen sind
Standard German	Und das ist**.** Aber eine Stunde haben wir laufen **müssen,** bis wir im Wald (gewesen) waren.
Translation	and this is | but we had to walk an hour until we were in the forest

Table 3.4: Example of R3.1

Original Utterances	< 66 > und das ist gezahlt worden vom italienischen staat | und auf dem dann sind sie auf dem schiff gewesen
Standard German	Und das ist vom italienischen Staat gezahlt worden. (und auf dem) Dann sind sie auf dem Schiff gewesen.
Translation	and this was paid by the Italian state | and on the then they were on the ship

Table 3.5: Example of R3.2

In Table 3.5 utterance < 66 > starts with the coordinating conjunction *und* (and) from the previous utterance based on a change in topic. The second *und* marks an SLU-boundary as it conveys the meaning and function of 'then' or 'next'. The actual appearance of *dann* (then) after the *und* solidifies this claim; *und dann* (and then) is very common in spoken German when narrating. One could also argue that the topic changes rather drastically from SLU 1 to SLU 2 in this example. Besides exemplifying R3.2, the sentence also shows why manual re-segmentation is a challenging task. The preposition and article from the prepositional phrase (*Präpositionalgefüge*) 'on the ship' appears twice. Furthermore, we find two conjunctions *und* (and) and *dann* (then), while in written Standard German we would only expect one. To com-

plicate issues even more, each of the two conjunctions would require a different word order in Standard German. This is due to the fact that *dann* is a *Konjunktionaladverb* and not a conjunction, while *and* is a true conjunction. If a *Konjunktionaladverb* is placed at the beginning of a main clause, the conjugated verb must follow. However, if a true coordinating conjunction is used, the position before the conjugated verb is available and must be occupied. The correct word order would be: *und* ***sie*** *sind* versus *dann sind* ***sie***. Although, the repetition and incorrect word order does not affect intelligibility in this sentences, yet it does in more problematic ones. The phenomenon simply arises as the speaker starts formulating the sentence and then restarts with a different expression. For a perfectly clean and grammatically correct test set, one would ideally strip *und auf dem* producing a proper sentence. Doing this is strongly discouraged for two reasons. 1) *und auf dem* is clearly missing a constituent that appears only a few words later (*Schiff* (ship)). 2) the ArchiMob corpus is full of similar utterances which constitute one of the main challenges when re-segmenting this corpus (and transcribed spoken language). If the manual test set is too correct, the results of the evaluation might not be representative compared to the performance when applied to the whole, not manually corrected corpus.

Table 3.6 displays a sentence containing *oder* in the function of a question tag. Besides that the second SLU contains an interesting phenomenon. The indirect object (*Dativobjekt*) *den Kunden* (to the customers) is positioned after the past participle *vorgeführt* (demonstrated) which is forbidden in proper Standard German without punctuation. This might occur in spoken language, as the indirect object is added after the end of the sentence as additional information. A more natural word order would be: Adverb - conjugated verb - subject - adverb - indirect object - direct object - remaining verb(s). This yields: *Dann hat man dort in Walenstadt den Kunden die Waffen vorgeführt.* (Then, 'one' demonstrated the weapons to the clients there, in Walenstadt). For a native speaker, word order does not affect intelligibility at all. It, however, renders the re-segmentation task more difficult. According to German grammar, if the predicate is a complex one, i.e. composed of several verbs, all of them, except the conjugated one, have to be placed at the last position of the sentence (there are rules governing the order). This implies that in most cases an SLU-split should occur after the verbs

marking the last position within the clause but apparently not in all cases. We must not isolate the indirect object by cutting it off from its clause. This is easy for a native speaker as one can simply rely on semantics and the fact that *vorführen* can be mono-, bi- or trivalent.[8] In the example, the verb is used as a trivalent verb, i.e. ditransitive.

Original Utterances	< 163 > das ist jetzt etwa neunzehnsechsunddreissig siebenunddreissig oder | dann hat man dort in walenstadt die waffen vorgeführt den kunden
Standard German	Das ist jetzt etwa 1936/37, oder. Dann hat man dort in Walenstadt die Waffen vorgeführt, den Kunden.
Translation	that was circa 1936/37 or | then one demonstrated weapons to the clients there in Walenstadt

Table 3.6: Example of R3.3

Original Utterances	< 17 > meine eltern haben ein zweifamilienhaus gehabt < 18 > und ein einfaches zweifamilienhaus
Standard German	Meine Eltern habe ein Zweifamilienhaus gehabt, und ein einfaches Zweifamilienhaus.
Translation	my parents had a two family house and a simple two family house

Table 3.7: Example of R4

Utterance < 18 > from Table 3.7 shows an ellipsis where everything except the direct object was left out, and therefore, no split can be introduced. < 18 > shares at least the subject *meine Eltern* (my parents) with the previous utterance.

Table 3.8 shows a very confusing sentence from the test set. It consists of only one utterance that is almost not intelligible. Without doubt, only the last SLU is clear in its meaning. The interviewee starts the utterance twice with almost identical phrasing but then re-decides and states something different. Even with a broader context, taking into account the previous and following utterances, it is not possible to disambiguate various aspects. It cannot be determined

[8] A usage with valency four is also possible

whether *hat* (formally: the third person singular, indicative present of the verb to have) is used as an auxiliary verb or as a full verb - which is relevant for dependency labels. Therefore, the English translation mentions 'has' in brackets marking the uncertainty with a question mark. Additionally, the *ä* causes problems as well. It is unclear if it represents a common sound made when pondering, reformulating or correcting an utterance just spoken, or if it is the indefinite article. As mentioned in Section 2.1.2, the ArchiMob corpus was normalized but in this case the normalization did not work properly. In any case, both clauses miss important constituents and they certainly do not share any. Based on the lack of information, it cannot be determined if they are semantically related or not. Although they most probably are, they are equally probably not semantically related to the third clause. The author decided to interpret the first two clauses as an attempt to formulate a chain of thought not related to the third one, thus leading to a third re-formulation attempt. In combination with R1, this led to splitting all three clauses into single SLUs.

Original Utterances	< 100 > und sie hat | sie hat ä | dort ist sie so tüchtig geworden
Standard German	Und sie hat. Sie hat ähm (ein?). Dort ist sie so tüchtig geworden.
Translation	and she did (has?) | she did (has?) uhm (a?) | there she became so competent

Table 3.8: Example of R5

Original Utterances	< 215 > ja also eben mein vater ist mit drei jahren in die schweiz *gekommen* < 216 > aus deutschland < 217 > mit der familie also handwerksfamilie | < 218 > und er ist der jüngste gewesen von
Standard German	Ja. Also (eben) mein Vater ist mit drei Jahren **aus Deutschland mit der Familie, also Handwerksfamilie, in die Schweiz gekommen**. Und er war der jüngste von
Translation	yes so indeed my father came with three years old to switzerland from germany with the family a craftsman-family and he had been the youngest of

Table 3.9: Problematic Example E1

Table 3.9 displays a last utterly complex example. E1[9], once more perfectly intelligible for native speakers, is another example where constituents are placed after the 'closing' past participle (see Table 3.6). *gekommen* (come) does not mark the end of the SLU. In E1, two constituents[10] follow after the past participle, and the second one is, moreover, followed by a loose apposition (non-restrictive apposition). The correct SLU-segmentation marked in Table 3.9 is easily determined by semantics but rather difficult in terms of grammar. < 218 > complicates matters further, since this utterance starts with *und er ist* (and he had). If we ignore semantics, it is hard to determine the SLU-boundary in *also handwerksfamilie* \| *und er ist*. This *und* is an example of coordinating conjunctions from guideline R3.2. *er* is nominative and thus the subject of *ist* (third person singular). If *und* were a coordinating conjunctions, the

[9]E1 is not part of the test set but were put aside during the manual test set creation as backup sentences.
[10]It is not common to find more than one constituent placed and separated from the closing constituent of a sentence.

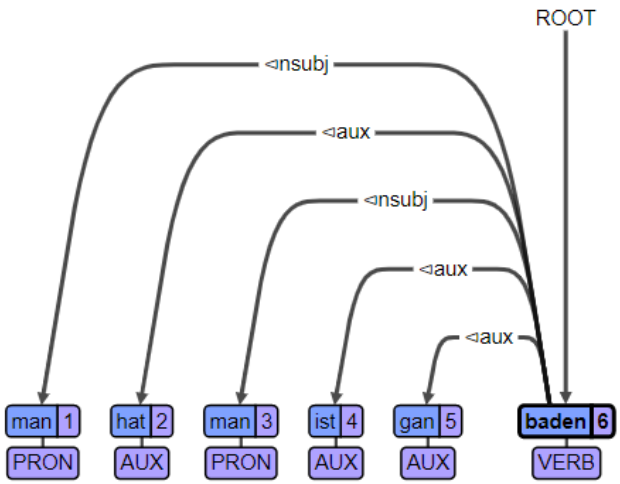

Figure 3.1: Syntax Tree Example from Test Set

following verb would be in plural. This leaves one wondering, how a computer should resolve this as it does not simply come down to a recognition of a pattern, even if all the morphological and syntactical information is provided - which is not the case for Swiss German.

3.2.2 Dependency Label Annotation

The dependency annotation was carried out using the official UD guidelines for German[11]. The guidelines, however, do not specify everything in detail. Annotating dependency labels requires a profound understanding of the grammar and the semantics of the language in discussion. Slight adaptation to and interpretation of the guidelines are based on (1) observations of the labels produced by UDPipe, (2) peculiarities in the ArchiMob data related to spoken language, e.g., repetition, and (3) characteristics of Swiss German that are not specified by the UD guidelines. In the following, examples for 1-3 these adaptations are provided and thoroughly explained or elaborated on. Adaptations (partially) coincide with 'problematic' labels as identified in Section 2.3.

(1) The UDPipe parser never produced the tag *discourse* which is used for interjections and other discourse elements that are not clearly linked to the structure of the sentence. Such tokens were annotated as *advmod*. ArchiMob would be full of such discourse elements, e.g., *jaja* (yes,

[11]Guidlines: `https://universaldependencies.org/de/`; complete list of labels used in German: `https://universaldependencies.org/de/dep/index.html`[27.02.2023]

yeah), annotating them properly would, however, worsen the parser performance artificially as it is not of importance if the tag *discourse* or *advmod* is used. Both usually get attached to the root of the sentence (*advmod* has different usages as well). *vocative* is used although UDPipe was never observed to output this tag. Simlarly, *appos*, apposition, does occur in the testset but UDPipe tends to label construction *appos* that are no appositions. UDPipe did, furthermore, not produce most available subtypes for German. Subtypes further specify a tag using a *:* notation. For example, *obl:arg*: *obl* is the tag for adjunct nominals or non-core nominals of a verb, *:arg* would be the subtyp for non-core nominals. *obl:arg* is, for example, the correct tag for (most) genitive objects. Subtypes used and produced by the parser are: *nsubj:pass, aux:pass, det:poss*.

(2) ArchiMob is full of *parataxis*, apart from not being produced by the UDPipe parser, such construction were split into SLUs (Table 3.1). Remaining cases of *parataxis* are annotated as *conj* without a coordinating conjunction *cc*. Many SLUs will start with a coordinating conjunction or have one dangling at the end. Accordingly, the test set contains a lot of *cc* connected to the root of the SLU without a conjunct (*conj*). As a consequence of the repetitions and reformulations in ArchiMob, some SLUs might have peculiar annotations, e.g., a finite verb (usually the root) that has various subjects (*nsubj*), a noun with various and different determiners (*det*). An example of an SLU with two identical subjects and two different auxiliary verbs is shown in Figure 3.1. This is a case of reformulation or speech repair (Dobrovoljc (2022) in Section 2.3).

(3) In the SLU *man hat man ist gan baden* (one had one 'went' swimming) in Figure 3.1 *gan* is an auxiliary verb that only occurs in Swiss German but never in Standard German(see Section 2.1). For native speakers, this is obvious. Firstly, in most dialects the auxiliary *to go* and the main verb are distinct (*go* vs. *gah*), as it is as well the case in the normalized form of the corpus (*gan* vs. *gehen*). Secondly, if *gehen* as a main verb is inserted into the sentence, it has to be placed at the: *man ist* ***gan*** *baden* ***gegangen*** which is a correct and natural sentence in Swiss German. Although easy for native speakers, such examples are utterly difficult for a dependency parser. Figure 3.1 shows the syntax tree of this example. Other peculiarities of Swiss German are less important but still challenging for a parser as it does not occur in Standard German training data, e.g., that proper names will have a determiner, *det*, at-

tached to them. The normalization also removed some peculiarities, as for example the merged indefinite article with a preposition that appears as two separate tokens in the normalized form.

The test set was first parsed with UDPipe and then UD dependency labels (lowercase) and POS-tags (uppercase) were manually corrected. This allowed to study phenomena with which UDPipe struggles. Errors in POS-tagging are relevant as they influence dependency parsing.

It is not surprising that UDPipe failed completely to recognized proper nouns (*PORNP*) from Switzerland (e.g. names of villages). Surprisingly, the parser also struggled to label common nouns like *Klasse* (class), *Name* (name), *Italiener* (Italian) correctly. It is 'expected' that UDPipe fails to distinguish adjectives (*ADJ*) and adverbs (*ADV*). This is challenging for German as many 'words' look the same and the distinction between adjective and adverb is (usually) not marked morphologically, for example *viel* (much or many) can be both. Distinguishing correctly between adverb and adjective requires a profound knowledge of German grammar and 'experimenting with word order'[12]. Similar issues occur with the articles (determiners), if they refer to a noun the correct UPOS-tag is *DET*, if they do not it is *PRON*. Not properly distinguishing *PRON* and *DET* influences dependency parsing: a *DET* can only be tagged as *det* while a *PRON* can receive a variety of dependency tags like *nsubj* (subject), *obj* (direct object) etc. The same is true for other incorrect UPOS-tags. Failing to distinguish *ADJ* and *ADV* leads to mistakes with dependency labels *advmod* and *amod*. *wo* (where) is question word (*ADV*) and relative pronoun (*PRON*) in Swiss German but only *ADV* in Standard German. Logically, UDPipe will fail to assign the correct UPOS-tag and hence, struggles with the correct dependency labels for relative clauses. UDPipe did also not manage to distinguish when *sein* (to be) is used as a full verb *VERB* and when as the copula *AUX*. As a consequence, UDPipe produces many errors with the dependency labels *root* (the full verb in most cases) and *cop* (copula).

More sources of common errors where identified. The examples above are sufficient to illustrate that a state-of-the-art-performance is not expected. This does not come as a surprise, Section

[12]True adverbs cannot be placed between article and noun, adjectives are allowed.

2.3 already stated that spoken language is challenging for parsers and that, depending on the language, specific tags cause problems.

As a closing remark, the manual annotated test set does contain syntax structure that might be invalid according to UD[13]. Some annotation might be subject to the author's interpretation. For example, if several *advmod* occur consecutively, one could modify the other or both could be attached to the root. German has a freer word order than English, thus, its constituents can be moved across the sentences which enables native speakers to better identify which constituents form a inseparable unit, which means that one *advmod* modifies the other, and which do not (both attached to the root). This, however, is strongly based on semantics and might be subjective.

3.3 Training Data

Ideally, training data from ArchiMob would be available but ArchiMob lacks an alignment between utterances and SLUs which is required for a supervised learning setting. Possible candidates for training data were introduced in the previous chapter. Section 2.1.1 introduced NOAH's corpus and SwissDial, besides ArchiMob. A third candidate is provided by SegCor (Section 2.2.1) used in Rehbein et al. (2020). NOAH's corpus might seem like a good candidate but is not exactly what we need. SLUs are provided as the corpus contains properly structured sentences. It does, however, not exist in a normalized form, which leads to high variation in the corpus. In other words, the same lexical word might appear in many different forms. This might pose a problem for the neural network approach presented at the beginning of this chapter in Section 3.1. Neural networks encode tokens using embeddings, and given the variation in the data, we would expect to end up with the same amount of variation for on lexical word in terms of the embeddings. Lastly, NOAH's corpus also consists purely of text and does not contain audio files.

SegCor consists of spoken language (audios and transcripts) and, as it is in Standard German,

[13]UDPipe does as well produce invalid structures

can be said to have a form similar to the normalized version of ArchiMob. The author could gain access to the SLU version of SegCor at some point but the definition of SLU was vastly different. The annotators of SegCor, for example, isolated (most) interjections into single SLUs which is different from the manual segmentation presented before, where interjections were always added to the following or previous SLU. SLUs in ArchiMob aim to be as close to proper sentences as possible, while the ones from SegCor rather represent chunks. SLUs from ArchiMob are thus longer and more complex than in SegCor. SegCor contains SLUs composed out of single tokens, whereas the manual test set from ArchiMob does not. While ArchiMob contains continuous narrations of childhood memories, SegCor consists of relaxed conversations with many speaker turns. The SLU-segmentation of SegCor was heavily influenced by prosodic features like pauses or speaker turns which are not frequent in ArchiMob and usually do not correspond to SLU-boundaries. Prosodic features were also encoded as part of the input for SegCor, which could in theory be applied to ArchiMob but, given the sparsity of prosodic features in ArchiMob, was disregarded. Furthermore, the two variants of German from each corpus are probably as distinct as possible. Not only are the speakers coming from opposite points of the German dialect continuum, with the Swiss Speakers originating in the very south and the ones from Berlin from the north, but they are also far apart in years. Both corpora contain vernacular that most probably is not mutually intelligible. Usually speaker of German dialects from Germany cannot understand Swiss German and Swiss German speakers tend to have problems understanding strong German dialects or vernacular. Given the differences in SLUS, vernacular and hypothesizing that a neural network would learn the SLU pattern applied in SegCor, which is different from the ArchiMob one, and thus yield poor results on ArchiMob; we decided to disregard SegCor as training data.

This leaves SwissDial (Section 2.1.3) as a candidate for training data. SwissDial does not consists of transcribed text but of aligned sentence pairs Swiss German - Standard German. The Standard German sentences are close to the normalization applied in ArchiMob and the properly structured text allows to interfere SLUs. The following section elaborates on the data processing workflow that also renders the out-of-domain data from SwissDial more similar to ArchiMob data.

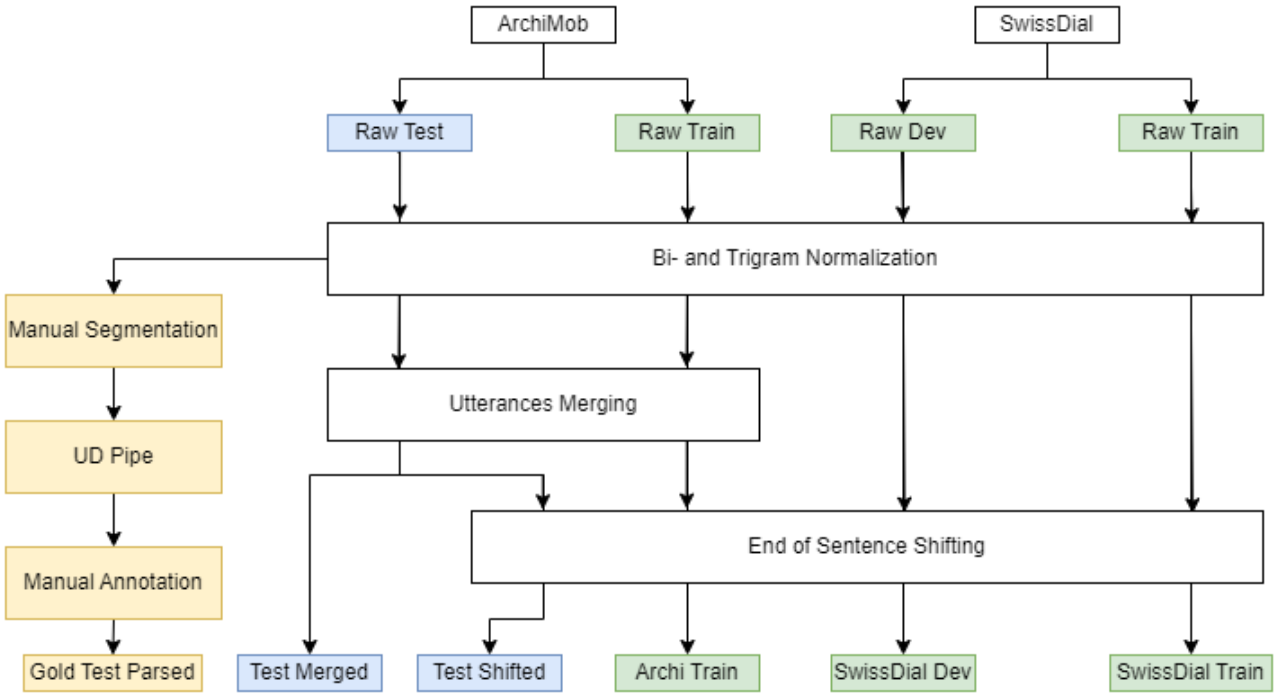

Figure 3.2: Data Processing Workflow

3.4 Data Processing Workflow

After establishing the choice for training and test data, it remains to present the data processing followed by the different splits of training, development (used for validation) and test data, and to display the corresponding statistics. Figure 3.2 shows the preprocessing pipeline and introduces terminology for the different shapes of data. As this thesis aims to examine the impact of input shape on dependency parsing, the same experiments were carried out with different input forms of data. It is particularly important to distinguish the different versions of the test data. Yellow refers to manually curated data, i.e., the gold test set. Blue is the same data as the manual set, but preprocessed completely or partially like the training data. For all experiments, Test Merged and Test Shifted were passed to the model for prediction. Green is the colour of the different training sets.

It is important to be aware that the steps shown in Figure 3.2 were applied to the normalized version of ArchiMob and to the Standard German sentences for SwissDial. For reasons of simplicity, the ArchiMob development set is not shown in the diagram, but it follows the exact same path as the Archi Train set. Some experiment settings (see Section 4.1) used extended

training data for which the ArchiMob and SwissDial sets were concatenated before using them for training, also not shown in Figure 3.2. Moreover, the diagram does not show the chronological order of the different steps. The manual set was created first and served as a base to design the other preprocessing steps. The manual creation was explained in detail in Section 3.2.1 and 3.2.2, called Gold Test Set. The Gold Test Set is only used for evaluation (see Section 4.1 and Section 4.2).

As a first step, the raw data was extracted from XML files for ArchiMob and from json for SwissDial. ArchiMob, however, is full of repeated words. In most cases, these repetitions consist of interjections like *Ja* (yes), *So* (so) or personal pronouns like *Ich* (I), *Er* (he), and *Sie* (she or they). Normalizing repetition is an important step to render ArchiMob more like proper written German, it might, however, induce an error in a few special cases. For example, in the utterance *wo die sirenen gegangen ist ist er weg kommen*[14] (When the sirens went, he went away) *ist* (is) occurs twice. In this special example, the two *ist* are necessary and would in proper writing be separated by a comma. The first *ist* serves as the auxiliary verb of the subordinated clause, while the second one is the auxiliary verb of the main clause. The auxiliary verb has to placed at the last position in the subordinated clause and at the second position for the main clause. As the first position is occupied by the subordinated clause, both auxiliary verbs are adjacent. It is to mention that these cases are rather rare and most of the times two identical finite verbs following each other are indeed repetitions.
Using sklearn's *CountVectorizer*[15] the identical bi- and trigrams (i.e., repeated tokens) were extracted and in a second step normalized to one token. This step was performed identically for both data sets and all subsets. Of course, as SwissDial consists of structured, clean sentences, this had no effect. It was ensured that only complete tokens are normalized in this step, no prefixes or suffixes were normalized.
As, especially after the normalization of bi- and trigrams, some of the utterances were very short, namely only containing one or a few tokens, a simple merging step was introduced.

[14]Standard German: Als die Sirene 'gegangen' ist, ist er weg gekommen.
[15]`https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html`[10.02.2023]

This step was, furthermore, motivated by the fact that the average token length of the manual segmented test set was roughly 2 tokens higher (9.91) than for the un-processed rest of the ArchiMob corpus (7.07 average tokens per utterance). As this is almost half the average tokens per utterance compared to SwissDial ($\approx$ 12 average tokens per sentence), very short utterances were merged. Any utterance with four tokens or less was merged with the previous or following utterance based on which one was shorter. For the rest of the ArchiMob data, excluding the test set, 11304 merges were executed, raising the average number per utterance from 7.07 to 8.70. For the Raw Test 17 merges were performed leading to an average token number per utterance of 9.67 bringing this value very close to 9.91 of the Gold Test Set (see Table 3.11 in the next section). Test Merged was not further preprocessed and consists of 208 lines. This merging step was not applied to SwissDial. Furthermore, merging was never applied across speaker turns, i.e., two utterances from different speakers were never merged.

SwissDial, in contrast, had still a different problem. As the sentences of SwissDial are properly structured text, the last token of each line will always mark the sentence boundary. As we hypothesize to capture the pattern governing the sentence boundaries in ArchiMob by fine-tuning a neural network, we had to ensure that the model will not learn that the last token of an input line is the sentence boundary. Thus, another step labeled *End of Sentence Shifting* was introduced. Shifting was performed in a very simple fashion. First, all sentences were concatenated and split into tokens creating one huge list. For each iteration, a number n was randomly chosen from the interval $[8, 16]$ and a sequence of n tokens were added to the training data. This was repeated until the complete list of tokens was split into training samples where for most samples the last token does not correspond to a sentence boundary.
To use ArchiMob for the expanded training data setup, artificial SLU-boundaries for ArchiMob were introduced after the merging step, simply considering the last token of the merged utterances an SLU-boundary. Next, the End of Sentence Shifting step was applied (Archi Train in Figure 3.2). As apparent in Figure 3.2, the end of sentence shifting step was applied to one version of the test data used as input for the model, Test Shifted. This is an important detail of the experiment set up, as we aim to examine the impact of input shape on dependency parsing.

Name	'Sentences'	Tokens	Avg. Tokens per Sent.	**I**nside	**E**nd
SwissDial Train	8970	108816	12.13	99846	8970
SwissDial Dev	2243	27058	12.06	24816	2243

Table 3.10: Development and Test Set for SwissDial

Name	'Sentences'	Tokens	Avg. Tokens per Sent.	**I**nside	**E**nd
Archi Train	8970	78000	8.70	69030	8970
Archi Dev	2243	21244	9.47	19001	2243
Gold Test Set	203	2012	9.91	1809	203

Table 3.11: Train, Development and Test Set for ArchiMob

3.5 Data Statistics

Tables 3.10 and 3.11 show the number of 'sentences' and tokens used for the following experiments. For the sake of simplicity, everything was labeled 'sentences'. SwissDial is effectively composed out of complete, grammatical sentences, while ArchiMob data actually consists of preprocessed utterances (see Section 2.1.2). Hence, SwissDial served as the original training data from which 20% were put aside for validation during training. The expanded training set (SwissDial Train and Archi Train concatenated) contains 17'940 'sentences' and the development set 4'486 'sentences'. The average token number per 'sentence' is reported. This is to ensure that the inputs that are passed on to the model, be it for training or prediction, have roughly the same shape in terms of length.
Tables 3.11 and 3.10 show the statistics after preprocessing, respectively manual processing (i.e., the bottom row in Figure 3.2). For the original statistics refer to Section 2.1.2 or 2.1.3.

We tackle the problem of re-segmentation into SLUs as a token classification task. As a consequence, only the last token of an SLU will have the label **E**nd of SLU, whereas all other tokens receive **I**nside of SLU. Therefore, the label distribution is highly unbalanced, as reported in Tables 3.11 and 3.10. SwissDial Train has roughly 10 times more labels **I** than **E**. As for the average tokens per 'sentence', this factor is lower for ArchiMob, roughly 7 for Archi Train and 8 for Archi Dev. The Gold Test Set contains almost 9 times more labels **I** than *E*, ranging

between the rest of the ArchiMob data an SwissDial. For SwissDial and the Gold Test Set the number of labels **E** corresponds naturally to the number of SLUs. For ArchiMob data, the label count is based on the artificially created SLUs mentioned in the previous Section 3.4. For all the data processing steps, it was assured that they were applied to the labels accordingly; e.g., if a trigram was normalized and thus reduced from three to one tokens, the same had to be done with the labels.

Chapter 4

Experiments and Results

The task of SLU-boundary detection is modeled as a binary token classification task fine-tuning a Bert model (see Section 3.1). To each token from the manual test set (Gold Test Parsed) and the training and development sets (ArchiMob and SwissDial) a label was assigned, **I** for inside and **E** for end of an SLU.[1] Argumentation for the choice of training data, the data processing workflow and the data statistics were elaborated in the corresponding sections of the previous chapter.

The experiment setup includes two evaluation steps; firstly, one for the classification task and, secondly, one for dependency parsing. From a high level point of view, the experiments included training a model, passing the test set in two different forms (Test Merged and Test Shifted) to the model for prediction, evaluate the labeled output of the model against the labels of Gold Test Parsed, passing the model output to the parser pipeline, and finally, evaluate the output of the parser as well against Gold Test Parsed. Figure 4.1 shows this high-level view of the experiment setup. Names of the different data sets an colors refer back to the data processing shown in Figure 3.2; green represents training data, blue the data passed for prediction and testing, while yellow stands for the manually curated test set used for evaluation.
In the following, the different aspects of the setup are presented in detail. Section 4.1 elab-

[1] Four models were also implemented as a three class classification task, adding **B** for beginning of an SLU.

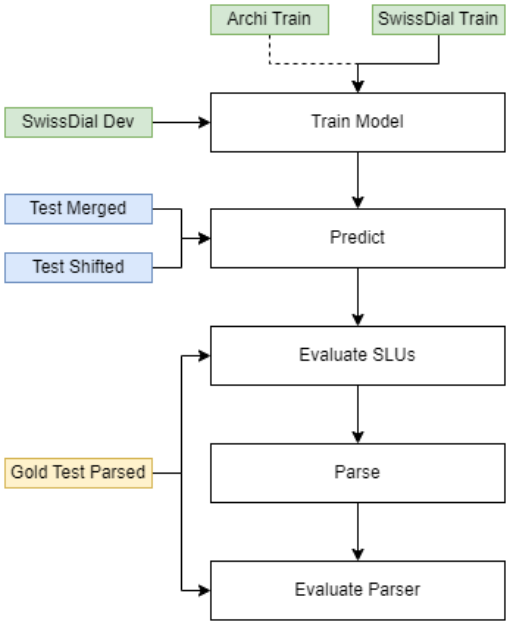

Figure 4.1: Experiment Setup

orates on the models and libraries used for the SLU-experiments. Furthermore, the different training settings, the evaluation setup and the chosen metrics are presented in detail. Section 4.2 introduces the same information for the dependency parsing experiments: the model (i.e., trained dependency parser), libraries and evaluation used for the second step of the experiments. Finally, Section 4.3 presents the results from the classification task and the parser evaluation, and studies the impact of re-segmenting the input into SLUs on dependency parsing.

4.1 SLU-Experiments

4.1.1 Pre-trained Models and Libraries

All models fine-tuned the pre-trained *dbmdz/bert-base-german-uncased*[2] (12-layer, 768-hidden, 12-heads, 110M parameters). This Bert model is a general, uncased language model for German trained on 16 GB of data (2'350'234'427 tokens) stemming from a Wikipedia dump, EU

[2]Available at: `https://huggingface.co/dbmdz/bert-base-german-uncased`[06.02.2023]

Bookshop corpus, Open Subtitles, CommonCrawl, ParaCrawl and News Crawl.

The *Simple Transformers* library[3] was used for integrating and fine-tuning the pre-trained models. The library acts as a wrapper for the *Transformers* library from *Hugging Face*[4], which itself wraps *keras* and *PyTorch*. Simple Transformers allows to train or fine-tune pre-trained models with a few clicks. As explained in Section 3.1, the experiments focuses on comparing different data and segmenting formats rather than models, thus no full control over all parameters of neural networks was required.

4.1.2 Training Settings and Hyperparameters

Training of the models was carried out on a 16GB Tesla T4 GPU card with CUDA 11.1[5]. All models were trained with PyTorch's *EarlyStopping* callback with a patience of three epochs[6]. Including the patience of three, the training time for the different models ranged from six up to twenty-four epochs, where most of the models ranged between six and ten epochs and only a few models surpassed ten epochs. For all the models, the learning rate was set to $4e-5$ and the AdamW optimizer was used with a linear scheduler with warm-up. Furthermore, the same random seed for parameter initialization was used for all models to ensure reproducibility and lower the effect of initialization.

As few parameters as possible were changed for the different models; only the validation metrics, the class weights and the training data were changed based on the results of the previous models. The default setting for training neural networks monitors the validation loss to track the learning. This was changed to monitoring the validation precision. Models that used precision for monitoring contain *Precision* in their name, if *Precision* does not appear in the name of the model, it was trained monitoring the validation loss. Cross Entropy[7] was used for

[3] `https://simpletransformers.ai/`[10.02.2023]
[4] `https://huggingface.co/`[10.02.2023]
[5] 8 cores 32 GB RAM
[6] $\delta : 0$
[7] See `https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html`[20.02.2023]

all models, no matter if used for monitoring or not, as the loss function. If we monitor the validation loss, we aim to minimize it; for precision we want to maximize it during training.

As elaborated in Section 3.5, the label distribution (**E** and **I**) in the data is highly unbalanced. Given the unbalanced class distribution in the data set, the weights were set to 0.05 for class **I** (Inside, no SLU-boundary) and 0.95 for **E** (SLU-boundary). The weights are based on slightly rounded values of the normalized inverse of the class distribution, i.e., $1-\frac{labelcount}{total}$, where the actual numbers would be 0.917 for **E** and 0.085 for **I**. Models using these weights are labeled with *Weighted*. To the authors best knowledge, the PyTorch documentation concerning weighted loss functions states only "weight (Tensor, optional) – a manual re-scaling weight given to each class. If given, has to be a Tensor of size C"[8] but does not elaborate on properties of such weights. Thus, a second set of weights was calculated for two other models. The second set used sklearn's *compute_class_weight*[9] yielding 7.7 for **E** and 0.5 for **I**. All implemented models using weights were trained with both sets of weights. Models using this second set of weights are simply referred to as *Weighted 2*.

Lastly, two models were trained with enhanced training data, adding data from ArchiMob (see Section 3.5), all other models used only SwissDial training data. All training data was passed to the models in a CoNLLU-like shape, i.e., each line consists of a token and a label, and sentences are separated by blank lines.

Table 4.1 shows all the eight training settings (or models). All models fine-tune the same pre-trained Bert model and are thus identical in terms of architecture (i.e., layers, attention mechanisms etc.). The different combinations of weights (*Weighted* or *Weighted 2*) and monitoring validation precision or loss add up to six models. Out of these models, two were additionally trained with the enhanced training data.

[8] See `https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html`[20.02.2023]

[9] `https://scikit-learn.org/stable/modules/generated/sklearn.utils.class_weight.compute_class_weight.html`[20.02.2023]

Model Name
Bert
Bert Weighted
Bert Precision
Bert Precision Weighted
Bert Weighted 2
Bert Precision Weighted 2
Bert Enhanced Training Data Weighted
Bert Enhanced Training Data Precision Weighted

Table 4.1: Different Models Overview

4.1.3 SLU-Evaluation

All models received two different test sets as inputs for prediction. Following the terminology introduced in Figure 3.2 in Section 3.4, the two test sets are called Test Merged and Test Shifted. For Test Merged the utterances were only merged as described in Section 3.4; Test Shifted was preprocessed identically as the SwissDial training set including the end of sentence shifting step. Predictions of the models were evaluated against the SLU-labels from the Gold Test Parsed. sklearn's *classification report*[10] was used for evaluation. Given the unbalanced label distribution, accuracy is not the best choice as a validation metric. For example, referring back to the label count reported in Figure 3.10 and 3.11, labeling everything as **I** in the SwissDial development set would still yield an accuracy of $\frac{24816}{27058} \approx 91\%$. For the manual gold test set, accuracy labeling everything **I** would be $\frac{1809}{2021} \approx 89\%$. Therefore, precision, together with F1-score, is considered to be the metric of choice. This is based on the fact that having precise SLU-boundaries is more important than catching all of them (recall). For example, missing an SLU-boundary and thus having two SLUs rendered as one should have less impact on the parser performance than having imprecise, arbitrarily distributed SLU-boundaries.

For the task of SLU-boundary-detection for transcribed Swiss German, no related work exists. Accordingly, no benchmark or baseline can be provided. Although, Rehbein et al. (2020) engaged in an almost identical task for Standard German, results should not be compared. The SLU-segmentation of their approach and ours are vastly different. They, furthermore, had access to training data coming from the same corpus as test data and could leverage more

[10] `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html`[20.02.2023]

prosodic features.

4.2 UD-Experiments

4.2.1 Pre-trained Models and Libraries

We use a pre-trained UDPipe parser that was, together with the dependency tag set, introduced in Section 3.1. Parsing was carried out using *spaCy*[11]. SpaCy provides a huge variety of pre-trained model based NLP-pipelines for many languages. SpaCy would include a dependency parser component, which uses its own set of dependency labels for German[12] that is not really related to any established dependency set like Universal Dependencies or The TiGer Dependency Bank. However, spaCy allows to add other components to the NLP pipeline. As elaborated in Section 3.2.2, this thesis uses universal dependency labels and the UDPipe parser. Thus, the *spacy-udpipe*[13] component was added. It is not clear which version and implementation of UDPipe spacy-udpipe uses, nor how the parser was trained. SpaCy, furthermore, does not output any specific data format. Therefore, a second component, *spacy_conll*[14], was added which allows to directly parse text into the CoNLLU-format. The German model, *german-gsd-ud-2.5-191206.udpipe*[15], provided by *spacy-udpipe* was used.

4.2.2 UD-Evaluation

As a first step for the UD-Evaluation, the predicted output of the models had to be reshaped. Using the predicted labels **E**, the output was reshaped such that each line contains exactly one SLU. This can easily be implemented by iterating over the tokens and labels and inserting

[11] `https://spacy.io/`[20.02.2023]

[12] See section 'PARSER': `https://spacy.io/models/de`[20.02.2023]

[13] SpaCy: `https://spacy.io/universe/project/spacy-udpipe`, source code: `https://github.com/TakeLab/spacy-udpipe`[20.02.2023]

[14] `https://spacy.io/universe/project/spacy-conll`[20.02.2023]

[15] `https://github.com/TakeLab/spacy-udpipe/blob/master/spacy_udpipe/resources/languages.json`[20.02.2023]

a line-break each time the current label is **E**. This reshaped output was passed line-wise to the spaCy-pipeline which parsed it using spacy-udpipe and outputted it in CoNLLU-format. Next, this CoNULL-file was evaluated against the manually annotated Gold Test Parsed. For evaluation, the official evaluation script from Universal Dependencies' tools was used[16].
While evaluation of a classification tasks uses commonly known metrics like precision, evaluation of dependency parsing requires different metrics. All of them are metrics commonly used in evaluation of dependency parsing and originate in the official CoNLL-U-tasks. Firstly, for all models a metric labeled 'Sentences' is displayed. According to the source code, it is a measure for "how well do the gold sentences match system sentences [=file passed for evaluation]"[17]. Secondly, the unlabeled attachment score (UAS) and the labeled attachment score (LAS) are reported. LAS expresses "how many words have been assigned both the correct syntactic head and the correct label"(Nivre and Fang, 2017, 89), while UAS only scores the correct labeling of the head and ignores the dependency label. Thirdly, the universal POS-tag score (UPOS) and the universal dependency tag score (UFeats) are reported. For these two metrics, precision, recall, F1-score and aligned accuracy are reported. Both metrics express how well the POS-tags or the dependency tags correspond with the manually annotated gold test set. The more fine-grained POS-tag score XPOS is not reported, as ArchiMob data contains some tags that do not belong to the standard Stuttgart-Tübingen-Tagset.

For the dependency parser evaluation, we could use state-of-the-art performance as a benchmark and or baseline (e.g., as presented in Section 2.3). On the one hand, performance of a dependency parser is highly dependent on the parser used, how it was trained and with which data, and which data is evaluated. On the other hand, we are interested in the impact of the SLU-re-segmentation on parser performance. Therefore, a more suitable baseline to study this impact is proposed. Table 4.2 displays the upper bound, the baseline and the performance of the *Senter* component from the spaCy model which was used as a wrapper for the UDPipe and CoNLLU component. For the upper bound, the manually split gold test data (Gold Test

[16] `https://github.com/UniversalDependencies/tools/blob/master/eval.py`[20.02.2023]
[17] Source code at: `https://github.com/UniversalDependencies/tools/blob/master/eval.py`[20.02.2023]

Model	Metric	Precision	Recall	F1 Score	Aligned Accuracy
Upper Bound	Sentences	100.0	100.0	100.0	-
	UAS	61.64	61.64	61.64	61.64
	LAS	44.88	44.88	44.88	44.88
	UPOS	80.94	80.94	80.94	80.94
	UFeats	97.35	97.35	97.35	97.35
Baseline	Sentences	41.35	42.36	41.85	-
	UAS	55.9	55.9	55.9	55.9
	LAS	40.86	40.86	40.86	40.86
	UPOS	80.3	80.3	80.3	80.3
	UFeats	93.14	93.14	93.14	93.14
Spacy Senter	Sentences	2.94	0.49	0.84	-
	UAS	44.24	44.24	44.24	44.24
	LAS	35.96	35.96	35.96	35.96
	UPOS	79.42	79.42	79.42	79.42
	UFeats	84.27	84.27	84.27	84.27

Table 4.2: Baseline and Upper Bound

Parsed in Figure 3.2) was passed on to the UDPipe parser integrated into spaCy and evaluated against the manually annotated gold test data. The 'Sentences' metric shows that the Input was already in the correct shape. UFeats is not 100% as some dependency tags were set incorrectly by the parser (especially tags *aux*, *cop*, *advmod*, and *nmod*). The baseline was created by passing Test Merged (see Figure 3.2) to the parser. The spaCy Senter baseline was created as follows: the whole test set was passed as one string to the spaCy Senter component which split the input into 'Sentences'. This output was parsed with UDPipe and evaluated. The purpose of this baseline is to have a benchmark to compare the models developed for the experiments with already available systems performing SLU-boundary-detection. However, to the best knowledge of the author, it is not clear if the spaCy Senter merely tries to disambiguate punctuation or if it applies a different approach. The spaCy Senter baseline has a very low score for the Sentence metric, showing that the sentences do not match with the SLU-structure of the manually split gold test data. The baseline, colored in grey, serves as the threshold for acceptable performance, everything above these values can be considered an improvement. Finally, the green values represent the upper bound. Any model surpassing these values re-shapes the input in such a way that it outperforms a human re-segmenting the data. Therefore, we hope to find performances between the grey and the green values to be able to conclude that the models do

Model	Precision Merged	F1 Score Merged	Precision Shifted	F1 Score Shifted
Bert	73.18	69.49	72.54	**71.42**
Bert Weighted	67.21	69.10	71.22	**74.04**
Bert Precision	65.71	60.8	74.32	**68.19**
Bert Precision Weighted	74.09	**73.71**	71.96	70.98
Bert Weighted 2	63.55	**63.96**	62.74	62.13
Bert Precision Weighted 2	69.7	67.19	71.82	**70.15**
Bert Enhanced Training Data Weighted	68.16	71.07	70.41	**73.72**
Bert Enhanced Training Data Precision Weighted	70.12	67.24	70.4	**68.37**

Table 4.3: Comparison Classification Results for All Models and Both Inputs in %

indeed improve the parser performance by reshaping the input data[18]. It is important to have a close look at the exact numbers. For LAS we have a range of roughly 4% separating the lower (merged) and the upper bound. For 'Sentences' it would be 60%. At the same time, however, this already implies that an improvement in the 'Sentence' metric would probably only improve the LAS marginally, given that an improvement of 60% corresponds to a LAS-improvement of only 4%. Similarly, we see that the range of improvement is rather small for the other two metrics as well, approximately 4% for UFeats and not even 1% for UPOS. This is of utter importance as a small improvement in numbers might represent a good improvement relative to the baseline. Compared to parsing properly structured data, the LAS achieved by the upper bound (manual segmentation) is 'low'. Section 3.2.2, however, explained and exemplified in detail, why parsing spoken Swiss German is difficult and performance is expected to be 'low'.

4.3 Results

Before diving into the numbers, a glance back at Figure 4.1 might be a helpful refresher for the experiment setup. Generally, the complete evaluation tables with all metrics introduced for both evaluations can be found in the appendix; tables within this section report precision and

[18]In fact, the baseline table already demonstrates that the shape of the input does have an impact on the parser performance and that more sentence-like input,i.e., the manual re-segmentation, does indeed yield better performance.

F1 for classification, and F1 and LAS for dependency parsing. For UPOS and UFeats results, refer to the appendix.

The first Table 4.3 displays the comparison of precision and F1-scores for all eight models and both test inputs (Test Merged, Test Shifted). Recall, however, is implicitly reported via the F1-score. For example, the simple binary *Bert* model (first row in Table 4.3) has a higher precision for Test Merged but the F1-score is higher for Test Shifted. This implies that the recall for Test Shifted is higher than for Merged.

The overall best performing model in terms of F1-score is *Bert Weighted* (marked in bold) with an F1-score of 74.04%, i.e., the binary Bert model fine-tuned with weights (**I**: 0.05, **E**: 0.95) and monitoring validation loss (Reported in appendix: accuracy 88.92%; recall: 78.53%). In terms of accuracy, the same model with Test Merged ranks on the sixth place. This indicates that models with a higher accuracy tend to label more tokens as **I**, thus increasing accuracy but lowering precision and recall; and confirms again that accuracy is not an insightful metric in this setting. Only considering Input Merged, the best performance was achieved by *Bert Precision Weighted* with an F1-score of 73.71%, precision of 74.09% (Reported in appendix 6: accuracy 90.11%; recall: 73.35%,). This model ranked accuracy-wise on the first place together with *Bert Precision*. *Bert Enhanced Training Data Weighted* ranked overall on the second place. This model was trained with the same amount of SwissDial and ArchiMob data. This indicates that presenting more data similar to the test set in terms of structure and average tokens per line did not improve the performance.

Table 4.4 displays the two most important metrics, 'Sentence' and LAS, for parser evaluation for the parsed predictions of all models and both inputs. As before, the model of which the predictions lead to the best parser performance is marked in bold and for each model the better result from both inputs is rendered in bold as well. The predictions of *Bert Precision Weighted* yielded the best performance with an LAS of 43.02% with Test Merged achieving almost a value 2% higher than the baseline (See baseline in Table 4.2). This is a solid improvement as LAS between the baseline and the upper bound is only 4%.

The predictions of this model score as well the second highest value for the 'Sentence' F1-score,

Model	Metric	F1 Merged	F1 Shifted
Bert	Sentences	10.92	**13.47**
	LAS	40.57	**40.96**
Bert Weighted	Sentences	12.74	**16.33**
	LAS	**42.63**	42.23
Bert Precision	Sentences	4.5	**12.96**
	LAS	**39.54**	39.39
Bert Precision Weighted	Sentences	**18.55**	16.49
	LAS	**43.02**	40.18
Bert Weighted 2	Sentences	**9.08**	7.92
	LAS	**36.7**	34.5
Bert Precision Weighted 2	Sentences	9.5	**14.36**
	LAS	40.13	**40.86**
Bert Enhanced Training Data Weighted	Sentences	15.0	**18.85**
	LAS	41.74	**42.14**
Bert Enhanced Training Data Precision Weighted	Sentences	9.07	**10.87**
	LAS	41.6	**42.04**

Table 4.4: Comparison Parser Evaluation for All Models and Both Inputs in %

meaning that it also matches better with the sentences from Gold Test Parsed. This model (*Bert Precision Weighted*) is not the best model considering the classification evaluation but ranks on the third place, roughly 0.3% behind the first place and only 0.01% behind the second place (F1-score). It is, however, the best performing model in classification for Test Merged. The best performing model for classification, *Bert Weighted*, is the second best in parser evaluation with a LAS roughly 0.4% lower. Although for both evaluations, not the same model performs best, we still see that the top performing models are roughly the same for both evaluations having similar differences in scores.

Furthermore Table 4.4 shows all the models of which the predictions achieved a performance above the baseline in grey. In general, we can observe that F1-score for Sentences is lower than the baseline ($\approx 41\%$) but performances in terms of LAS are all close to the baseline. five out of eight models outperformed the baseline. *Bert Precision Weighted 2* scored the same LAS as the baseline. We can observe, furthermore, that the two models, *Bert Precision* and *Bert Weighted 2*, which do not outperform the baseline, rank on the last places in the classification task. At this point, we can conclude that models trained in a classification task can help to improve parser performance. Every model except *Bert Weighted* with Test Shifted also outperformed the spaCy Senter baseline which scored a LAS of 35.96%. We can conclude, furthermore, that

our models outperform available NLP components for SLU-boundary-detection[19].

As stated in the introduction 1, we are interested in inquiring if re-shaping input into SLUs does improve dependency parsing which we can confirm at this point. We still want, however, to examine the results more closely. Of special interest is the connection between performance in classification and dependency parsing.

Figure 4.2 displays F1-score from the classification task on the x-axis and LAS from parser evaluation on the y-axis. The black solid line represents the regression lines. For Merged it is $y = 0.13x + 32.3$ and for Shifted $y = 0.12x + 32.51$ (rounded values). This scatter plot shows a connection between F1-score and LAS, i.e., between classification and parser performance. Both regression lines have a positive and almost identical slope. The trend is more apparent for Test Merged. We calculated the Pearson product-moment correlation coefficients[20]: Test Merged: 0.3953 and Test Shifted: 0.3090. These values solidify that the correlation is higher for Test Merged, but that both inputs exhibit the trend. We can conclude that F1-score from classification and LAS do correlate but not too strongly.

These results confirmed our research question stating that re-shaping the input into SLUs does positively impact the performance of dependency parsing. It was shown that performance in classification and dependency parsing do correlate. There are, however, further insights to be gained from the results just presented.

4.3.1 Further Analysis

As elaborated in Section 4.1, we aimed to increase precision by using it as a validation metric instead of validation loss. Accordingly, we want to examine if we managed to do so and if a higher precision also leads to a higher F1-score. Comparing the different metrics (precision and F1-score) in Table 4.3, we do not see a clear pattern: Nor do models trained monitoring the precision display a consequently higher precision than their counterpart trained monitoring the validation loss, nor does Test Shifted yield consequently higher values for precision than with Test Merged although the F1-scores tend to be higher. For example, *Bert* has a higher precision

[19] SpaCy's senter uses the term 'sentence' and claims to identify sentence boundaries.

[20] with *numpy*

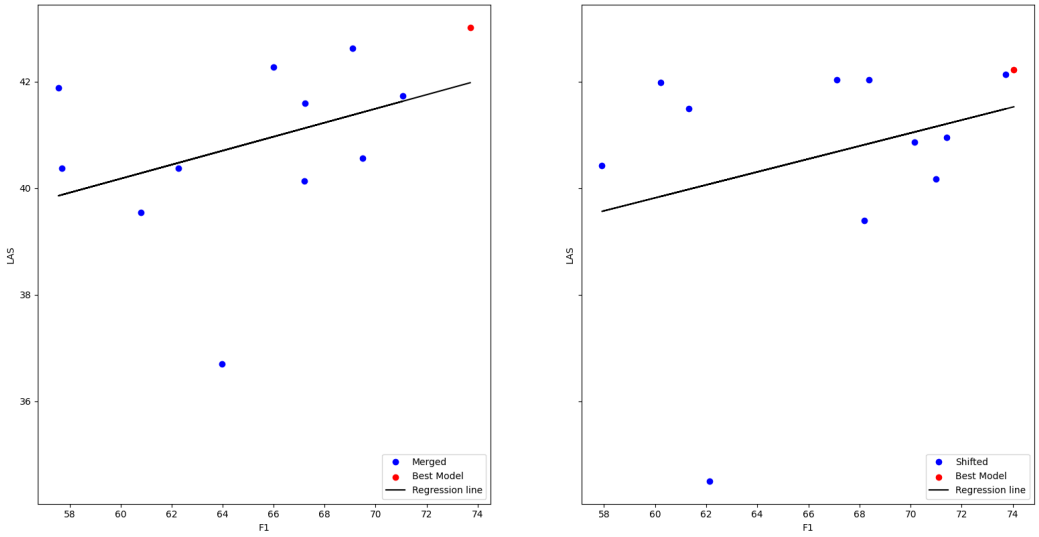

Figure 4.2: Scatter Plot: F1-Score and LAS

with Test Merged but a higher F1-score with Test Shifted. *Bert* with Test Merged has a higher precision than *Bert Precision* with the same input. This comes rather as a surprise as the only difference between the two models is exactly monitoring the precision which was an attempt to enforce higher values for precision. Furthermore, the F1-score follows the same pattern, i.e., it is higher for *Bert* than for *Bert Precision*. More confusingly, the same models show a different pattern with Test Shifted. *Bert* has a lower precision but a higher F1-score, *Bert Precision* has a higher precision but a lower F1-score. If we compare other pairs of models that are only distinguished by monitoring precision or not, we do not see the same pattern as just elaborated. *Bert Enhanced Training Data Weighted* was outperformed by its precision counterpart for Test Merged but not for Test Shifted, but always outperformed its precision counterpart in terms of F1-score no matter the test input. We can conclude that enforcing training on precision does not necessarily yield a higher precision and does not automatically produce a better result in terms of F1.

Two different sets of class weights were used as well. A quick look at Table 4.3 discloses that every *Weighted 2* model (weights calculated with sklearn. **E**: 7.7; **I** 0.5) was outperformed by its *Weighted* counterpart (manually calculated weights: **E**: 0.95; **I**: 0.05). All Weighted models score better in terms of Precision and F1-score for both inputs.

Model	Precision Merged	F1 Score Merged	Precision Shifted	F1 Score Shifted
Bert (t)	59.82	57.55	61.53	**60.21**
Bert Weighted (t)	63.14	65.99	63.90	**67.10**
Bert Precision (t)	64.65	57.70	64.65	**57.92**
Bert Precision Weighted (t)	65.48	**62.25**	64.55	61.33

Table 4.5: Comparison Classification Results for Three-Class Models and Both Inputs in %

Some small experiments were carried out during training raising the patience and trying to overcome a possible local minimum in the loss function. However, all models were only trained by as many epochs more as the patience was raised. This indicates that either it is the global minimum (with the given hyperparameters) or more than ten epochs would be required to escape the local minimum.

Before engaging in a binary classification task, a three-class-classification approach was tackled. For this purpose, the first token of an SLU was additionally marked with a special label **B** for beginning of SLU. Classification results from this setup are shown in Table 4.5, as well as the results from the parser evaluation in Table 4.6. The naming convention of these models is identical to the binary models and indicates variation in the training setup. Almost all three-class models were outperformed by the binary ones. Only the best performing three-class model, *Bert Weighted (t)*, outperformed the worst binary model. It is interesting that for binary and three-class models the Weighted training setting led to the highest performance. An analysis of the predicted labels revealed that all three class models failed to learn that the number of **B** is always identical to **E**. All models predicted more labels **B** than **E**. A binary approach seems more appropriate to tackle the problem of SLU-boundary-detection. Table 4.6 shows that the re-shaped output of the three-class models does improve parser performance and three out of four models achieved a LAS above the baseline. This confirms once again that re-shaping the input into SLUs does positively impact parser performance.

All experiments were conducted with two different inputs, while all other parameters were identical. As a motivation, we aimed to study if the utterance structure from ArchiMob (Test Merged) performs better than a 'random' segmentation (Test Shifted). Looking at the

Model	Metric	F1 Merged	F1 Shifted
Bert (t)	Sentences	9.72	**12.09**
	LAS	41.89	**41.99**
Bert Weighted (t)	Sentences	13.46	**17.26**
	LAS	**42.28**	42.04
Bert Precision (t)	Sentences	6.19	**9.85**
	LAS	40.37	**40.42**
Bert Precision Weighted (t)	Sentences	8.31	**15.77**
	LAS	40.37	**41.5**

Table 4.6: Comparison Parser Evaluation for Three-Class Models and Both Inputs in %

classification results in Table 4.3 (binary) and 4.5 (three-class-classification), we observe that classification performed better with Test Shifted (nine out of twelve models). The 203 SLUs from the test set are split into 208 lines for Test Merged and 254 lines for Test Shifted. Test Shifted, thus, contains less tokens per line on average. We might speculate that Test Shifted leads to better results as Test Shifted (and the ArchiMob data in general) has less tokens per SLU (or line) than the SwissDial data and is therefore more similar to ArchiMob data.

We can observe the same tendency for parser evaluation in Tables 4.4 (binary predictions) and 4.6 (three-class-classification predictions). Parsed predictions from Test Shifted tend to yield better results. Further experiments and examinations are required to confirm this tendency. Our results indicate that the predictions that achieve higher scores in classification (Test Shifted), do so in parsing as well.

As a last step, we analysed the connection between the number of predicted SLU-boundaries (or labels **E**) and F1-score from the classification task. The number of labels **E** are reported in the extended Tables 6.5, 6.6, 6.7 and 6.8 in the appendix. One model, *Bert Enhanced Training Data Weighted* predicted high numbers of **E** for both inputs (Merged: 656; Shifted: 706), whereas all other models predicted a label count of **E** between 100 and 300. Therefore, we excluded F1-score and label count of this model as outliers for the analysis. The scatter plot in Figure 4.3 shows the distribution with count **E** on the x-axis and F1-score on the y-axis. Additionally, the best performing model is marked and the regression line[21] is displayed. As for the analysis of LAS-F1, we calculated the Pearson product-moment correlation coefficients: Test Merged 0.5852; Test Shifted 0.5113. Correlation coefficients and slops of the regression

[21] Merged: $y = 0.0418x + 5.73$; Shifted: $y = 0.0459x + 5.84$

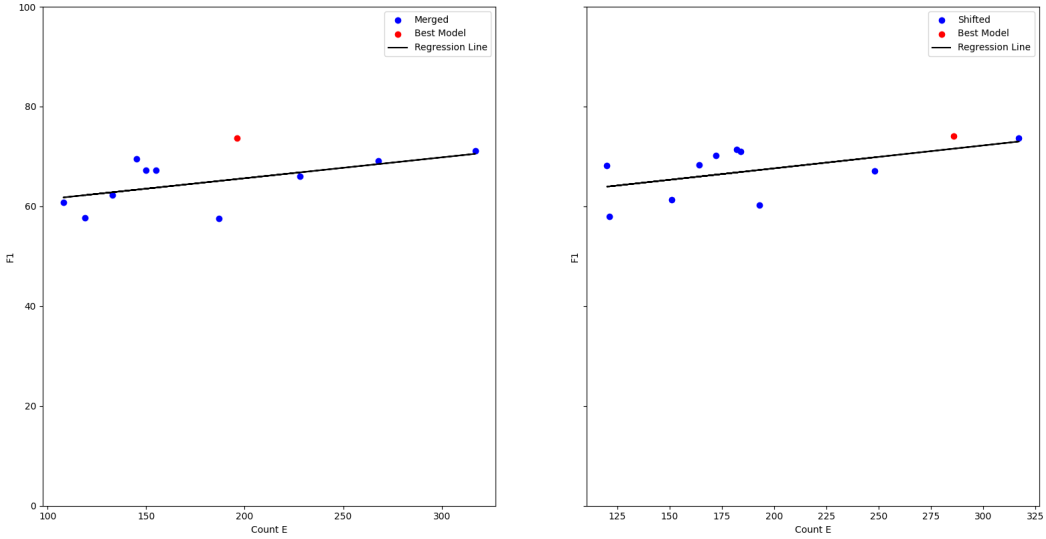

Figure 4.3: Scatter Plot: Count **E** and F1-Score

lines are almost identical and demonstrate the connection. We can conclude that up to a label count of roughly 300 labels **E** the performance in terms of F1-score and the number of predicted labels do correlate.

Our further analysis identified that enforcing precision during training did not yield the desired results, nor could we force a longer training by raising the patience of early stopping. Our results demonstrated clearly that our manually calculated weights outperformed the ones from sklearn (Weighted 2) and that binary classification models yield better predictions than three-class models. We examined if Test Merged or Test Shifted leads to better results, although there is a tendency for Test Shifted, further experiments are necessary to confirm this. Lastly, the connection between number of predicted labels **E** and F1-score could be shown up to roughly 300 labels **E**.

Chapter 5

Conclusion

5.1 Summary of Thesis Achievements

Dependency parsing, one one variant of syntactical parsing, is challenging for transcribed spoken language. Spoken language is unstructured and differs from the properly structured data that is usually used to train dependency parsers. Spoken language contains a lot of disfluencies, repetitions and truncated or interrupted sentences. This is further complicated by the fact that we worked with Swiss German audio transcripts. Swiss German, an umbrella term for the German variants spoken in Switzerland, lacks standardization and exhibits a high degree of variation in vocabulary, spelling, morphology and syntax. Swiss German is, furthermore, a low-resource language for which little data exists, and existing data is usually not normalized and thus full of variation.

We tackled the problem of dependency parsing for Swiss German audio transcripts by re-segmenting the unstructured transcripts into SLUs (sentence-like units) and studied the effect of this re-segmentation on dependency parser performance. Our data comes from ArchiMob (Section 2.1.2), a corpus containing transcribed Swiss German interviews with elderly native speakers. Existing algorithms for sentence segmentation leverage punctuation or require data already providing some sort of a sentence structure in the data - which ArchiMob transcripts both lack. Thus, we framed the problem of re-segmentation as a binary token classification task

and fine-tuned a pre-trained German Bert model to predict tokens with label **E** marking the **E**nd of an SLU (Chapter 3). Our experiment setup included fine-tuning a model, predicting labels for two different test inputs, re-shaping these inputs into SLUs using the predicted labels and passing them to a dependency parser. We thus included two evaluation steps, one for token classification and one for dependency parsing. As availability and quality of data is an issue for Swiss German, no test set was available and we created manually a test set composed of roughly 200 SLUs and manually annotated the dependency labels. Our setup is supervised and requires training data that contains a form of SLU-segmentation and normalization; we used data from SwissDial (Section 2.1.3) as out-of-domain training data.

In this last chapter, we presented the results and studied the impact of the re-segmentation on parser performance in detailed. The re-segmentation leveraging the predictions from our models improved the parser performance over a baseline. The baseline was created using one of the inputs that resembles the original utterance structure from ArchiMob and that was also presented to the models for prediction. The re-shaped predictions from the best model scored a LAS 2% over the baseline compared to only 4% separating the baseline and performance on manually re-segmented data (Gold Test Parsed). Our experiments demonstrated that re-shaping the input into SLUs does improve the performance of the dependency parser in terms of LAS. Moreover, the induced SLU-structure performs better than the original utterance structure represented by the baseline. We showed, furthermore, that performance in classification correlates with performance in dependency parsing.

With our work, we contribute to the problem of syntactical parsing for spoken language and to processing spoken Swiss German. As Swiss German is mainly an oral language, we consider our work an important step for bringing (transcribed) spoken Swiss German into a more structured written form. As to our best knowledge, we are the first to tackle this problem for transcribed Swiss German, we provide, furthermore, a point of reference for future work.

5.2 Future Work

Possibilities for future work taking this research as a starting point are numerous. Firstly, of course, it remains to demonstrate that results found in the experiments do apply to other parsers as well. Secondly, it would be interesting to study if dependency parsing for other low-resource languages can be improved with the same approach presented here. Furthermore, fine-tuning the proposed models and extensive experiments with hyperparameters might enhance the improvement in performance achieved.

Other possibilities revolve around examining the results presented here further. Our experiments used after all only normalized Swiss German and Standard German data. It remains, therefore, an open question if an application of the same approach using not normalized Swiss German from different dialects yields the same results and insights. Aepli and Clematide (2018) provided a dependency parser for Swiss German that can be used to reproduce our research for purely Swiss German text. It would also be interesting to study if the models trained on Standard German data in this thesis exhibit differences in performance for different dialects. A good start for this might be an in detail analysis of the SLU-segmentation of the test sets predicted by the models trying to identify certain linguistic patterns and situations where the algorithm fails to predict the correct SLU-boundary.

Chapter 6

Appendix

- Code and test set available on request on: `https://github.com/PaprikaSteiger/CHunking`

Model	Metric	Precision	Recall	F1 Score	Aligned Accuracy
Bert	Sentences	13.1	9.36	10.92	-
	UAS	52.38	52.38	52.38	52.38
	LAS	40.57	40.57	40.57	40.57
	UPOS	80.01	80.01	80.01	80.01
	UFeats	87.95	87.95	87.95	87.95
Bert Weighted	Sentences	11.19	14.78	12.74	-
	UAS	57.13	57.13	57.13	57.13
	LAS	42.63	42.63	42.63	42.63
	UPOS	80.01	80.01	80.01	80.01
	UFeats	86.97	86.97	86.97	86.97
Bert Precision	Sentences	6.48	3.45	4.5	-
	UAS	49.93	49.93	49.93	49.93
	LAS	39.54	39.54	39.54	39.54
	UPOS	80.21	80.21	80.21	80.21
	UFeats	85.25	85.25	85.25	85.25
Bert Precision Weighted	Sentences	18.88	18.23	18.55	-
	UAS	56.49	56.49	56.49	56.49
	LAS	43.02	43.02	43.02	43.02
	UPOS	80.21	80.21	80.21	80.21
	UFeats	88.63	88.63	88.63	88.63
Bert Weighted 2	Sentences	5.95	19.21	9.08	-
	UAS	52.33	52.33	52.33	52.33
	LAS	36.7	36.7	36.7	36.7
	UPOS	78.15	78.15	78.15	78.15
	UFeats	83.0	83.0	83.0	83.0
Bert Precision Weighted 2	Sentences	10.97	8.37	9.5	-
	UAS	51.59	51.59	51.59	51.59
	LAS	40.13	40.13	40.13	40.13
	UPOS	80.21	80.21	80.21	80.21
	UFeats	87.07	87.07	87.07	87.07
Bert Enhanced Training Data Weighted	Sentences	12.3	19.21	15.0	-
	UAS	57.52	57.52	57.52	57.52
	LAS	41.74	41.74	41.74	41.74
	UPOS	79.42	79.42	79.42	79.42
	UFeats	86.77	86.77	86.77	86.77
Bert Enhanced Training Data Precision Weighted	Sentences	10.67	7.88	9.07	-
	UAS	53.6	53.6	53.6	53.6
	LAS	41.6	41.6	41.6	41.6
	UPOS	80.16	80.16	80.16	80.16
	UFeats	88.14	88.14	88.14	88.14

Table 6.1: Parser Evaluation Binary Merged

Model	Metric	Precision	Recall	F1 Score	Aligned Accuracy
Bert	Sentences	14.21	12.81	13.47	-
	UAS	54.58	54.58	54.58	54.58
	LAS	40.96	40.96	40.96	40.96
	UPOS	80.25	80.25	80.25	80.25
	UFeats	87.21	87.21	87.21	87.21
Bert Weighted	Sentences	13.94	19.7	16.33	-
	UAS	57.47	57.47	57.47	57.47
	LAS	42.23	42.23	42.23	42.23
	UPOS	80.3	80.3	80.3	80.3
	UFeats	87.9	87.9	87.9	87.9
Bert Precision	Sentences	17.36	10.34	12.96	-
	UAS	51.98	51.98	51.98	51.98
	LAS	39.39	39.39	39.39	39.39
	UPOS	80.45	80.45	80.45	80.45
	UFeats	87.41	87.41	87.41	87.41
Bert Precision Weighted	Sentences	17.3	15.76	16.49	-
	UAS	53.31	53.31	53.31	53.31
	LAS	40.18	40.18	40.18	40.18
	UPOS	80.25	80.25	80.25	80.25
	UFeats	87.85	87.85	87.85	87.85
Bert Weighted 2	Sentences	5.1	17.73	7.92	-
	UAS	49.26	49.24	49.25	49.29
	LAS	34.51	34.49	34.5	34.53
	UPOS	77.5	77.46	77.48	77.54
	UFeats	81.67	81.63	81.65	81.71
Bert Precision Weighted 2	Sentences	15.61	13.3	14.36	-
	UAS	54.09	54.09	54.09	54.09
	LAS	40.86	40.86	40.86	40.86
	UPOS	80.5	80.5	80.5	80.5
	UFeats	88.24	88.24	88.24	88.24
Bert Enhanced Training Data Weighted	Sentences	15.46	24.14	18.85	-
	UAS	57.91	57.91	57.91	57.91
	LAS	42.14	42.14	42.14	42.14
	UPOS	79.62	79.62	79.62	79.62
	UFeats	87.51	87.51	87.51	87.51
Bert Enhanced Training Data Precision Weighted	Sentences	12.12	9.85	10.87	-
	UAS	54.58	54.58	54.58	54.58
	LAS	42.04	42.04	42.04	42.04
	UPOS	80.11	80.11	80.11	80.11
	UFeats	87.65	87.65	87.65	87.65

Table 6.2: Parser Evaluation Binary Shifted

Model	Metric	Precision	Recall	F1 Score	Aligned Accuracy
Bert (t)	Sentences	10.11	9.36	9.72	-
	UAS	55.12	55.12	55.12	55.12
	LAS	41.89	41.89	41.89	41.89
	UPOS	80.16	80.16	80.16	80.16
	UFeats	86.72	86.72	86.72	86.72
Bert Weighted (t)	Sentences	12.72	14.29	13.46	-
	UAS	56.34	56.34	56.34	56.34
	LAS	42.28	42.28	42.28	42.28
	UPOS	80.06	80.06	80.06	80.06
	UFeats	86.87	86.87	86.87	86.87
Bert Precision (t)	Sentences	8.33	4.93	6.19	-
	UAS	51.84	51.84	51.84	51.84
	LAS	40.37	40.37	40.37	40.37
	UPOS	80.16	80.16	80.16	80.16
	UFeats	86.23	86.23	86.23	86.23
Bert Precision Weighted (t)	Sentences	10.45	6.9	8.31	-
	UAS	52.23	52.23	52.23	52.23
	LAS	40.37	40.37	40.37	40.37
	UPOS	80.45	80.45	80.45	80.45
	UFeats	86.82	86.82	86.82	86.82

Table 6.3: Parser Evaluation Three Class Models Merged

Model	Metric	Precision	Recall	F1 Score	Aligned Accuracy
Bert (t)	Sentences	12.37	11.82	12.09	-
	UAS	55.66	55.66	55.66	55.66
	LAS	41.99	41.99	41.99	41.99
	UPOS	79.96	79.96	79.96	79.96
	UFeats	87.11	87.11	87.11	87.11
Bert Weighted (t)	Sentences	15.66	19.21	17.26	-
	UAS	56.98	56.98	56.98	56.98
	LAS	42.04	42.04	42.04	42.04
	UPOS	80.16	80.16	80.16	80.16
	UFeats	87.85	87.85	87.85	87.85
Bert Precision (t)	Sentences	13.11	7.88	9.85	-
	UAS	52.43	52.43	52.43	52.43
	LAS	40.42	40.42	40.42	40.42
	UPOS	80.4	80.4	80.4	80.4
	UFeats	86.87	86.87	86.87	86.87
Bert Precision Weighted (t)	Sentences	18.42	13.79	15.77	-
	UAS	54.04	54.04	54.04	54.04
	LAS	41.5	41.5	41.5	41.5
	UPOS	80.94	80.94	80.94	80.94
	UFeats	87.85	87.85	87.85	87.85

Table 6.4: Parser Evaluation Three Class Models Shifted

Model Name	Accuracy	Precision	Recall	F1	Count **E**
Bert (6E)	79.52	59.82	55.87	57.55	187
Bert weighted (5E)	80.86	63.14	70.35	65.99	228
Bert precision (12E)	81.46	64.65	53.96	57.7	119
Bert precision weighted (19E)	82.41	65.48	60.23	62.25	133

Table 6.5: Three Class Classification Results, Merged

Model Name	Accuracy	Precision	Recall	F1	Count **E**
Bert (6E)	80.08	61.53	59.09	60.21	174
Bert weighted (5E)	80.77	63.91	71.88	67.1	248
Bert precision (12E)	81.41	64.65	54.22	57.92	121
Bert precision weighted (19E)	81.51	64.55	59.0	61.33	151

Table 6.6: Three Class Classification Results, Shifted

Model Name	Accuracy	Precision	Recall	F1	Count **E**
Bert	90.36	73.18	67.09	69.49	145
Bert weighted	87.23	67.21	71.9	69.1	268
Bert precision	88.82	65.71	58.79	60.8	108
Bert precision weighted	90.61	74.09	73.35	73.71	196
Bert weighted 2	75.8	63.55	82.82	63.96	317
Bert precision weighted 2	89.36	69.7	65.44	67.19	150
Bert enhanced training data weighted (6E)	86.98	68.16	76.58	71.07	656
Bert enhanced training data precision weighted (24E)	89.51	70.12	65.3	67.24	155

Table 6.7: Binary Classification Results, Merged

Model Name	Accuracy	Precision	Recall	F1	Count **E**
Bert	90.11	72.54	70.45	71.42	182
Bert weighted	88.92	71.22	78.53	74.04	286
Bert precision	90.61	74.32	65.04	68.19	120
Bert precision weighted	89.91	71.96	70.12	70.98	184
Bert weighted 2	73.51	62.74	81.99	62.13	317
Bert precision weighted 2	89.91	71.82	68.81	70.15	164
Bert enhanced training data weighted (6E)	88.17	70.41	79.86	73.72	706
Bert enhanced training data precision weighted (24E)	89.51	70.4	66.83	68.37	172

Table 6.8: Binary Classification Results, Shifted

Bibliography

N. Aepli. Parsing approaches for swiss german. Master's thesis, Januar 2018. URL https://doi.org/10.5167/uzh-174602.

N. Aepli and S. Clematide. Parsing approaches for swiss german. In *Swiss Text Analytics Conference*, 2018.

C. R. F. d. Andrade and V. d. O. Martins. Speech fluency variation in elderly. *Pro. Fono.*, 22 (1):13–18, Mar. 2010.

M. Averintseva-Klisch. *Textkohärenz*. Kurze Einführungen in die germanistische Linguistik Band 14. Universitätsverlag Winter GmbH, Heidelberg, 2 edition, 2018. ISBN 9783825379780.

W. Ayres-Bennett and J. Bellamy, editors. Cambridge Handbooks in Language and Linguistics. Cambridge University Press, 2021.

Bundesamt für Statistik. *Statistik der Schweizer Städte 2022*. Number 22364335. Bern / Neuchâtel, April 2022a. ISBN 978-3-303-00689-4. URL https://dam-api.bfs.admin.ch/hub/api/dam/assets/22364335/master.

Bundesamt für Statistik. Sprachen, 2022b. URL https://www.bfs.admin.ch/bfs/de/home/statistiken/bevoelkerung/sprachen-religionen/sprachen.html.

Ç. Çöltekin, B. Campbell, E. Hinrichs, and H. Telljohann. Converting the TüBa-D/Z treebank of German to Universal Dependencies. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 27–37, Gothenburg, Sweden, May 2017. Association for Computational Linguistics. URL https://aclanthology.org/W17-0404.

D. de Kok and T. Pütz. Self-distillation for german and dutch dependency parsing. *Computational Linguistics in the Netherlands Journal*, 10:91–107, Dec. 2020. URL https://clinjournal.org/clinj/article/view/106.

M.-C. de Marneffe, C. D. Manning, J. Nivre, and D. Zeman. Universal Dependencies. *Computational Linguistics*, 47(2):255–308, 07 2021. ISSN 0891-2017. doi: 10.1162/coli_a_00402. URL https://doi.org/10.1162/coli_a_00402.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

E. Dieth and C. Schmid-Cadalbert. *Schwyzertütschi Dialäktschrift: Dieth-Schreibung*. Lebendige Mundart. Sauerländer, 1986. ISBN 9783794128327. URL https://books.google.ch/books?id=eTEeAQAAIAAJ.

K. Dobrovoljc. Spoken language treebanks in Universal Dependencies: an overview. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1798–1806, Marseille, France, June 2022. European Language Resources Association. URL https://aclanthology.org/2022.lrec-1.191.

P. Dogan-Schönberger, J. Mäder, and T. Hofmann. Swissdial: Parallel multidialectal corpus of spoken swiss german. *CoRR*, abs/2103.11401, 2021. URL https://arxiv.org/abs/2103.11401.

C. M. Downey, S. Drizin, L. Haroutunian, and S. Thukral. Multilingual unsupervised sequence segmentation transfers to extremely low-resource languages. *CoRR*, abs/2110.08415, 2021. URL https://arxiv.org/abs/2110.08415.

P. Eisenberg. *Duden - die Grammatik : unentbehrlich für richtiges Deutsch*. Dudenverlag, Mannheim, 8 edition, 2009. ISBN 978-3-411-04047-6.

K. Evang, V. Basile, G. Chrupała, and J. Bos. Elephant: Sequence labeling for word and sentence segmentation. *EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pages 1422–1426, 01 2013.

M. Forst, N. Bertomeu, B. Crysmann, F. Fouvry, S. Hansen-Schirra, and V. Kordoni. Towards a dependency-based gold standard for German parsers. the TIGER dependency bank. In *Proceedings of the 5th International Workshop on Linguistically Interpreted Corpora*, pages 31–38, Geneva, Switzerland, aug 29 2004. COLING. URL https://aclanthology.org/W04-1905.

M. Gasser. Bärenstarkes berndeutsch – warum eigentlich?, 2020. URL https://www.srf.ch/radio-srf-1/radio-srf-1/mundart/beliebtester-dialekt-baerenstarkes-berndeutsch-warum-eigentlich.

J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. Convolutional sequence to sequence learning. *CoRR*, abs/1705.03122, 2017. URL http://arxiv.org/abs/1705.03122.

I. Glaser, S. Moser, and F. Matthes. Sentence boundary detection in german legal documents. In *ICAART*, 2021.

P. Hoffman, E. Loginova, and A. Russell. Poor coherence in older people's speech is explained by impaired semantic and executive processes. *eLife*, 7:e38907, sep 2018. ISSN 2050-084X. doi: 10.7554/eLife.38907. URL https://doi.org/10.7554/eLife.38907.

N. Hollenstein and N. Aepli. Compilation of a Swiss German dialect corpus and its application to PoS tagging. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 85–94, Dublin, Ireland, Aug. 2014. Association for Computational Linguistics and Dublin City University. doi: 10.3115/v1/W14-5310. URL https://aclanthology.org/W14-5310.

P.-E. Honnet, A. Popescu-Belis, C. Musat, and M. Baeriswyl. Machine translation of low-resource spoken dialects: Strategies for normalizing Swiss German. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*,

Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL https://aclanthology.org/L18-1597.

L. Hufe and E. Avramidis. Experimental machine translation of the Swiss German sign language via 3D augmentation of body keypoints. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 983–988, Abu Dhabi, United Arab Emirates (Hybrid), Dec. 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.wmt-1.95.

L. Lambrecht, F. Schneider, and A. Waibel. Machine translation from standard German to alemannic dialects. In *Proceedings of the 1st Annual Meeting of the ELRA/ISCA Special Interest Group on Under-Resourced Languages*, pages 129–136, Marseille, France, June 2022. European Language Resources Association. URL https://aclanthology.org/2022.sigul-1.17.

L. H. Leeper and R. Culatta. Speech fluency: effect of age, gender and context. *Folia Phoniatr. Logop.*, 47(1):1–14, 1995.

C. D. Manning, K. Clark, J. Hewitt, U. Khandelwal, and O. Levy. Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proceedings of the National Academy of Sciences*, 117(48):30046–30054, 2020. doi: 10.1073/pnas.1907367117. URL https://www.pnas.org/doi/abs/10.1073/pnas.1907367117.

W. G. Moulton. *Sandhi in Swiss German dialects*, pages 385–392. De Gruyter Mouton, Berlin, New York, 1986. ISBN 9783110858532. doi: doi:10.1515/9783110858532.385. URL https://doi.org/10.1515/9783110858532.385.

J. Nivre and C.-T. Fang. Universal Dependency evaluation. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 86–95, Gothenburg, Sweden, May 2017. Association for Computational Linguistics. URL https://aclanthology.org/W17-0411.

J. Nivre, M.-C. de Marneffe, F. Ginter, J. Hajič, C. D. Manning, S. Pyysalo, S. Schuster, F. Tyers, and D. Zeman. Universal Dependencies v2: An evergrowing multilingual treebank

collection. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.497.

K. Ortmann. Chunking historical German. In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 190–199, Reykjavik, Iceland (Online), May 31–2 June 2021. Linköping University Electronic Press, Sweden. URL https://aclanthology.org/2021.nodalida-main.19.

I. Pak and P. Teh. *Text Segmentation Techniques: A Critical Review*, volume 741, pages 167–181. 01 2018. ISBN 978-3-319-66983-0. doi: 10.1007/978-3-319-66984-7_10.

I. Rehbein, J. Ruppenhofer, and T. Schmidt. Improving sentence boundary detection for spoken language transcripts. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 7102–7111, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.878.

J. Ruppenhofer and I. Rehbein. Detecting the boundaries of sentence-like units on spoken german. Preliminary proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019), October 9 – 11, 2019 at Friedrich-Alexander-Universität Erlangen-Nürnberg, pages 130 – 139, München [u.a.], 2019. German Society for Computational Linguistics & Language Technology und Friedrich-Alexander-Universität Erlangen-Nürnberg. URL https://nbn-resolving.org/urn:nbn:de:bsz:mh39-93174.

T. Samardžić, Y. Scherrer, and E. Glaser. Normalising orthographic and dialectal variants for the automatic processing of swiss german. 11 2015.

Y. Scherrer. Syntactic transformations for Swiss German dialects. In *Proceedings of the First Workshop on Algorithms and Resources for Modelling of Dialects and Language Varieties*, pages 30–38, Edinburgh, Scotland, July 2011. Association for Computational Linguistics. URL https://aclanthology.org/W11-2604.

Y. Scherrer, T. Samardžić, and E. Glaser. Digitising swiss german: how to process and study

a polycentric spoken language. *Language Resources and Evaluation*, 53, 12 2019. doi: 10.1007/s10579-019-09457-5.

Schweiz für Dummies. Ausdrucksform in schweizerdeutsch. URL `https://schweizfuerdummies.weebly.com/usdrucksform.html`.

A. Speyer. zusammengezogener satz. In S. J. Schierholz, editor, *Wörterbücher zur Sprach- und Kommunikationswissenschaft (WSK) Online*. De Gruyter, Berlin, Boston, 2016. URL `https://www.degruyter.com/database/WSK/entry/wsk_id_wsk_artikel_artikel_9233/html`. 2013.

M. Stevenson and R. Gaizauskas. Experiments on sentence boundary detection. In *Sixth Applied Natural Language Processing Conference*, pages 84–89, Seattle, Washington, USA, Apr. 2000. Association for Computational Linguistics. doi: 10.3115/974147.974159. URL `https://aclanthology.org/A00-1012`.

M. Straka, J. Hajič, and J. Straková. UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4290–4297, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL `https://aclanthology.org/L16-1680`.

K. Sugisaki. Word and sentence segmentation in german: Overcoming idiosyncrasies in the use of punctuation in private communication. In G. Rehm and T. Declerck, editors, *Language Technologies for the Challenges of the Digital Age*, pages 62–71, Cham, 2018. Springer International Publishing. ISBN 978-3-319-73706-5.

H. Uszkoreit, G. Smith, A. Schwartz, B. Schrader, M. Rower, M. Pußel, C. Preis, O. Plaehn, L. Michelbacher, R. Langner, C. Kirstein, J. Janitzek, S. Hansen, P. Eisenberg, S. Dipper, V. Demberg, T. Brants, S. Brants, T. Bracht, and H. Hirschmann. Tiger annotationsschema. 07 2003.

B. Wang, L. Shang, C. Lioma, X. Jiang, H. Yang, Q. Liu, and J. G. Simonsen. On position

embeddings in {bert}. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=onxoVA9FxMw.

X. Wang, M. Utiyama, and E. Sumita. Online sentence segmentation for simultaneous interpretation using multi-shifted recurrent neural network. In *Proceedings of Machine Translation Summit XVII: Research Track*, pages 1–11, Dublin, Ireland, Aug. 2019. European Association for Machine Translation. URL https://aclanthology.org/W19-6601.

S. Westpfahl and T. Schmidt. FOLK-gold — a gold standard for part-of-speech-tagging of spoken German. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1493–1499, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL https://aclanthology.org/L16-1237.

A. Wyler and B. Siebenhaar. *Dialekt und Hochsprache in der deutschsprachigen Schweiz. 5. vollständig überarbeitete Auflage.* Pro Helvetia Schweizer Kulturstiftung. Information. Pro Helvetia, Zürich, 1997.

I. Zribi, I. Kammoun, M. Ellouze, L. Hadrich Belguith, and P. Blache. Sentence boundary detection for transcribed Tunisian Arabic. In *Konvens-2016*, Bochum, Germany, 2016. RUHR-UNIVERSITAT BOCHUM. URL https://hal.archives-ouvertes.fr/hal-01462133.