Universität Zürich[UZH]

Master's Thesis
presented to the Faculty of Arts and Social Sciences
of the University of Zurich

for the Degree of
Master of Arts

[REL Enriching] [ARG1-GOL BERT Embeddings] [ARG2-PPT with Semantic Role Labels] [ARGM-GOL for Natural Language Understanding Tasks in German]

Author: [ARG0-PAG **Jonathan Schaber**]
Student ID Nr.: 11-771-359

Examiner: Dr. Simon Clematide

Department of Computational Linguistics

Submission Date: May 27, 2021

Abstract

Employing pretrained word embeddings from large language models as input representations has become state-of-the-art (SOTA) in many Natural Language Processing (NLP) Tasks. Contextualized representations of modern transformer based architectures lead to SOTA results on standardized Natural Language Understanding (NLU) datasets like General Language Understanding Evaluation (GLUE), often on par with measured human performance. This is all the more astonishing given that models like Bidirectional Encoder Representations from Transformers (BERT) learn their embeddings from raw text lacking additional explicit linguistic structures by implementing self-supervised pre-training. Despite this, BERT embeddings have proven to transfer remarkably well to NLU tasks through few-shot fine-tuning on small task-dependent datasets. Subsequent research, however, exposed that BERT's NLU capabilities are considerably limited: BERT fails in certain, often trivial, linguistic contexts to reliably extract the semantic content of a sentence — for example, BERT is surprisingly error-prone in recognizing profound changes in meaning triggered by negative polarity items. In this thesis, I investigate if enriching pure BERT embeddings with explicit linguistic information counteracts those deficiencies. To this end, I concatenate pretrained BERT embeddings with numerically encoded, automatically predicted Semantic Role Labels (SRLs) as input representation for an end-to-end system on downstream tasks. To assess semanticity increase, I devise several head architectures and compare the performance differences of the enriched to the pure embeddings on the GerGLUE dataset comprising various NLU tasks in German, which I compiled for this thesis. The results indicate that combining raw, contextualized word embeddings with explicit linguistic information leads to significant performance increases, suggesting enhanced semanticity capabilities of these representations. However, dataset and SRL quality are paramount — translation noise, deficient SRL detection, and insufficient training data lead to suboptimal model fitting and selection.

Zusammenfassung

Und hier sollte die Zusammenfassung auf Deutsch erscheinen.

Acknowledgement

I want to thank Simon Clematide, Y and Z for their precious help. And many thanks to whoever for proofreading the present text.

Contents

List of Figures

List of Tables

List of Acronyms

BERT	Bidirectional Encoder Representations from Transformers
BLEU	Bilingual Evaluation Understudy
BPE	Byte Pair Encoding
BOW	Bag of Words
CPOSTAG	Coarse-grained Part-Of-Speech tag
CNN	Convolutional Neural Network
deISEAR	German International Survey on Emotion Antecedents and Reactions
FFNN	Feed Forward Neural Network
GRU	Gated Recurrent Unit
LCS	Lexical Conceptual Structure
LSTM	Long Short-Term Memory
ML	Machine Learning
NLP	Natural Language Processing
NLU	Natural Language Understanding
OOD	Out-of-Domain
OOV	Out-of-Vocabulary
POS	Part-Of-Speech / Poverty of Stimulus
POSTAG	Fine-grained Part-Of-Speech tag
RNN	Recurrent Neural Network
SCARE	Sentiment Corpus of App Reviews
SOTA	State of the Art
SRL	Semantic Role Labelling / Semantic Role Labeller
STTS	Stuttgart-Tübingen-TagSet
USD	Universal Stanford Dependencies

1 Introduction

To begin with, the introduction will describe the intent of this MA thesis in the bigger context. In order to do so, I will outline the general problems and topics of Natural Language Processing and elaborate on the methods and techniques that were developed to address those questions. Then, I will show how my approach ties in with current research and outline the aims and questions of my thesis. The introduction will conclude with a short description of the structure of this thesis.

1.1 Motivation

Human language holds some truly mesmerizing features and puzzles, a lot of which are still not yet understood in all their depth: For example, it is still unclear how children are able to learn the grammar of their mother tongue from the corrupted and comparatively scarce language material they are exposed to (cf. Lust [2006]; Lewis and Elman [2001]).[1]

One of the most trivial and enigmatic traits about human language and communication is that we actually *understand* each other so well: That during a discourse, person X can reliably recover the encoded meaning of the expressions in the lan-

[1]The famous term describing this phenomenon, "Poverty of Stimulus" (POS), was coined by Chomsky [1980], who makes the case for an innate language learning/processing faculty. This has led to a fierce debate over that issue: Many researchers claim that the POS-motivated necessity for such an innate, human-specific, genetic trait does not hold, and humans learn language by means of extremely sophisticated statistical analysis. It has indeed been shown that infants are amazingly adept at extracting statistical information from auditive input [Saffran et al., 1996]. However, as was countered by Chomsky and others, this does not fully explain the ability of children to apply these statistical input cues to *hierarchical* structures : "The issue that is so central to this particular POS problem is tacit knowledge by the child that grammatical rules apply to such [hierarchical] structures" [Berwick et al., 2013] — there is no behavioral explanation for the tacit knowledge about hierarchical structures in the mind of the child. Therefore, the POS argument still holds, although it has been somewhat weakened by empirical findings about language acquisition driven by statistical insights.

guage signal transmitted by person Y, and vice versa, especially given the "noisy channel" problem (cf. Plotkin and Nowak [2000]), which is elaborated on a little more below. Further, we are able to reconstruct a lot of information that is not explicitly stated in a verbal expression, and uphold such a mutually shared belief of the state of affairs during a discourse. In fact, this "shared intentionality" is hypothesized as being one of the traits that separates human cognition and language from that of other animals (cf. Tomasello and Carpenter [2007]).

Because rational thinking, reasoning, and cognition in general are being perceived as essentially linked to human linguistic competences, communicative abilities have been proposed to act as a proxy for attesting intelligence to non-human entities. One of the most famous example is the test proposed by Turing [1950]. Named after it's inventor, the "Turing-Test" is a method for determining the capabilities of artificial intelligence systems: If the communicative behavior of an artificial communicator is indistinguishable from that of a human agent, it is justifiable, according to this view, to attribute "intelligence" to such a system.

However, reproducing human linguistic capabilities has turned out to be much harder than was predicted during the 1950s, where scientists were enthused by then recent developments of computers.[2] The difficulty of reproducing human linguistic capabilities is mainly due to the fact that humans are incredibly adept at handling the various complexities present in the linguistic signal; It is with such unconcscious ease that we process language, that we are rarely aware of how we handle complexities like these:

Vagueness We have no problem dealing with vague statements like "Most people never heard of it"; in a specific conversation situation we rely on extra-linguistic cues like pragmatics and common world knowledge to decide if "most" means 7.5 billion people, or 70% of our friends, or if the approximate number is even relevant.

Ambiguity The fact that many linguistic signs cannot be interpreted in only one way is a ubiquitous phenomenon in human language. This ambiguity can be observed on all levels of language: phonological, lexical, syntactic, semantic, pragmatic. A classic example for syntactic ambiguity would be the phrase "He saw the elephant with a telescope".

Corruption Contrary to how we perceive language when speaking to each other,

[2] For instance, researchers of the Georgetown University and IBM forecasted in 1954 that machine translation would be a solved problem within a few years [Hutchins, 2005].

utterances are mostly not well-formed, grammatical sentences, but show a varying degree of stutter, incomplete phrases, repetitions,grammatical flaws, and other "mistakes". While not as present in written language, depending on the domain, there still is quite some noise present, e.g. in online chat threads "Wers nocj wach??!!" etc. Still, we mostly have no problems at all reconstructing the encoded information from such a corrupted signal — contrary to algorithmic models, as will become evident during the course of this thesis[1 .

Common world knowledge Normally the information we encode in ordinary conversation is highly condensed and as scarce as possible to enable swift conversation — most of the actual information is reconstructed by the receiver, making use of general knowledge about the world (factual knowledge, such as "Bern is the capital of Switzerland" and "Switzerland lies in Europe", which together with logical reasoning, e.g. the *modus ponendo pones*, leads to true deductions like "Bern must be geographically located in Europe") and the actual situation, time, and place the conversation takes place. A good counter-example would be legal texts, where every detail and implication of a judicial text must be tediously mentioned — a circumstance which actually makes law documents nearly incomprehensible for the untrained reader.

So, every system that is built built with the aim of processing natural language in a "deep", human-like manner must cope with this inherent fuzzinesses of human communication. In the field of applied computational linguistics, often referred to as Natural Language Processing (NLP), this subfield of research is known as Natural Language Understanding (NLU).[3] NLU aims at building systems which are able to retrieve the semantic content encoded in natural language and are able to further act upon it: For example, a chatbot should be capable of "understanding" that the

[3]Note that I will not engage in the discussion about whether it is philosophically appropriate to claim that computational models "understand" human language. A lot of controversy has arisen around this issue, with positions ranging from completely denying language models any sort of (linguistic) understanding [Bender et al., 2021] to the current trend of concentrating on beating NLU SOTAs with ever larger models and blindly taking this as proof of building architectures capable of learning human language. I would argue that the thruth lies somewhere in between: Of course it is not enough to perform well on a standardized data set to speak of "understanding", however, as Sahlgren and Carlsson [2021] point out: Also in humans, especially children, we measure language competences indirectly "by using various language proficiency tests, such as vocabulary tests, cloze tests, reading comprehension, as well as various forms of production, interaction, and mediation tests [...]". Therefore, I think it is not completely unsound to assess NLU capabilities of artificial systems by measuring their performance on standardized datasets.

questions "What's the weather like?", "Can you tell me today's weather forecast, please?", "Will I need an umbrella today?" all have more or less the same meaning and should provoke the same answer and/or reaction of the system.[4]

1.1.1 History, Methods, Problems of NLP and NLU

NLP, and NLU as a sub-discipline of it, is "[t]he engineering side of computational linguistics" that is "largely concerned with building computational tools that do useful things with language" Johnson [2009]. Therefore, the main goal of NLP is not to describe or explain structures in human language employing computational theories, but rather to solve specific problems regarding human language: For example, one might want to design and implement a system that automatically detects the emotion that is transported in a given text snippet. In order to approach these problems, researchers attempt to build systems and models using different approaches.

In the relatively young history of the field of NLP, there are three phases identifiable regarding the main architectural strategies of these models or systems, which I will briefly outline:

During the first, or *symbolic*, phase of NLP that lasted approximately from the 1950s until the 1980s, systems addressing tasks such as machine translation were architectures that consisted of carefully hand-written symbolic grammars and knowledge bases. This period was heavily influenced by works in theoretical linguistics, especially Chomsky's transformational-generative grammar [Chomsky, 2009, 2014]. The initial enthusiam over the theoretical proficiency of such models, however, was soon severely dampened when it became clear that the complexity of human language in real-world applications had been underestimated.

Beginning in the 1990s, a *statistical turn* took place, and NLP turned towards data-driven solutions, making use of the increased availability of large amounts of electronic text on the one hand, and computers with increased computating power and memory on the other [Liddy, 2001]. NLP problems were now being addressed by learning patterns from huge data collections. One driver of this paradigm shift were the various difficulties the traditional, symbolic systems entailed: Their development "requir[ed] a great deal of domain-specific knowledge engineering. In addition,

[4] Although one could argue that the third question differs from the first two since it is a polar question and therefore a simple "yes" or "no" would be grammatically correct, I would argue that one would perceive this as a very dry, or even rude, answer and would expect a more elaborated answer in a regular conversational context.

the systems were brittle and could not function adequately outside the restricted tasks for which they were designed" [Brill and Mooney, 1997, p. 13]. The new, statistical approach tried to tackle NLU-problems by shifting the focus from tedious hand-crafting "to empirical, or corpus-based, methods in which development is much more data driven and is at least partially automated by using statistical or machine-learning methods to train systems on large amounts of real language data" [Brill and Mooney, 1997, p. 13]. The main challenge for engineers and scientists now lay in identifying suitable features and encoding them in appropriate data structures, according to which the algorithm would learn helpful patterns from the language data for solving the task at hand.

This orientation towards data-driven NLP solutions enabled new evaluation methods. Using standardized data sets, different architectures could be compared to evaluate their capabilities regarding the problem the dataset was constructed to display.

Half a decade ago, NLP and NLU entered a new stage: we are now in the midst of the *neural age* of computational linguistics, where models are constructed implementing deep, i.e. multi-layered, neural networks for learning generalizable patterns and functions from data. For many people in the NLP field, the advent of these machine learning algorithms - powerful, but obscure and difficult to interpret - also posed a threat to the established approaches regarding NLP problems: "Deep Learning waves have lapped at the shores of computational linguistics for several years now, but 2015 seems like the year when the full force of the tsunami hit the major Natural Language Processing (NLP) conferences" [Manning, 2015, p. 701]. Nevertheless, deep learning approaches are an integral part of the vast majority of NLP applications developed since 2015.

In contrast to the statistical period's main challenge — the identification and extraction of suitable features —, the algorithms in the current, neural approaches are learning the features that are most informative for a given task from the raw data - on their own.[5] The human part in the process is now "reduced" to designing the overall model architecture, defining the hyperparameters of the model and the training set up, and compiling large enough amounts of data that are, in the best case, also of good quality.

[5]This is probably most obvious in computer vision contexts, where the input data for neural models is the raw RGB pixel image data, in contrast to pre-deep learning algorithms, where the input of a machine learning system would be a data structure produced by some feature detection algorithm, such as SIFT [Lowe, 1999]. For text, however, the case is not as clear as to what the "raw" representation of it for a model would be, as will be shown in the next chapter.

While the roughly sketched methods above apply to a wide range of applications in NLP, I will now point to some of the inquiries NLU aims at: Simply put, NLP is concerned with the structural side of natural language text, while NLU looks at the content of these utterances. For example, typical NLP tasks, such as dependency parsing, POS-tagging, and coreference soltution do not require a semantic representation of words or phrases — often, it suffices to analyze structural properties such as morphology, unsophisticated frequency statistic, or transition probabilities to solve such problems to an acceptable extent (cf. Kumawat and Jain [2015]). By contrast, NLU tries to process language in a manner similar to humans: We infer something from language, reconcile communication information with our world knowledge, answer questions, detect logical inconsistencies etc. NLU is the endeavor of building artificial agents that are able to do these things as well.

However, all disagreements about the very nature of "understanding" and "communication" set aside, there probably is consensus that the field is far from being able to construct a system that would pass the Turing test, i.e. an agent which communicative abilities are on par with a human being. Therefore, the strategy is to analyze the general human communicative "understanding" capability and identify core building blocks of it, which are then addressed in isolation.

1.1.2 Subtasks of Solving NLU

As the endeavor of building a holistic, fully NLU capable agent is probably an objective too ambitious to achieve in one go, the strategy in modern computational linguistics is to break it down into smaller "subproblems": Several core building blocks of NLU have been identified and datasets comprising examples tailored at those have been compiled. The performance of new models and approaches on such data sets is often taken as a proxy for attesting NLU capabilities targeted at the task.

Some of these proposed subproblems or proto-tasks of NLU are the following:

Sentiment Detection Given a sentence, we can normally assess if the emotion transmitted by it is rather positive, negative, or neutral: "Oh my, what a lovely day!" is clearly positive, while "asdasd" probably is not.

Grammaticality Recognition Being able to understand utterances in a language also implies being able to judge the grammaticality of any statement in that language: "Eagles that fly can eat" is judged as valid by English speakers while "Eagles that fly eat can" is not.

Entailment Recognition An important ability of understanding is the recognition of the logical relationship in which two utterances stand. In other words, given two sentences A and B, does B follow from A? For example: Given the sentence "The weather forecast predicts rainfall the next days", does the sentence "Tomorrow will be a beautiful, sunny day" conform, contradict or stand in a neutral relation to the first sentence?

Question Answering To locate a (or rather, the) relevant text span in a given context according to some questions is also a task where one would assume that some sort of understanding is needed: Given the question " What was the name of the King of England?" and the context "Henry V (16 September 1386 – 31 August 1422) was King of England from 1413 until his death in 1422.", a system would would need to extract the span "Henry V".

Since each of these tasks and datasets are focused on a specific subpart of general NLU, it is by no means implied that a system which e.g. performs well on entailment recognition will be able to extract answer spans to an acceptable degree. Therefore, to assert that some model or approach does indeed capture some properties of general NLU, it has become standard to evaluate models on several of those subtasks. One of the best established compilations of such subtasks is the GLUE benchmark [Wang et al., 2018]. GLUE, which stands for **G**eneral **L**anguage **U**nderstanding **E**valuation, is " a collection of NLU tasks including question answering, sentiment analysis, and textual entailment, and an associated online platform for model evaluation, comparison, and analysis." [Wang et al., 2018].

Since this thesis is concerned with German and there is no such standardized benchmark, I compiled one myself — GerGLUE. GLUE and GerGLUE are described in more detail in chapter 3.

1.1.3 Text Representations

Text can be analyzed and represented in various ways; in fact, those characters on the sheet you are reading right now are one form of representing language. However, for an algorithm, language represented by means of an alphabetical script is not a well-suited encoding — since de Saussure's lectures at the start of the 20th century, it is established that the actual characters are completely arbitrary in what they denote [De Saussure, 1989]. There is nothing in the properties of the character(-sequence) *itsef* which would relate it necessarily to the thing it represents. Nothing in the mere shape of the character sequence "contrary" implicates on what meaning, or signifié, this string, or signifiant, maps onto. As human beings, if capable of

reading the Latin alphabet and knowing the English language, we decipher the string to symbolize the linguistic concept, we pronounce /ˈkɑntɹəɹi /.[6] In other words, for a computational system to act upon human language input, we must find a way to encode this input in a way that actually represents structural information; depending on what the model is aimed for, this may be phonological, syntactical, or semantic properties.

For example, one could represent each word of a sentence as a number, e.g. the page number of the Oxford English Dictionary on which the definition of the word — or its lemma base form —is given: <Every> <event> <has> <a> <cause> → <234> <229> <388> <12> <176>. However, that would probably not be very informative. But if we represent each word of a sequence by its Part of Speech, a computational system designed for e.g. identifying Named Entities might retrieve task-supporting information from it — especially if both representations, the "normal" one and the POS one, are combined: <Every DET> <event NOUN> <has VERB> <a DET> <cause NOUN>.

Embeddings

Most NLU models don't operate on the text as it is, i.e. as an array of UTF-8 encoded signs. Often, it is easier to implement an algorithm that processes text in some way, to encode this text numerically. A strategy adopted quite early is called "Bag of Words" (BOW) [Harris, 1954] technique, which encoded a sequence of words s in one vector, basically indicating what items of a given vocabulary are present in s. While this BOW technique is quite effective for certain tasks, e.g. information extraction systems often make use of it, it has some flaws: It fails in particular to reflect the core property of language being inherently sequential — the BOW encodings of "Alice hit Bob" and "Bob hit Alice" are indistinguishable.

Therefore, other methods of representing word sequences numerically have been devised, which assign to each word in a sequence a numeral representation, thus preserving the sequential information. Examples of such a representation would be Latent Semantic Analysis [Furnas et al., 1988], which generates word vectors by reducing high-dimensional word co-occurrence matrices via singular value decomposition, or word2vec [Mikolov et al., 2013]. This opened the possibility of letting a neural model compute those embeddings, while applying a language modeling

[6] There exist what are called "featural writing systems" which actually encode phonological information in the form of their characters, e.g. the Korean Hangul alphabet, which encodes the position of the tongue for producing a consonant [Sampson, 1985].

objective.

Contextualized Word Embeddings

Research on the generation algorithms of such numerical word embeddings, which can be used as text representations for various NLP tasks, has experienced tremendous interest in recent years, producing various architectures and frameworks, leading to the current state-of-the-art: contextualized word embeddings. Contextualized word embeddings account for the fact that the semantic function of a word is heavily dependent on the context in which it appears. For example, the word "building" has quite different semantic and syntactic functions in the following sentences: "This is a very high building!" and "She was building an empire" — in this obvious example[7], a word representation framework which does not account for the context, will clearly fail to adequatly interpret such sentences.

The afore mentioned neural approaches in NLP have proven to be very adept at producing such contextualized word representations: "The gains so far [from Deep Learning] have not so much been from true Deep Learning (use of a hierarchy of more abstract representations to promote generalization) as from the use of distributed word representations—through the use of real-valued vector representations of words and concepts." [Manning, 2015, p. 703]

One effective framework of producing such contextualized word embeddings is BERT [Devlin et al., 2018], which I also employ for the experiments in my thesis. The general concepts of contextualized word embeddings, transfer learning, and BERT in particular will be described in depth in the following chapter 2, section 2.1.

Semantic Structures

While the motivation for representing a sentence numerically is to let algorithms operate "autonomously" on these representations, for example by computing some distance measures between a query and a set of texts for information retrieval, one can also try to encode some structural information of the sentence at hand. To support a model with information about what relations are present between different parts of a sentence, one could provide the algorithm with the syntax parse tree of the

[7] The two sentences exemplify the problem of homonymy: Two different lemmata having the same word form. However, even if the same lemma is used in different context, subtle changes in semantics and grammatical function are present.

sentence. This way, the difference between the two "hitting"-sentences mentioned earlier would be clearly distinguishable.

Semantic Role Labels

Semantic Roles are a linguistic tool developed for analyzing a sentence with regard to the semantic relations that hold between different entities involved in the action described by the sentence, thus "explaining both the syntactic structure as well as the meaning of sentences" Bussmann [2006]. It is sometimes considered to be one of the oldest concepts in linguistic, conceptually contemplated as early as in the 6th century BCE by the old-Indian grammarian Pāṇini (cf. [Gildea and Jurafsky, 2002]).

The core intention is to identify generalizable semantic functions, or semantic roles, that participants in an event can engage in. With such an instrument, it is possible to model the semantic equivalence of grammatically and syntactically quite distinct sentences (after Palmer et al. [2010]):

(1.1) John broke the window.

(1.2) The window broke.

In the first sentence, "break" is a transitive verb, while it is intransitive in the second. "Window" is the grammatical object in prior, while being the grammatical subject in the latter sentence. Despite these differences, the expressed action, or the encoded information, is the same both times: an object (in the physical, not grammatical sense), the window, was shattered.

Semantic Roles are an attempt to formalize the semantic content of sentences by attributing each constituent a generalizable label. In this case, "the window" would be labelled as *patient* , i.e. the participant undergoing a change of state, in both sentences. In the first, where there is also a clear initiator of this change, "John" would be labelled as the *agent* of the event.

Thus, as a further form of representation, any sentence could be represented by the words in it replaced by their Semantic Role Label (SRL): <John> <broke> <the> <window> → <AGENT> <PREDICATE> <PATIENT> <PATIENT>. (Or, as mentioned before, a combination of the "normal" text representation combined with SRLs)

A more detailed elucidation on the concept semantic roles and its history is provided in chapter 2, section 2.3.

1.2 Contributions/Goals

The goal of this thesis is to explore if a combination of the two text representation techniques described above — Contextualized Word Embeddings and Semantic Role Labels — may lead to embeddings which support machine learning algorithms targeted at NLU tasks. For English, Zhang et al. [2019b] showed that combining BERT embeddings with numerically encoded SRL information leads to an increase in performance on NLU tasks, compared to the vanilla[8] BERT architecture. However, their findings are somewhat inconsistent and, probably due to the format of a short paper, the description of the architecture and sound analysis of the results are rather meager.[9]

Although my approach stands in the tradition of recent machine learning trends in NLP — i.e. training a model on data and using test data to asses its capabilities — I nevertheless also view it as an attempt at re-integrating linguistically motivated structures into NLP, as the linguistic nature of NLP is sometimes left on the sideline. As [Manning, 2015, p. 702] observes: "Recently at ACL conferences, there has been an over-focus on numbers, on beating the state of the art. Call it playing the Kaggle game." Despite the apparent success of linguistic-agnostic architectures on NLP benchmarks, those models tell us little about the nature of human language and the structural properties of it. To continue with Manning, p. 706: "It would be good to return some emphasis within NLP to cognitive and scientific investigation of language rather than almost exclusively using an engineering model of research."

By implemeting textual representations that are explicitly grounded in linguistic theory, I see part of the contribution of my model in antagonizing the purely data-driven score-games of modern NLP. I hope to provide an approach that deviates from the brute-force data engineering of many current research projects, which makes use of the affordances of contemporary computational linguistics while being motivated by linguistic concepts and established frameworks and the possibility of gaining

[8] In computer science and hacker culture in general, "vanilla" is the term describing the use of computer hard and/or software without customization of its original implementation (cf. The Jargon File).

[9] For example, it is not clear *what* results for the datasets are being reported: The best run out of several experiments, the average performance of several experiments, etc. (due to the random initialization of the weight matrices of the head modules and SRL encoder, there are expectable fluctuations in performance, which makes it highly improbably that only one model per task was trained). Neither control Zhang et al. for statistical significance, which makes sound statements about architecture superiority quite disputable.

insights into the inner workings of human language.

1.3 Research Questions

This thesis sets out to answer the following research questions:

1. Does the combination of BERT embeddings with structured SRL information have a positive, measurable effect on NLU tasks?
   - Since I replicate, in some sense, the work of Zhang et al. [2019b], are my findings similar to what they report?
   - Are there differences between datsets, registers, tasks, etc.?
   - Are the datasets well-suited for making sound statements about the effect of enriching BERT embeddings with SRLs?
   - If the preceding question has to be answered negatively, is there a way to determine nevertheless if SRLs might have a positive effect in more appropriate settings?
   - Can I determine what aspect(s) of SRLs support models in downstream tasks?
2. Which head architecture for fine-tuning BERT-based, SRL-enriched text representations is best suited for NLU tasks?
3. What are the effects of different representation methods of combinations of BERT-subtokens with SRLs and different encoding techniques on the multi-layered nature of semantic roles?
   - The number of considered predicate-argument structures is fixed to be three per sentence — is it more effective to fill empty slots with 0-SRLs, or duplicating existing SRLs?
   - To merge BERT embeddings with SRLs, either the splitted BERT subtokens need to be merged back or the SRLs have to splitted up — does this lead to different results?

1.4 Thesis Structure

In this first chapter I gave a very brief overview of some general problemse and proposed solutions in NLP over the past decades and highlighted the critique recent purely data-driven approaches have been receiveing from the linguistic fraction of NLP.

Chapter 2 introduces the basic concept of BERT and its shortcomings, and presented solutions or improvements. Further, I explain my approach and the relating linguistic concepts.

The datasets I compiled in my GerGLUE collection to test my models on are described in detail in chapter 3.

In chapter 4, I describe the architecture of my GliBERT model in all detail: The identification and encoding of the semantic role labels, the different combination procedures of the BERT embeddings with these, as well as the different head architectures.

The overview of the results and the discussion of the performance of the various model architectures on the GerGLUE data set is made in chapters 5 and chapter 6.

Finally, I draw my conclusions and summarize the gained insights in the last chapter 7.

2 Approach

In this chapter, I will give a brief overview of several things: In a first step, I will elaborate on the topic of pretrained embeddings, transfer learning, and BERT, which is probably the most successful architecture implementing these concepts. Secondly, I will briefly demonstrate problems that have been identified relating to the performance of BERT and its relation to semantics. Then, I will point out some submitted solutions countering those problems. Lastly, I will describe my approach and how it makes use of semantic roles.

2.1 Pretrained Embeddings and Transfer Learning

In recent years, it has proven to be very effective to use word embeddings precomputed by sentence or document encoders as input representations for task-specific architectures, which may fine-tune those embeddings while learning the task at hand. Pretrained word embeddings have the advantage that they do not need to be learned from scratch by the system solving the task at hand.

Bengio et al. [2003] first implemented distributed word representations by using a neural n-gram language model which could be re-used in downstream tasks. In "Natural language processing (almost) from scratch", Collobert et al. [2011] showed the utility of such embeddings for representing text as input for other neural networks which addressed a multitude of classical NLP tasks. Mikolov et al. [2013] proved that such embeddings could be computed using a modest one-layer neural network targeted at self-supervised training objectives, namely CBOW and skip-gram. Context-sensitvity, i.e. taking into account the surrounding words in the embeddings, was introduced by the ELMo architecture Peng et al. [2019]. Since then, representing words using pretrained, contextualized embeddings has become the standard in most NLP architectures.

The architecture computing contextualized word embeddings that has caused the most uproar recently was probably BERT Devlin et al. [2018], a model that led to so many variants of it, that it created a whole new field inside the NLP community —

winkingly baptized as "BERTology" Rogers et al. [2020]. BERT is a good example for a typical neural age NLP system: Its architecture is completely agnostic of symbolic or structural, inguistic knowledge about language whatsoever, it "merely" operates on sequential concatenations of symbols. Therefore, it also does not employ preprocessing of the data of any kind — no POS-tagging, no dependency parsing, no NER.[1] Despite this complete lack of any sort of explicit linguistic knowledge, by essentially extracting statistical patterns which it learns from processing huge amounts of text, the resulting BERT embeddings achieved several SOTAs on well-established NLU data sets, such as GLUE Wang et al. [2018].

The basic concept of BERT is described by Devlin et al. as being a two-stepped framework: (1) Pretrain the model on unlabeled data over different pretraining tasks[2] and (2) for downstream tasks, use the embeddings by initializing the model with the pretrained parameters and fine-tune all of the parameters by using labeled data from the task at hand.

One of the distinctive features of BERT is the minimal difference in architecture between pretraining and finetuning: The transformer based network which computes the contextualized token embeddings during the language modeling pretraining is also used in downstream tasks, only with a task-specific head on top of it. This makes re-implementing the BERT model in other architectures quite convenient, which is one of the reasons I decided to implement BERT in my approach as well.

Further, the numerical embeddings BERT "learned" to compute during the pre-training phase show to be very apt at transferring to a wide range of down-stream tasks: As Devlin et al. demonstrate, with marginal architecture adaption, i.e. the "head", to the task at hand, vanilla BERT embeddings achieved several SOTA results on tasks ranging from natural language inference (NLI) to question answering.

Another advantage of BERT is that the cost, hardware, and data intensive pre-training of the embeddings must only be computed once; the downstream task-

[1] Because of this non-linguistic specific architecture, BERT can easily be adapted to operate on other sequential data. This has actually been done: Ji et al. [2020] for example trained a DNABERT model for successfully deciphering non-coding DNA.

[2] Devlin et al. use the BooksCorpus, consisting of 800 million words, plus the English Wikipedia, consisting of 2.5 billion words. BERT optimizes its parameters on two training objectives: (1) Presented with a sentence containing one random word masked, the model has to predict it, and (2) BERT has to decide if, given two randomly sampled sentences, the second is a valid continuation of the first. Crucially, both tasks can be generated automatically, no tedious human annotation of data is needed.

dependent fine-tunig can then be carried out in a lean set up.[3]

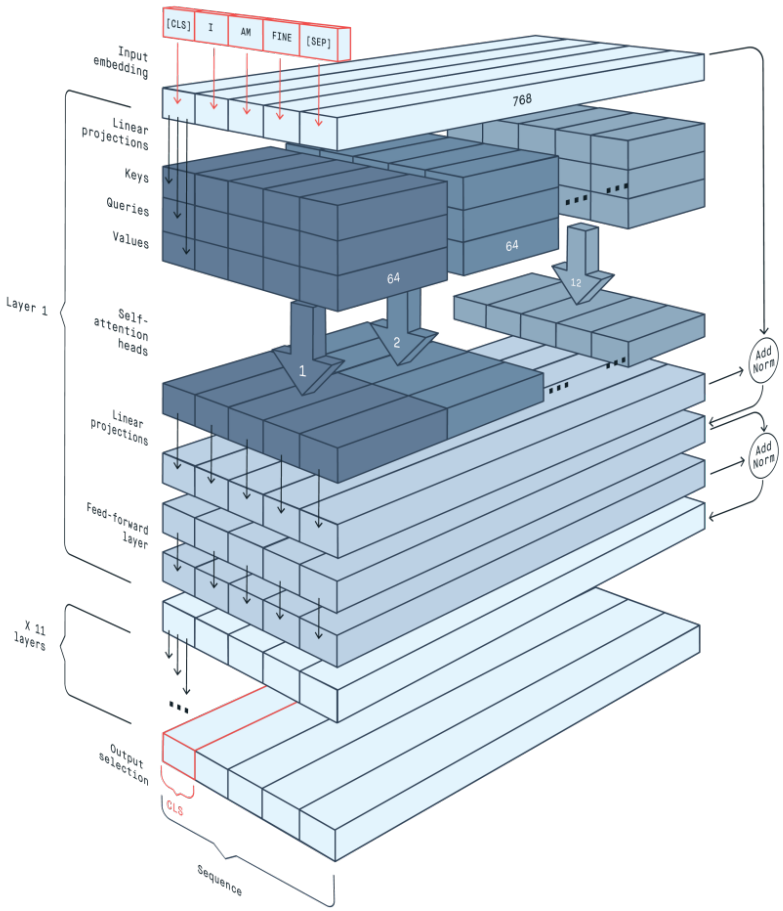

Figure 1: 3D-visualization of the BERT architecture. The 12 attention heads and following linear projections in one block (here called "Layer") are vividly represented. The information flow is top-down, with the tokenized input sentence starting with the [CLS] token on top, and the computed embeddings on bottom. Credit for the image goes to Peltarion.

From a more technical point of view, BERT is first and foremost a multi-layered

[3]To give an impression on the expenses of pre-training the BERT architecture: Schwartz et al. [2019] estimate the pre-training for BERT-large to have lasted four days on 64 TPU chips, resulting in power expenditures of about $7,000. However, this has to be considered rather cheap compared to recent architectures' sizes: The largest architecture to date is the T-NLG (Turing Natural Language Generation) built by Microsoft, possessing a staggering 17 billion parameters — that is approximately 48 times the size of BERT-large (350 million parameters), cf. Sharir et al. [2020]. Open AI's GPT-3's [Brown et al., 2020] pretraining is estimated to have costed $12 million [Floridi and Chiriatti, 2020]. This trend of increasingly bigger language models has earned severe critique from several sides, ranging from ecological and social to linguistic concerns over such models; for a good overview of these points see Bender et al. [2021].

bidirectional transformer encoder. At its heart lies an implementation of the self-attention mechanism, the transformer, introduced by Vaswani et al. [2017]; the main difference, and apparent advantage, to other architectures that implement the transformer architecture, e.g. Open AI's GPT [Radford et al., 2018], is that BERT is a bi-directional architecture. In simple terms, BERT takes an input sequence, tokenizes it and computes contextualized vector representations for each token via stacked blocks, depicted in figure 1, employing self-attention and linear combination. For reasons of space, I will not go into more detail here about BERT's architecture; the interested reader is referred to the original BERT paper by Devlin et al. [2018], as well as Clark et al. [2019], which provides insight into the inner workings of BERT's attention mechanisms.

2.1.1 Problems, Weak Spots, Semanticity

In recent years, a lot of research went into analyzing and improving the BERT architecture, as well as trying to find explanations for the striking transfer learning capabilities of its embeddings.

While BERT showed to be highly effective on several established data sets and benchmarks such as GLUE [Wang et al., 2018], it soon became obvious that it also had its weak-spots: Ettinger confronted BERT with three language tasks originating in psycholinguistics which are well-known to be difficult to tackle. After carrying out several experiments and error analyses, he concludes that BERT "struggles with challenging inferences and role-based event prediction — and it shows clear failures with the meaning of negation" [Ettinger, 2020, p. 46].

Apparently, while being able to solve NLU tasks to an impressive extent, BERT seems to be prone to fail in situations where proper semantic understanding of text sequences is crucial, such as detecting role reversal or sentence completion tasks.

Jiang and de Marneffe [2019] also state "that despite the high F1 scores, BERT models have systematic error patterns", which for them suggests "that they still do not capture the full complexity of human pragmatic reasoning."

Jin et al. [2020] go even further and create what are called adversarial attacks for laying open weak spots of BERT: After observing that BERT seems to rely only on the statistical cues of a small number of the input tokens to form its predictions, Jin et al. built a sophisticated algorithm to permute the input sentences without actually changing its meaning. Taken from the SNLI [Bowman et al., 2015] dataset, here is an example for such an attack:

(2.1) **Premise**: A child with wet hair is holding a butterfly decorated beach ball.

Original Hypothesis: The *child* is at the *beach*.
Adversarial Hypothesis: The *youngster* is at the *shore*.

The italicized words are the ones that were affected by the adversarial algorithm. Obviously — for a human — the meaning of the adversarial hypothesis has not changed from the original one. One could argue that there is a slight difference in style (the adversarial sounds somewhat overblown to me) but it would still count as an entailment of the premise. However, as the authors report, BERT is affected by such attacks and changes its predictions.

For SNLI, Jin et al. report to bring down BERT's original accuracy of 89,4% to an astonishing 4,0% by permuting 18,5% of the input tokens. Given that the permutations are essentially nothing more than exchanging a word with a synonym or a semantically related substitute, this outcome does not speak for BERT's quality.

2.1.2 Solutions / Related Work

But the NLP community was not only passive and/or destructive concerning BERT, it also produced a vast number of adaptations, variations, and improvements to the vanilla BERT. The motivations behind all those BERTlings are as manifold as one can think: some are adaptions to languages other than English, some are general variations (in hope of improvement) of the BERT architecture, still others attempt to explicitly address the problems outlined above.

The following overview sheds some light on the iridescent potpourri of the BERT family.[4] Note, however, that this is by no means an exhaustive presentation of all BERT-variants produced so far.

Adapting to other languages One of the most straightforward modifications to the BERT model is to pre-train it on different langauges. Examples for this are the French CamemBERT Martin et al. [2019], the Italian AlBERTo Polignano et al. [2019], or the Dutch BERTje de Vries et al. [2019].[5]

Adapting to specialized domains BERT was pre-trained on two corpora: (1) The

[4]This compilation is partly drawn from the towards-data-science article "A review of BERT based models" by Ajit Rajasekharan.

[5]Devlin et al. [2018] also trained a multi-lingual BERT (mBERT), which was trained on 104 languages. However, language-specific BERTs have been shown to be more performant than employing the mBERT.

BooksCorpus [Zhu et al., 2015], consisting of 800 million words and comprising 16 different genres; and (2) the English Wikipedia, lists and tables excluded (2.5 billion words). Examples of BERTs pre-trained on specialized domains are for example BioBERT Lee et al. [2020], which is an adaption to biomedical language, and LEGAL-BERT Chalkidis et al. [2020] which is a whole family of BERT models pre-trained on legal texts.

Including different modalities Another, highly interesting amplification of the BERT architecture is the inclusion of additional modalities, for example (moving) images: VideoBERT Sun et al. [2019a] learns embeddings for image-enriched texts and can be used for image captioning or image classification tasks. Several researchers claim that the future of NLP relies on combining text with sensory, e.g. visual, data for creating more stable and reliable models (cf. [Bisk et al., 2020; Bender et al., 2021]).

Optimizing architecture/training objective(s) Several BERT-variations modify the actual architecture of BERT: DistilBERT Sanh et al. [2019] is a variant 60% of the size of the original BERT while retaining 97% of its original performance. RoBERTa Liu et al. [2019] essentially modifies core hyper-parameteres such as batch-size, byte-level BPE, and the like, creating a more stable BERT. DeBERTa He et al. [2020] modifies the attention mechanism and the position encoding, while TransBERT Li et al. [2021] introduces a new pre-training framework.

Incorporating structured information One of the strengths of BERT is the self-supervised pre-training on unstructered, raw text. However, research has shown that including structured linguistic information can stabilize BERT and even counterbalance some of the known weaknesses (see above) to some extent: ERNIE Sun et al. [2019b] includes a knowledge graph into BERT, making structural fact representations available to BERT. VGCN-BERT Lu et al. [2020] combines a Vocabulary Graph Convolutional Network with the standard BERT, and Zhang et al. [2019b] include semantic role labels in their SemBERT during fine-tuning.

2.2 GliBERT

Infected by the pandemic BERT-fever and persuaded by its proven transfer-learning capabilities, I decide to also use BERT as pretrained, contextualized word representations in my architecture. Specifically, I concentrate on German and choose the

German BERT, pretrained and provided by deepset[6]. Concentrating on the weak spots identified relating to the semantic understanding of language, my approach will combine the German BERT embeddings with structured linguistic information to counter those shortcomings.

There exist several linguistic structures which could be hypothetically included into BERT. For example GermaNet, [Hamp and Feldweg, 1997] is a large lexical-semantic net that relates noun, verbs, and adjectives semantically by grouping lexical units that express the same concept into synsets and by defining semantic relations between these synsets. As such, GermaNet can be characterized as a thesaurus or a light-weight ontology. One difficulty relating to such structures is that the encoding of such hierarchical information and the subsequent combining with contiguous data — the sequence of BERT-embedded tokens in a text — is not straightforward and requires an elaborate pipeline of different subsystems.

Another, more plain, possibility would be to implement some linguistic-semantic "mark-up" of the tokens in a text: E.g. Dependency-parse the text and concatenate the BERT-embedded head tokens with a numerically encoded representation of the directed binary grammatical relation that it governs ("direct object", "determiner", etc.).

Despite these options, I choose to employ semantic roles for several reasons. Primarily, because Zhang et al. demonstrated successfully the feasibility of this undertaking for English. Moreover, the addition of semantic roles to BERT strikes a good balance between two extremes: (1) Including sophisticated knowledge structures, or semantic information, which would require extensive preprocessing (stemming, identifying content words, potential word sense disambiguation, look-up in the knowledge base) and rather cumbersome encoding; and (2) straightforward, on the fly "mark-up" of input text, with low information substance in the case of named entities. With semantic roles, I get the best of both worlds: Structured information which is relatively easy to implement, as discussed below, while — hopefully — truly adding semantic substance to the vanilla BERT embeddings. In addition to outlining their goals, history and applications, the next section describes in detail what Semantic Roles are and how I use them to augment BERT.

Since it is common practice to give your enhanced/variegated BERT architecture an appropriate name, I decided to not deviate from this tradition and call my breed **G**erman **l**inguistic **i**nformed BERT, or short: **GliBERT**.

[6] `https://deepset.ai/german-bert`

All code relating to the following dataset set-ups, GliBERT architecture, and training can be found in my GitHub repository[7].

2.3 Semantic Roles

One difficulty every system targeted at NLU must tackle, is the ability to cope with the vast amount of flexibility and freedom in natural language to express or describe one and the same state of affairs. There may be subtle differences in emphasis, markedness, or style, but the following sentences all roughly describe the same states of affairs:[8]

(2.2) The ship leaked critically due to the big waves and went down.

(2.3) Severely damaged by the hurricane, the vessel sank to the ground.

(2.4) The crew — unable to save the stricken freighter — had to be evacuated by air.

Although these three sentences make use of very different vocabulary — an unweighted BLEU score[9] is virtually zero between them — it is obvious to a speaker of English that they all convey more or less the same meaning, that all of them refer to the same state of affairs: A ship sank because of the forces of nature. In linguistics, this is often referred to as proposition: The "lexical kernel of a sentence that determines its truth conditions, regardless of the syntactic form and lexical filling of the given form of expression" [Bussmann, 2006, p. 959]. In other words, the three sentences above describe the same states of affairs, which means that they are verified or falsified by the same conditions in the real world — i.e. *if* what they are denoting *is* really the case.

Maybe the most obvious way of encoding the same proposition with different words is synonymy: "Ship", "vessel", and "freighter" all refer to the same object in the examples above. Having several options when choosing a word to denote something is a paramount feature of human language. As discussed in 2.1.1, merely exchanging

[7] `https://github.com/JonathanSchaber/Masterarbeit`

[8] Of course, from an aesthetic, literary point of view, the choice of the right words is crucial and should by no means be played down — "The difference between the almost right word and the right word is really a large matter. 'tis the difference between the lightning bug and the lightning", as Mark Twain famously put it.

[9] BLUE is a metric which captures n-gram overlap between two ore more sentences, see chapter 3, section 3.2.3 for more details.

a small subset of the words in a sentence with their synonyms potentially has a severe impact on models which are aimed at NLU tasks on such sentences. In other words, vanilla BERT often fails to recognize that two slightly different word sequences have the same meaning.

Further, the description of one and the same event, e.g. the aforementioned proposition of a sinking ship, can be linguistically encoded in various ways: In the second sentence, the process is denominated explicitly using the verb "to sink"; in the first, the semantically more obscure semi-fixed expression "to go down" is used to inform about that very situation; while in the third, the sinking of the ship is not mentioned explicitly but inferable from the circumstance of "not being able to save" it.

For a human speaker, all this disentangling, recognizing coreference, reconstructing not explicitly mentioned information, etc. happens automatically and without effort — for an algorithm, however, phenomena like the ones mentioned pose serious challenges. In other words, despite the differences that may be encountered on several linguistic levels, such as the vocabulary, word ordering, emphasis, etc., there is a remarkable capability in human language processing which reliably extracts the propositional content out of any linguistic statement.

Therefore, equipping an NLU algorithm with tools that enable semantic interpretation capabilities is a core issue that needs to be addressed: "For computers to make effective use of information encoded in text, it is essential that they be able to detect the events that are being described and the event participants." [Palmer et al., 2010]

As I laid out in section 2.1.1, in modern, purely data-driven models like BERT, all linguistic, semantic, and factual knowledge the model acquires is infered, or learned by it implicitly from raw text data. Nevertheless, and this is why the NLU field is so intrigued by it, BERT seems to perform surprisingly well in tasks where such "understanding" of events are being tested.[10] Simultaneously, some investigated failures of BERT seem to indicate that this "understanding" does not go too deep and that, among other things, the recognition of proposition equivalency between sentences is partially poor.

Semantic Roles are an attempt at creating an instrument with which it is possible to analyze the meaning of sentences in a structured manner and being able to express in generalizable terms their semantic properties, e.g. that two sentences express the

[10] Of course, one does not really measure epistemic understanding in such tests, but this is maybe the closest we can get (cf. Sahlgren and Carlsson [2021])

same proposition. The central idea here is that every utterance has an underlying semantic structure[11] (sloppily phrased: "Who did What to Whom, and How, When and Where?"), which can be realized in different surface structures. There have been various undertakings in creating a vocabulary for describing such structures, putting the focus on different aspects and showing varying degrees of analytic detail.

The work "The Case for Case" [Fillmore, 1967] is often seen as the starting point for the theory of semantic roles in modern linguistics. In it, Fillmore argued that what he called "Deep-Cases" play a crucial role in the Deep-Structure of sentences; the hitherto prevalent view in Generative Grammar was that case was a purely Surface-Structure related phenomenon and only one of several possibilities to realize syntactic relationships. Interestingly, these "Deep-Cases" were semantically-motivated: in combination with so called verb-frames, these cases would capture the semantic core of the proposition embedded in the Deep-Structure. A verb like *open* would e.g. form a frame denoting an "opening event" involving an actor, the "Opener", and an object, "the thing opened" (cf. [Fillmore, 1967, p. 46f.]). Seven of such Deep-Cases were proposed by him, for example the "Agentive": " [T]he case of the typically animate perceived instigator of the action identified by the verb"; or the so-called "Factitive": " [T]he case of the object or being resulting from the action or state identified by the verb, or understood as a part of the meaning of the verb" [Fillmore, 1967, p. 46]. The Deep-Structure of a sentence would then be realized via certain transformational rules as actual, linguistic utterance. For instance, the question "Did he really go to school in XYZ?" and "He went to school in XYZ" are realizations of the same underlying Deep-Structure. Once these transformational rules and Deep-Structures are understood and described in sufficient manner, one can turn the analysis around: by relating the observable surface structures of an utterance to the underlying Deep-Structures via the mapping rules, we have an instrument for retrieving the underlying semantic structure or proposition.

Building on these core concepts introduced by Fillmore, other linguists added features to the project of formalizing the core semantic structures of propositions, as summarized by Palmer et al.: In the beginnings of the 1970s, Jackendoff [1972] expanded and refined Fillmore's model by introducing the concept of primitive conceptual predicates and their property of governing arguments, which were conceptualized as bearing some proto-semantics, similar to the Fillmorian Deep-Cases. This approach, known as "Lexical Conceptual Structure" (LCS), proved to be an elegant theory and capable of generalizing well between different verbs. In the 1990s LCS

[11] Often, especially in Generative Grammar traditions, this level is also known as deep structure, or D-structure.

was implemented as system for representing semantics in early NLU and translation models [Palmer et al., 2010]. But, due to its detailed analysis of verbs into (several) primitive predicates and their highly verb-specific conceptualized semantic roles, LCS turned out to be cumbersome to extend to the whole range of a vocabulary of a language.

In contrast, Dowty [1991] approached the problem of constructing a framework for analyzing core conceptual semantic structures from a different angle: Instead of providing a detailed description of the primitive predicate and idiomatic argument structure for each individual verb, he attempted to identify general functions of noun phrases in what he called "thematic proto-roles". To accomplish this, Dowty drew from the theory of "family resemblance" and defined a set of attributes which would indicate such a thematic role. "The hypothesis put forth here about thematic roles is suggested by the reflection that we may have had a hard time pinning down the traditional role types because role types are simply not discrete categories at all, but rather are cluster concepts [...]" [Dowty, 1991, p. 571]

For example, [Dowty, 1991, p. 572] defines the property bundle of semantic Proto-Agents as follows:

a volitional involvement in the event or state

b sentence (and/or perception)

c causing an event or change of state in another participant

d movement (relative to the position of another participant)

e (exists independently of the event named by the verb)

Similar clusters of characteristics can also be defined for other proto-roles, like patients or themes, etc.

However, like most theories in linguistics, Semantic Roles remain a disputed topic in the field until today: "There may be general agreement on the cases (or Thematic Roles or Semantic Roles) [...], but there is substantial disagreement on exactly when and where they can be assigned and which [...] should be added, if any" [Palmer et al., 2010]. However, the general, agreed upon objective of semantic roles and similar concepts may be paraphrased as follows:

(2.5) **Semantic Roles are systematic abstractions of semantic functions that are attributed to the participants in a proposition expressed in human language.**

The volitional acting entity in a situation is abstracted, e.g. as "(Proto-)Agent"; regardless of the actual, concrete event denoted by the verb. Similarly, noun phrases which denote participants that undergo some state of change are captured as "proto-patients".

For some time now, there are lexical resources implementing one approach of structural semantic annotation. For example, the PropBank [Palmer et al., 2005], which adds "a layer of predicate-argument information, or semantic role labels, to the syntactic structures of the Penn Treebank"; or FrameNet [Baker et al., 1998], which aimed at producing "frame-semantic descriptions of several thousand English lexical items and backing up these descriptions with semantically annotated attestations from contemporary English corpora".

I focus here on the PropBank approach since the semantic role labeler I use in my architecture was trained on the German part of CoNLL '09 [Hajič et al., 2009], which consisted in semantic role labeling according to the PropBank scheme. In PropBank each verb, or rather each verb sense, is attributed with a so-called frame. This frame consists of a definition of the event denoted by that particalur verb sense and the semantic arguments associated with. For example, the first frame of the verb *to sink*[12] is analysed as follows:

sink.01 (cause to) go down, esp into water, downward motion

Roles:

Arg0-PAG causer of sinking (vnrole: 45.4-agent)

Arg1-PPT thing sinking (vnrole: 45.4-patient)

Arg2-EXT EXT

Arg3-DIR start point

Arg4-GOL end point, destination

Arg5-MNR instrument (vnrole: 45.4-instrument)

Applied to the sentence 2.3, this would lead to the following PropBank annotation:

(2.6) Severely damaged [$_{\text{Arg0-PAG}}$ by the hurricane], [$_{\text{Arg1-PPT}}$ the vessel] [$_{\text{Rel}}$ sank] [$_{\text{Arg4-GOL}}$ to the ground].

[12] `http://verbs.colorado.edu/propbank/framesets-english-aliases/sink.html`

Although the PropBank annotations are verb-specific, there are some generalizations in the first number of arguments as to what proto-role they denote, as the authors of PropBank write: "For a particular verb, Arg0 is generally the argument exhibiting features of a Prototypical Agent [...], while Arg1 is a Prototypical Patient or Theme" [Palmer et al., 2005, p. 75]. In their English PropBank Annotation Guideluines, Bonial et al. [2012] nevertheless propose generalizations even for the higher arguments:

Arg0 agent

Arg1 patient

Arg2 instrument, benefactive, attribute

Arg3 starting point, benefactive, attribute

Arg4 ending point

ArgM modifier

Rel Relation (can be a verb, noun, or adjective)

That is to say, even if the higher numbered argument's proto-role is semantically somewhat fuzzier than for argument zero and one, there are potentially generalizable patterns from which a model — trained on such annotated sentences — might still be able to detect task-supportive structures in the data.

In the following, some example sentences from the PropBank frames are presented[13]. Semantic roles are highlighted using the colors from the previous list. Note that only one relation is marked in the sentences, even if there are multiple. Since DAMESRL, the module I use for SRL tagging, only treats verbs as semantic roles distributing relations, I include only verbal "Rel"s in the examples:[14]

(2.7) [Arg0 Yasser Arafat] has [Rel written] [Arg2 to the chairman of the International Olympic Committee], asking him to back a Palestinian bid to join the committee.

(2.8) Once [Arg0 he] [Rel realized]

[13] accessible through this GitHub repository

[14] To me, not all annotations in PropBank are beyond all doubt; for example, in sentence 2.11 "an investment banker" is labelled as agent "maximzing" the the patient "shareholder value" — however, I would argue that it's rather the "alternative" that take proto-agentive role in maximizing the shareholder values.

[Arg1 that Paribas's intentions weren't friendly] , he said, but before the bid was launched, he sought approval to boost his Paribas stake above 10%.

(2.9) [Arg1 National Market System volume] [Rel improved] [Arg4 to 94,425,00 shares] [Arg3 from 71.7 million Monday] .

(2.10) [Arg0 The new round of bidding] would seem to [Rel complicate] [Arg1 the decision making] [Arg2 for Judge James Yacos] .

(2.11) The action followed by one day an Intelogic announcement that it will retain [Arg0 an investment banker] to explore alternatives "to [Rel maximize] [Arg1 shareholder value] ," including the possible sale of the company.

(2.12) [Arg0 He] [ArgM-mod would] scream and [Rel cut] [Arg1 himself] [Arg3 with rocks] .

Thanks to lexical resources such as the PropBank, a multitude of models aiming at labeling sentences with semantic roles are now available. For German, there is e.g. DAMESRL [Do et al., 2018], trained on the CoNLL '09 [Hajič et al., 2009] data (which implements PropBank style SRLs). This is also the semantic role labeler I employ in my GliBERT system. Since I treat the semantic role labeler essentially as a blackbox in my architecture and use it as an off-the-shelf SRL predictor, I will not go into details about the concrete implementation of DAMESRL, as well as the different approaches to automatic semantic role labeling in general — the interested reader is referred to Do et al. [2018] for the former and e.g. Palmer et al. [2010] for the latter.

The following list contains some examples of DAMESRL-labelled sentences stemming from the GliBERT corpus (see chapter 3). The sentences are represented vertically with the leftmost column being the actual sentence; each column represents one identified verb (B-V) and its predicted semantic roles, labelled using the BIO-schema[15] All the columns to the right of the verticalized sentence represent one argument-predicate structure for this sentence. Note that the labels are slightly differently labelled than in the PropBank, as listed before: `V` ("Verb") stands for `Rel` ("relation"), `A`*n* stands for `Arg`*n* (both are abbreviations for "argument").

[15]Introduced by [Ramshaw and Marcus, 1999], the BIO-schema is an established way of adding a label to each token in a sequence, indicating if it belongs to a certain subgroup, or chunk, of the sequence. For example, to mark the prepositional phrase in a syntagma like "He is running from the bear", one would mark the word beginning the PP with **B**, any other words inside the PP with **I**, and all other words outside of it, using **O**: "He[O] is[O] running[O] from[B-PP] the[I-PP] bear[I-PP]".

deISEAR

Ich	B-A0	O
fühlte	B-V	O
[MASK]	B-A1	O
,	I-A1	O
als	I-A1	O
ich	I-A1	B-A0
aus	I-A1	O
Versehen	I-A1	O
schlechte	I-A1	B-A1
Milch	I-A1	I-A1
getrunken	I-A1	B-V
habe	I-A1	O

MLQA

Welche	B-A1	B-A2
Positionen	I-A1	I-A2
muss	O	O
man	B-A0	B-A0
erreichen	B-V	O
,	O	O
um	O	O
die	O	B-A1
von	O	I-A1
Kaius	O	I-A1
angeordnete	O	I-A1
Position	O	I-A1
eines	O	I-A1
Läufers	O	O
einzunehmen	O	B-V
?	O	O

XNLI

Es	O	O	O
war	O	O	O
das	O	O	O

Wichtigste	O	O	O
was	B-A1	O	O
wir	B-A0	O	O
sichern	B-V	O	O
wollten	O	O	O
da	O	O	O
es	O	O	O
keine	O	B-A1	O
Möglichkeit	O	I-A1	O
gab	O	B-V	O
eine	O	B-A1	B-A3
20	O	I-A1	I-A3
Megatonnen	O	I-A1	I-A3
-	O	I-A1	I-A3
H	O	I-A1	I-A3
-	O	I-A1	I-A3
Bombe	O	I-A1	I-A3
ab	O	I-A1	O
zu	O	I-A1	B-A5
werfen	O	I-A1	B-V
von	O	I-A1	I-A5
einem	O	I-A1	I-A5
30	O	I-A3	I-A5
,	O	O	O
C124	O	O	O
.	O	O	O

A detailed discussion of the SRLs produced by DAMESRL will be done in chapter 5, section 6.1.4. In advance, it should be noted that the automatically predicted SRLs seem to capture semantic properties quite well (e.g. the second SRL-layer of the first sentence, where the predicate “getrunken”, the agent “ich” and the theme “schlechte Milch” are correctly identifed). However, there are some irritating results as well (e.g. the third layer in the last example, where there is an uninterpretable Arg5 “zu von einem 30”, interrupted by the predicate “werfen”).

In the next chapter 3, I will present the GerGLUE NLU corpus, and the subsequent chapter 4, the GliBERT architecture and actual implementation of SRLs into the BERT architecture are described.

3 Data Sets

3.1 GerGLUE

Because semantics is such a fuzzy, hard to formalize property of language, it is not easy to assess the capabilities of an architecture designed at solving problems related to meaning. In the data-driven NLP community today, it is common practice to measure the power of a model by measuring it's performance on some standardized data set. However, a model aiming at capturing semantics of human langauge "must be able to process language in a way that is not exclusive to a single task, genre, or dataset", as Wang et al. [2018] correctly point out.

To provide a standardized collection of datasets for the NLP community to compare different NLU-targeted models, Wang et al. compiled the General Language Understanding Evaluation benchmark, in short GLUE. It consists of nine data sets addressing different NLU problems; from acceptability tasks (is the phrase "Saw the man the dog." an acceptable English sentence?) to detecting textual entailment (is the meaning of "A boy is at the beach" entailed by the sentence "Two kids are building a sandcastle at the beach"?). See table 1 for a list of all GLUE data sets, their tasks and further characteristics.

Following Wang et al. [2018], I compile a NLU dataset collection for German, en-

[1]Wang et al. [2018] reformulate the original SQuAD task CITE of predicting an answer span in the context into a sentence pair binary classification task: They pair each sentence in the context with the question and predict whether or not the context sentence includes the answer span.

[2]Wang et al. [2018] combine several data sets into RTE; for data sets that have three labels — *entailment*, *neutral*, and *contradiction* — they collapse the latter two into one label *not_entailment*.

[3]In the original Winograd Schema Challenge CITE, the task is to choose the correct referent of a pronoun from a list. Wang et al. [2018] reformulate this to a sentence pair classification task, where the original sentence is paired with the original sentence with each pronoun substituted from the list and then predicting whether the substituted sentence is entailed by the original one.

Data Set	NLP Task	ML Task	# Examples	Splits
	Single-Sentence Tasks			
CoLA	Acceptability	Binary Classification	8.5k/1k	train/test
SST-2	Sentiment Analysis	Binary Classification	67k/1.8k	train/test
	Sentence Pair Tasks			
MNLI	Natural Language Inference	Multi-Class Classification	393k/20k	train/test
MRPC	Paraphrase Identification	Binary Classification	3.7k/1.7k	train/test
QNLI	Question Answering	Binary Classification[1]	105k/5.4k	train/test
QQP	Paraphrase Identification	Binary Classification	364k/391k	train/test
RTE	Natural Language Inference	Binary Classification[2]	2.5k/3k	train/test
STS-B	Sentence Similarity	Regression (1 - 5)	7k/1.4k	train/test
WNLI	Coreference Resolution	Binary Classification[3]	634/146	train/test

Table 1: Original GLUE data sets and tasks (following the table from Wang et al. [2018]).

compassing several tasks. Unfortunately, the data availability for German is not as extensive as for English, so that not for all tasks in the original GLUE there was a German counter part; e.g. I couldn't find a sentence similarity dataset.

Data Set	NLP Task	ML Task	# Examples	Splits
	Single-Sentence Tasks			
deISEAR	Emotion Detection	Multi-Class Classification	1,001	-
SCARE	Sentiment Analysis	Multi-Class Classification	1,760	-
	Sentence Pair Tasks			
MLQA	Question Answering	Span Prediction	509/4,499	dev/test
PAWS-X	Paraphrase Identification	Binary Classification	49,402/2,001/2,001	train/dev/test
XNLI	Natural Language Inference	Multi-Class Classification	2,489/5,009	dev/test
XQuAD	Question Answering	Span Prediction	1,179	-

Table 2: GerGLUE data sets and tasks.

3.1.1 General Issues

There are a few remarks and strategic decisions that apply to all collected dataset in GerGLUE:

(1) All of the datasets except for deISEAR are not monolingual, i.e. German, sources, but bi- or multilingual corpora. To compile a German GLUE corpus I only use the German subset of those corpora. For example, the MLQA data set provides all 49 combinations of the languages it contains: Context in Arabic, question in

Hindi; context in English, question in Spanish, etc. Also in this case, I choose only the German-German part of the data set for my corpus.

(2) The data sets I chose for my GerGLUE corpus are being provided in different modes: While three of the corpora, namely MLQA, PAWS-X, and XNLI, come with predefined splits, the others are made available in one set, without defining training, development, and test parts. In the latter case, I split the data sets into train, development, and test splits using a 0.7, 0.15, and 0.15 portion, respectively. Interestingly, the data sets that come with splits, mostly provide only a development and test portion. To ensure that my results are comparable with those that the authors of the different data sets report, I leave the test split as it is, and split the development set into a train and development set, implementing a 85:15 ratio. The only dataset having explicitly predefined training, development, and test splits is PAWS-X.

(3) Several of the datasets were constructed by translating existing monolingual English sources (semi-)automatically into the different target languages. As I show in chapter 5, this does not come without introducing noise into the data.

3.2 Corpora

In this section, I give a detailed description of the selected data sets in alphabetical order: What kind of task is addressed, what is the text variety, and report some statistical measures, e.g. the average length of examples in the different sub-sets (Training, Development, Test). I also document the SOTA benchmark results that the authors specify on their datasets; anyhow, the results of GLiBERT will not be compared directly with them since my focus is not on dataset SOTAs but differences between GliBERT with and without SRL information. Nevertheless, for most datasets, the GliBERT performance is on par with what the authors report.

3.2.1 deISEAR

This data set addresses the task of Emotion Recognition, a sub-task of Sentiment Analysis [Cambria et al., 2017]. Technically, it is a sequence classification problem: Given a sequence of tokens $x_1 \dots x_n$, predict the correct label y from a fixed set of emotions Y. Or, in a more natural way of speaking, to automatically determine what emotion a certain statement expresses. Following the original study "International Survey on Emotion Antecedents and Reactions" [Scherer and Wallbott, 1994], Troiano et al. [2019] constructed the deISEAR data set for German:

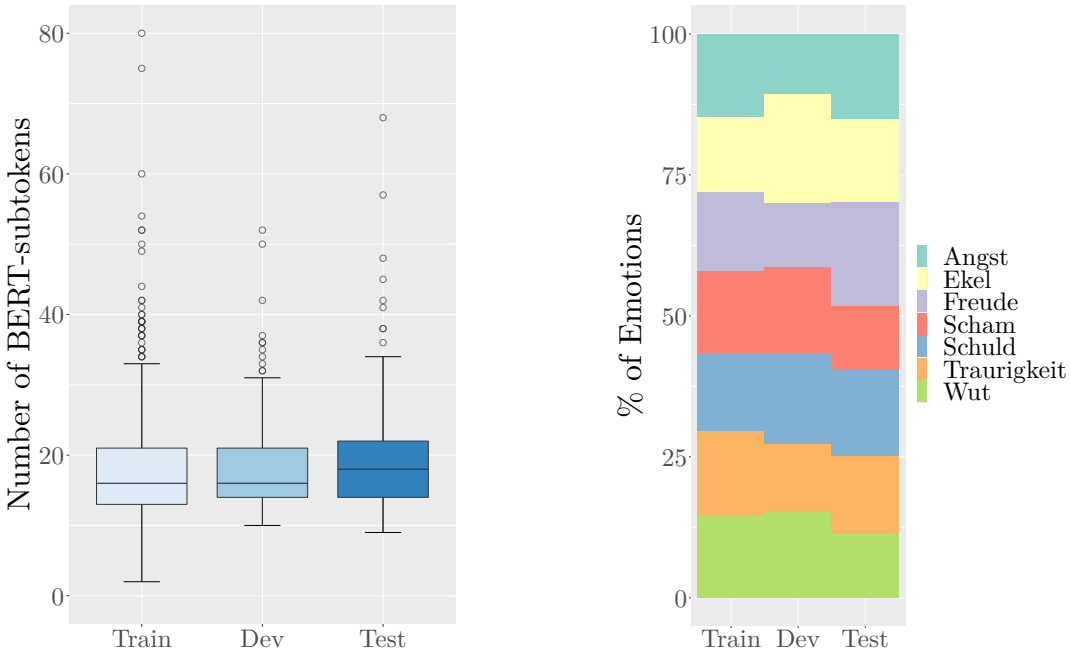

Figure 2: Left: Length of subtokenized deISEAR sentences. Note that one extreme outlier in the development set comprising 300 BERT-subtokens is not included in the plot. **Right**: Label distributions of deISEAR datasets.

In a first step, the authors presented annotators with one of seven emotions, and asked them to come up with a textual description of an event in which they felt that emotion. The task was formulated as a sentence completion problem, so the annotators which were recruited via a crowdsourcing platform, had to complete sentences having the following structure: "Ich fühlte *emotion*, als/weil/dass ...". The seven emotions that were given for which the descriptions had to be constructed were: Traurigkeit, Ekel, Schuld, Wut, Angst, Scham, Freude.[4]

The second phase of the data generation process comprised of re-labeling the generated sentences such that five annotators each re-annotate every sentence. The emotion word was omitted, and the annotators had the list of seven emotions at hand. Troiano et al. report that for approximately half of all sentences, the inter-annotator agreement was perfect; i.e. each of the five annotators attributed the same emotion. However, some emotions seem to be prone for not being clearly separable: Shame, for example, gets confused with guilt and vice versa in 17% and 19% of the cases, respectively. Remarkably, this pattern is not visible for the GLiBERT predic-

[4]Interestingly, out of these seven emotions, six represent rather negative emotions — only *Freude* is a clearly positive sensation. Maybe negative emotions are more lucidly detactable (by humans and/or machines) than positive ones and therefore the study focused on it.

tions, as is exemplified for one ensemble in table 3: Shame and guilt are reliably told apart by the model, however, it has more general problems in recognizing sadness and anger.

		Predicted						
		Angst	Ekel	Freude	Scham	Schuld	Traurigkeit	Wut
True	Angst	17	0	0	0	4	0	2
	Ekel	0	18	0	0	2	0	2
	Freude	1	0	22	0	1	2	2
	Scham	0	4	2	10	1	0	0
	Schuld	0	2	1	3	17	0	0
	Traurigkeit	2	1	0	2	2	13	1
	Wut	2	1	1	1	2	2	8

Table 3: Confusion matrix for the best deISEAR ensemble (α +SRL subtokenized zeros FFNN head). Interestingly, the observed confounding of Shame and Guilt in the human re-annotations is not present for the GliBERT predictions. However, the overall most difficult, i.e. most often confused emotions for the reported GliBERT head seem to be sadness and anger.

The authors do not report any metric regarding the reliability of their labels based on the human re-classifications, but since they provided the complete results of phase 2, I was abe to compute the Fleiss' κ, a standard metric for estimating annotator agreement, and thus, the reliability of these labels; the computed value equals to 0.66, which corresponds to "substantial" agreement in the interpretation scale for th variable, proposed by Landis and Koch [1977].

Following are seven example sentence randomly picked out of the deISEAR corpus, one for each emotion.

(3.1) Ich fühlte **[Traurigkeit]**, als mein Laptop kaputt ging und die Garantie schon abgelaufen war.

(3.2) Ich fühlte **[Scham]**, weil mir mal beim Urlaub das Geld ausging.

(3.3) Ich fühlte **[Freude]**, als ich mit meinen Arbeitskollegen ohne Ende Witze gerissen habe.

(3.4) Ich fühlte **[Angst]**, als der Chef sagte dass Mitarbeiter gekündigt werden müssen.

(3.5) Ich fühlte **[Wut]**, als ich die Nachricht gelesen habe, dass der VfB Stuttgart nicht in neue Spieler investieren wird.

(3.6) Ich fühlte **[Ekel]**, als ich verschimmeltes Essen im Kühlschrank gefunden

habe.

(3.7) Ich fühlte **[Schuld]**, dass ich meinen besten Kumpel versetzt habe.

Now it is up to you: Here are four sentences with masked emotions — try to assign what you thin is the correct one. The possible emotions are **Angst, Ekel, Freude, Scham, Schuld, Traurigkeit, Wut**.[5]

(3.8) Ich fühlte **[?]**, als ich meine kleine Tochter zum Schwimmen abgeholt habe.

(3.9) Ich fühlte **[?]**, als meine Mutter mich zur Schule begleiten musste als ich die schule geschwänzt hatte

(3.10) Ich fühlte **[?]**, als die Ärzte im KH bei meiner im sterben liegenden Großmutter einen künstlichen Zugang legen wollten um die Schilddrüsenmedikamente zu verabreichen.

(3.11) Ich fühlte **[?]**, als eine Feuerwerksrakete in Richtung meiner Kinder abgefeuert wurde und mein kleiner weinend davon lief.

deISEAR is one of the data sets that are made available without any pre-defined training/development/test splits. Therefore I shuffle all 1,001 sentences and split with a 70:15:15 ratio, resulting in a training set of 700, a development set of 150 and a test set of 151 sentences.

In figure 2, the length of the sentences and the label distributions in the three data sets are plotted. While the data sets were created randomly, there are some peculiarities observable: The lengths of the sentences in the training set show a greater variation compared to the development and test set than one would except.

Troiano et al. [2019] train a maximum entropy classifier with L2 regularization with boolean unigram features on the original ISEAR corpus (7,665 instances). Since the original ISEAR study and data collection was carried out in English, they then machine translate the 1,001 deISEAR examples and evaluate on them. Using this strategy, the authors accomplish an average micro F_1 of 47. (Note: micro F_1 in settings where each example gets exactly one label assigned is the same as accuracy.)

3.2.2 MLQA

The term question answering subsumes several related tasks or problems: In it's most general form, question answering refers to the ability to give a meaningful an-

[5] 3.8: Freude, 3.9: Scham, 3.10: Wut, 3.11: Angst

swer to any possible inquiry. This variety is normally termed *open-domain question answering* and models addressing this task require a whole pipeline of algorithms in the background to produce acceptable results (cf. Chen and Yih [2020]). Another form is so called *multi-choice question answering*, where the task for the model is to select the correct answer out of a list of options given the answer (cf. Welbl et al. [2017]). The subsort of question answering MLQA addresses is so called *span prediction question answering*. The goal here is to extract the correct answer span out of a context text given the question.

Lewis et al. [2019] compiled the MLQA data set using Wikipedia articles. First, they "automatically identify sentences from [...] articles which have the same or similar meaning in multiple languages."[6] Secondly, they crowdsourced questions for the English paragraphs, let them humanly translate into the target languages, and finally annotate the answer spans in the corresponding paragraphs.[7]

Following are five random examples out of the MLQA data set:

(3.12) **Context**: Rita Sahatçiu Ora (* 26. November 1990 in Priština, SFR Jugoslawien) ist eine britische Sängerin und Schauspielerin kosovarischer Herkunft. Von 2010 bis 2016 stand sie bei Jay Z und Roc Nation unter Vertrag. Seit 2017 steht sie bei Atlantic Records unter Vertrag.

Question: Wann wurde Rita Sahatçiu Ora geboren?
Answer: 26. November 1990

(3.13) **Context**: Während Somalia an militärischer Stärke gewonnen hatte, wurde Äthiopien aufgrund innenpolitischer Umstände geschwächt. 1974 hatte die Derg-Militärjunta den abessinischen Kaiser Haile Selassie gestürzt, sich aber bald in interne Machtkämpfe verstrickt, woraufhin es zu Unruhen kam. In verschiedenen Landesteilen waren Derg-feindliche und separatistische Kräfte aktiv. Das regionale Machtgleichgewicht hatte sich zugunsten Somalias verschoben.

Question: Zu wessen Gunsten verlagerte sich die Balance of Power?
Answer: Somalia

(3.14) **Context**: Das Johnston-Atoll verlassend, drehte John nach Nordwesten ab

[6] However, taking the sometimes huge contexts into account, I think the better formulation would have been "paragraphs" with similar meaning, instead of "sentences".

[7] If this was done manually or by implementing some other techniques is, unfortunately, not reported. The presence of sometimes strange offsets (included commas, missing prepositions as in example 3.14 etc.) seem to indicate a not fully hand-made annotation — at least to me.

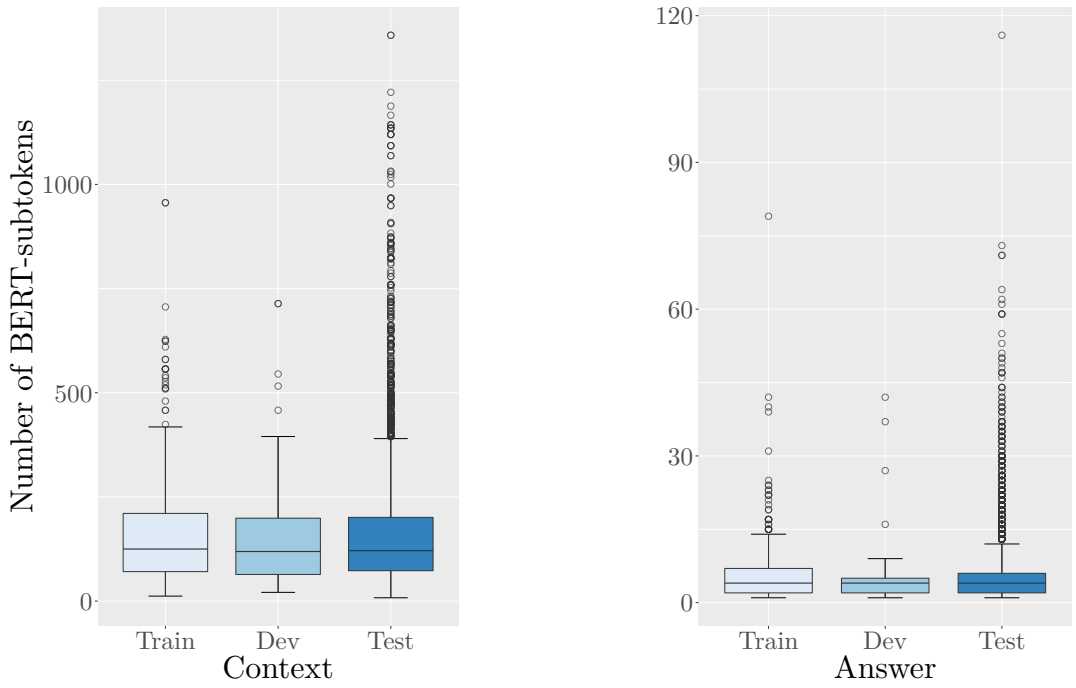

Figure 3: Left: Length of subtokenized MLQA contexts. **Right**: Length of subtokenized MLQA answers. Note the difference in y-axis scaling between the two plots: The contexts are longer by orders of magnitude to the answer spans. In fact, most answer spans consist of only a handful of (subtokenized) words.

und begann sich erneut zu intensivieren, als die Windscherung nachließ. Am 27. August Ortszeit erreichte John einen sekundären Höhepunkt mit Windgeschwindigkeiten von 210 km/h. Kurz darauf überquerte John die Datumsgrenze bei etwa 22° nördlicher Breite und gelangte in das Beobachtungsgebiet des Joint Typhoon Warning Center (JTWC) auf Guam. Durch seinen Aufenthalt im westlichen Pazifischen Ozean wurde Hurrikan John zum Taifun John. Kurz nach dem überschreiten der Datumsgrenze schwächte sich John wieder ab und die Vorwärtsbewegung kam fast zum Stillstand. Am 1. September hatte sich Taifun John zum tropischen Sturm abgeschwächt und veränderte seine Position knapp westlich der Datumsgrenze kaum. Dort blieb der Sturm die nächsten sechs Tage, während der John eine mehrere Tage andauernde Schleife entgegen dem Uhrzeigersinn zog, bis am 7. September ein Trog in die Gegend gelangte und John schnell nach Nordosten abzog. Am 8. September überquerte John die Datumslinie wieder nach Osten und gelangte erneut in den Zentralpazifik.Dort angelangt erreichte John seinen tertiären Höhepunkt mit Windgeschwindigkeiten von 145 km/h als starker Kategorie-1-Hurrikan, ein gutes Stück nördlich der Midwayinseln. Der Trog nahm die Struktur von John auseinander und das

kalte Wasser des nördlichen Zentralpazifiks tat sein Übriges. Am 10. September wurde die 120. Sturmwarnung zu John ausgegeben, mit der das System als außertropisch erklärt wurde, etwa 1600 km südlich von Unalaska.

Question: Wo wurde John zum Taifun?
Answer: westlichen Pazifischen Ozean

To demonstrate that this is by no means a trivial task — at least for us humans —, try to identify the correct answer span[8] for the following context-question pair:

(3.15) **Context**: Das britische Parlament genehmigte Königin Victoria, ihrer Tochter als Mitgift 40.000 Britische Pfund (in heutiger Kaufkraft 3.662.803 Pfund) zu zahlen und legte die jährliche Apanage der Prinzessin auf 8000 Pfund fest. König Friedrich Wilhelm IV. gewährte seinem Neffen ein jährliches Einkommen von 9000 Talern. Das Einkommen des Prinzen war damit nicht ausreichend, um die Kosten eines standesgemäßen Haushaltes zu decken, und einen Teil der Haushaltskosten würde zukünftig Prinzessin Victoria aus ihrem Vermögen tragen müssen. Der zukünftige Hofstaat des jungen Paares wurde von der preußischen Königin und der zukünftigen Schwiegermutter Prinzessin Auguste ausgewählt. Die beiden Frauen entschieden sich überwiegend für Personen, die bereits länger im Hofdienst standen und damit deutlich älter als das prinzliche Paar waren. Prinz Alberts Bitte, seiner Tochter doch wenigstens zwei gleichaltrige und britische Hofdamen zu gewähren, wurde nicht entsprochen. Als Kompromiss wurden mit den Komtessen Walburga von Hohenthal und Marie zu Lynar zwei Hofdamen gewählt, die Prinzessin Victoria wenigstens altersmäßig entsprachen. Immerhin konnte Prinz Albert Ernst von Stockmar, den Sohn seines jahrelangen Beraters Christian Friedrich von Stockmar, als persönlichen Sekretär der Prinzessin durchsetzen. Prinz Albert, der überzeugt davon war, dass der preußische Hof die Einheirat einer britischen Prinzessin als Bereicherung und Ehre ansähe, bestand außerdem darauf, dass Prinzessin Victoria den Titel einer Princess Royal of the United Kingdom of Great Britain and Ireland beibehielt. An dem überwiegend antibritisch und prorussisch eingestellten preußischen Hof löste dieser Schritt allerdings nur Verärgerung aus.Der Hochzeitsort war Anlass für weitere Meinungsverschiedenheiten. Für das preußische Königshaus war es selbstverständlich, dass ein Prinz, der als zweiter in der Thronfolge stand, in Berlin heiratete. Letztlich konnte sich aber Königin Victoria durchsetzen, die

[8] überwiegend antibritisch und prorussisch

> als regierende Monarchin für sich in Anspruch nahm, ihre älteste Tochter in ihrem Land zu vermählen. Das Paar trat schließlich am 25. Januar 1858 in der Kapelle des St James's Palace in London vor den Traualtar.
>
> **Question**: Was war die Position der Berliner Gerichts gegenüber Großbritannien und Russland?

MLQA is a set that comes with pre-defined development and test splits. The test set contains 4,499 examples, while the dev set is made up of 509 examples. As for all the other pre-splitted data sets, a training set is not part of the splits, this has to do with the approach that Lewis et al. [2019] intend to use their data set, which will be explained later on. In a first set up, I treat the MLQA development set as training set and split off a 15% portion of it as development set, resulting in 432 training instances and 77 development examples. In a second set up I shuffle the pre-defined splits and create a more suiting 70:15:15 set ratio; resulting in the following numbers of examples per set: Training 3,506, development 751, test 751.

As can bee seen in the left figure of 3, MLQA comprises quite a high number of examples — 868, to be precise — which do not fit subtokenized into a normal BERT model. However, those examples where the answer span lies inside the maximum BERT sequence are simply cut off after the maximum length and are kept in the data set for the experiments. Only the 22 examples where the answer span lies partially or fully outside the maximum length were dropped.

Lewis et al. define two tasks they use to evaluate model performance on MLQA: The first one, cross-lingual transfer (XLT), means training models in English, conducting model selection on the development set, and evaluating on the test sets for the various languages. The second case, generalized cross-lingual transfer (G-XLT), is not of interest to this thesis, since it involves a mixing of languages: the context is presented in one language and the question in another. The authors include two models for the zero-shot transfer approach: multilingual BERT and XLM and use SQuAD (100,00 instances) as the training set.

The best results for German in the XLT mode Lewis et al. reoprt, is a 47.6% exact match accuracy, achieved by XLM. However, as the training procedure and amount of traning data differs quite drastically from my set up, it's not possible to directly compare their benchmark with my results. Detailed GliBERT results on the GerGLUE question answering tasks are reported in chapter 5, section 5.3.

3.2.3 PAWS-X

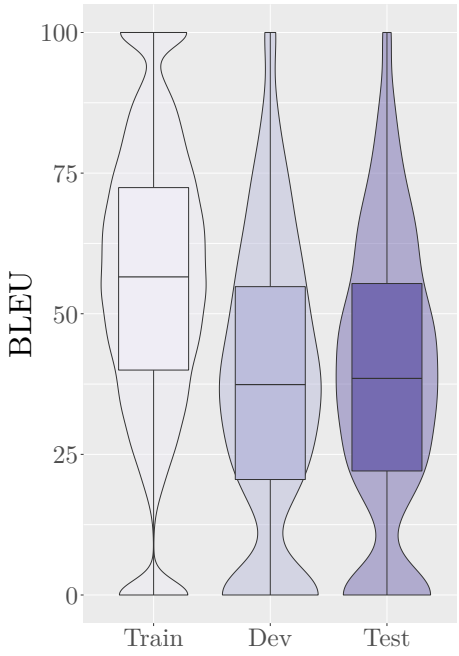

Figure 4: BLEU scores of the PAWS-X data sets. Clearly visible is the diffence between the training set on the one hand, and the development and test set on the other:

The PAWS-X corpus Yang et al. [2019] was compiled to provide a multilingual source for training models that address the NLU problem of paraphrase identification. This task is typically formalized as a binary classification task: given two sentences S_1 and S_2, the model must determine whether they convey the same meaning — performing well on such a dataset is often taken as an indicator of reliably capturing the semantics of an utterance [McKeown, 1980]. Since most corpora for this task are available only in English the authors compiled this corpus by humanly translate a subset of the original English PAWS corpus Zhang et al. [2019a].

Concretely, Yang et al. "translate the Wikipedia portion of the original PAWS corpus from English" to six target languages. Out of these, I include the German part into GerGLUE. These corpora were created by hiring human translators to translate the original PAWS development and test sets into the six languages, while for the training set a "neural machine translation" system was employed. The following examples are randomly drawn out of the whole German part of PAWS-X:

(3.16) Die Familie zog 1972 nach Camp Hill, wo er die Trinity High School in Harrisburg, Pennsylvania, besuchte.

1972 zog die Familie nach Camp Hill, wo er die Trinity High School in Harrisburg, Pennsylvania, besuchte.

True

(3.17) Prestige gehört der verheirateten Kiribati-Frau an, sie steht jedoch beträchtlich unter der Autorität ihres Mannes.

Die verheiratete Kiribati-Frau ist ein inhärentes Prestige, aber sie steht unter der Autorität ihres Mannes.

True

(3.18) Die österreichische Schule geht davon aus, dass die subjektive Entscheidung

des Einzelnen, einschließlich des individuellen Wissens, der Zeit, der Erwartungen und anderer subjektiver Faktoren, alle wirtschaftlichen Phänomene verursacht.

Die österreichische Schule geht davon aus, dass die subjektive Entscheidung des Einzelnen, einschließlich des subjektiven Wissens, der Zeit, der Erwartung und anderer individueller Faktoren, alle wirtschaftlichen Phänomene verursacht.

False

(3.19) "Es ist der vierte Track und die dritte Single aus ihrem Durchbruch ""Smash"" (1994)."

Es ist der vierte Track und die dritte Single von ihrem Durchbruchalbum “ Smash ” (1994).

True

(3.20) Die Mannschaft reagierte auf die Änderungen im nächsten Spiel am selben Abend am 19. Februar.

Die Mannschaft reagierte auf die Änderungen im selben Spiel am nächsten Abend des 19. Februars.

False

You are invited to test the examples yourself and predict if the following sentence pairs are in fact rephrasings of the same semantic content. The gold labels are in this footnote[9].

(3.21) Die Single wurde am 12. Oktober 2012 im italienischen Radio Airplay gespielt und am 3. Dezember 2012 weltweit verschickt.

Die Single wurde am 12. Oktober 2012 nach Italien zu Radio Airplay geschickt und am 3. Dezember 2012 weltweit veröffentlicht.

(3.22) Lloyd gründete und leitete sein Unternehmen, um mit dem Verkauf von Spielzeug und Geschenken zu beginnen, und er erweiterte das House of Lloyd, mit Sitz in Grandview, Missouri, während das Geschäft mit den Geschenken wuchs.

Lloyd gründete und leitete sein Unternehmen zum Verkauf von Spiel- und

[9] 3.21: False, 3.22: True

> Geschenkwaren und erweiterte das in Grandview, Missouri, liegende House of Lloyd mit dem Wachstum des Marktes für Geschenkwaren.

The average lengths of the sentences is very regularly distributed with averages of 27.5 (σ 8.6), 27.7 (σ 8.4), and 28.1 (σ 8.4) for the subtokenized training, development, and test sets. Also, the lenght differences between the first and second sentence in each example is virtually nonexistent, which is not surprising if the creation of the original PAWS dataset is taken into account: "Our automatic generation method is based on two ideas. The first swaps words to generate a sentence pair with the same BOW, controlled by a language model. The second uses back translation to generate paraphrases with high BOW overlap but different word order" [Zhang et al., 2019a].

Really striking about PAWS-X is the enormous difference in examples per set — the training set encompasses twelve times as many examples as the development and test set combined: There are 48,977 instances in the training set versus approx. 2,000 examples in the development and test set, each. The labels are very uniformyl distributed over all sets, each of the three labels comprising virtually one third of the examples.

Since the training data are solely machine-translated while the development and test data are human-translated, there needs to be some clarification as to how differently those sets are. One measure to capture similarities between sentences is the **Bi**l**i**ngual **E**valuation **U**nderstudy (BLEU) score Papineni et al. [2002]: This score measures the overlap of n-grams between two sentences, thus asserting the similarity between them.[10] The BLEU score is a value between 0 (no n-gram overlaps) to 1 (perfect n-gram overlaps), where a BLEU score of 1 means that the two sentences are identical. As for other measures, like accuracy e.g., the value is sometimes multiplied by 100 for better readability, which I will also do here.

The BLEU scores indicate that the sentence pairs in the training set are much more similar to each other than in the development and test set. Taken into account how the data sets were generated, this makes actually sense, however: While the development and test sets were translated from English to German by humans, the huge training set was automatically translated. Since the original differences in the sentence pair might well have been rather subtle, it is no surprise that an algorithm might exhibit difficulties in grasping those differences; resulting in similar

[10] BLEU was developed to automatically rank machine translation candidate sentences by comparing them to human translations — the higher the BLEU metric, the more similar to the human sentence a translation is, and therefore, at least often, perceived as more natural.

translations for two similar sentences. Note that due to the difficulties mentioned before, the automatic translation resulted in 3,209 sentence pairs (6.6% of all the sentence pairs) with a BLEU score of 100.00 in the training set — which means they are identical.[11]

Yang et al. [2019] achieve their best result — 89.2% accuracy for German — employing the following model architecture: They train a multilingual BERT on all languages, including the original English pairs and the machine-translated data in all other languages and evaluate on the individual languages.

During the preprocessing of this data set, the following considerations are taken into account: In the predefined development and test splits, there are some examples where one or both sentences consist only of the string "NS". I decided to not include this examples into the data used for training and evaluating my models, since those examples don't contribute any useful features for the model.[12] Further, some examples consist of empty strings; I treat those the same way as the examples mentioned before. Upon non-systematic manual inspection, there were indications of problematic data in PAWS-X, which will be examined in more detail rigor in chapter 5, section 6.1.

3.2.4 SCARE

SCARE is a dataset assembled for sentiment analysis, i.e. the task of assessing the transmitted emotion of an utterance. The data collected for the **S**entiment **C**orpus of **A**pp **Re**views stems, as the name suggests, from product reviews in the Google Play Store in German. Sänger et al. [2016] describe the language contained in the dataset the following way:

> Unlike product reviews of other domains, e.g. household appliances, consumer electronics or movies, application reviews of-

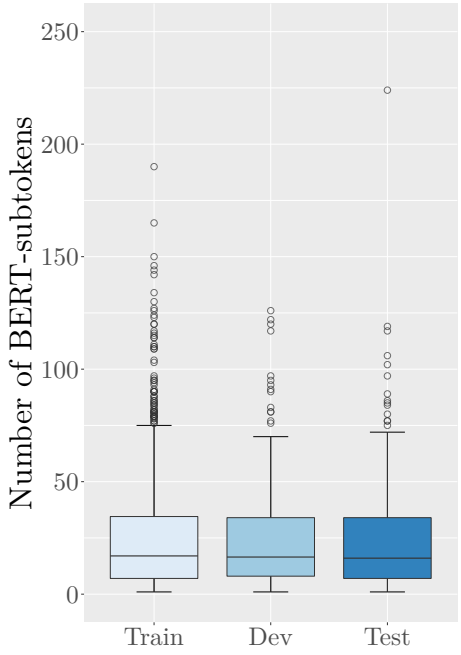

Figure 5: Length of subtokenized SCARE reviews. Mostly, the reviews are rather short, with an average of 25.29 subtokens over all sets, but there is quite a number of outliers — indicating further that it is quite a heterogeneous data set, also concerning this aspect. Note

[11] I reported this to the authors of the corpus, but didn't receive an answer from them.

[12] The authors don't comment on these obscure sentences, so I do not know what was the reasoning behind including these into the data sets.

> fer a couple of peculiarities which deserve special treatment: The way in which users express their opinion in app reviews is shorter and more concise than in other product reviews. Moreover, due to the frequent use of colloquial words and a flexible use of grammar, app reviews can be considered to be more similiar [sic] to Twitter messages ("Tweets") than reviews of products from other domains or platforms [...].

In contrast to other datasets, e.g. [Socher et al., 2013; Go et al., 2009], that attributes one sentiment label to a whole text (may it be a review, a tweet, etc.), Sänger et al. [2016] annotated their dataset on an aspect-based manner: Not each review gets labelled for a certain polarity — i.e. *positive*, *negative*, or *neutral* — but what the authors call *aspects* and correlating *subjective phrases*. An aspect is a category, feature, or topic that is being talked about: It may be the application itself, parts of the application, a feature request regarding the application, etc. A subjective phrase "express[es] opinions and statements of a personal evaluation regarding the app or a part of it, that are not based on (objective) facts but on individual opinions of the reviewers" [Sänger et al., 2016, p. 1116]. In other words, aspects are facts about the App and subjective phrases are user opinions regarding them.

This fine level of annotations leads often to several annotations per review, the sentiment of which may not always match. As illustration, consider the following review:

(3.23) guter wecker... || vom prinzip her echt gut...aber grade was die sprachausgabe betrifft noch etwas buggy....[13]

There are the following annotations for the aspects and their corresponding subjective phrases annotated in SCARE (aspects are bold, the subjective phrase is italic and the polarity is normal):

- **Wecker**, *guter* → positive
- **Prinzip**, *echt gut* → positive

[13] The "||" denotes that the text left of it is the user given "title" of the review, and the part on the right is the actual review.

– **Sprachausgabe**, *etwas buggy* → negative

As is clear from this example, in a given review there may be several aspects with a corresponding subjective phrase per review. It is well possible, as in the provided example, that the sentiment of these is not always the same.

For integrating the SCARE corpus into my GerBLUE corpus, I need to prepare the data, so it can be handled by the model architecture; following the original GLUE sentiment task, the model needs only to predict one sentiment label for each example. Since there exist mostly multiple annotations for each review in this data set, the data needs to be pre-processed in a way, so that there is one review-label per example.

To generate the review-label, I carry out a majority class decision: The polarity that is most often annotated for a given review is also the review-label; in the example above comprising three aspect-subjective polarities, two of them positive, one negative, the whole example would be labelled as positive. If there is no majority label, the review-label is set to "neutral". This is also the chosen strategy for 51 reviews that had no labels at all; an example of such a review is the following one:

(3.24) Ich bin die erfuinderin || Ich bin die erfunden!!!!!!!!!!!!!!!!!!!!!!!!

Figure 6 shows the outcome of this preprocessing and the confidence of the such generated labels for the examples. Below are 5 sentences with their generated labels — as is apparent, at least for a human judge, the labels seem to capture the overall sentiment of the sentences quite good.

(3.25) Ganz okay || Hatte ein Problem mit der APP aber die updates neu installiert und jetzt gehts wieder vorläufig mal Und Ordner wären schön wenn man diese erstellen kann

Neutral

(3.26) Ssssereeehhhr gut

Positive

(3.27) Wie kann man so eine gute app machen und dann nicht auf wvga anpassen. Weg mit den matschtexturen und vor allem dem Icon x-(

Negative

(3.28) spitze || Daran sollte sich MS ein Beispiel nehmen!

Positive

(3.29) Läuft nicht auf dem Acer A500 || Stürzt leider immer beim Abspielen eines Videos ab. Honeycomb 3.2

Negative

Following are three examples to test for yourself with the gold labels in this footnote[14].

(3.30) Für android einfach schwach || Ich habe die app, weil ich selbst im verein spiele!app stürzt sehr oft ab,ob jetzt kurz nach starten der app oder mitten beim suchen kommt die meldung fussball.de angehalten und man bekommt die möglichkeit ein bericht zu senden oder auf ok zu drücken um nochmal die app zu starten einfach schwach.die 3 sterne sind eigentlich auch zu viel aber weil wenigstens kleine updates kommen,die zwar nicht was ändern wollen wir mal nicht so sein.

(3.31) For Free is ok || Es fehlen Hausnummern...also scheint die Genauigkeit der Karten nicht absolut zu sein...weiteres nach längerer Testzeit

(3.32) Alle Funktionen, die .... || ... ich als Langschläfer brauche.

Analyzing the label distributions after turning the original aspect-based SCARE into an emotion detection dataset reveals that the positive class amasses the most examples on itself (1,071), while the neutral class only comprises 193 items out of a total of 1,760 examples (cf. figure 6). However, as the analysis of the GliBERT predictions in chapter **??** discloses, the models do not simply favour the majority class.

The peculiar `title || comment` structure of the reviews suggests that it could be encoded as sentence pair, in the style of PAWS-X and XNLI. But by examining the data, a lot of reviews are actually of the following structure:

(3.33) Erkennt Drucker nicht... || ...,wenn er als Netzwerkdrucker an der FritzBox hängt, und eine manuelle Konfiguration über IP-Adresse ist nicht möglich.

i.e. that the title and the text of the review form one continuous sequence, rendering a introduction of the [SEP] special token rather superfluous, if not harmful. Nevertheless, the separator symbol arguably "||" introduces some noise, especially for the ParZu parser, which needs to parse the sequence.

SCARE is a rather small dataset with a total of 1,760 examples. Since the dataset

[14] 3.30: Negative, 3.31: Positive, 3.32: Positive

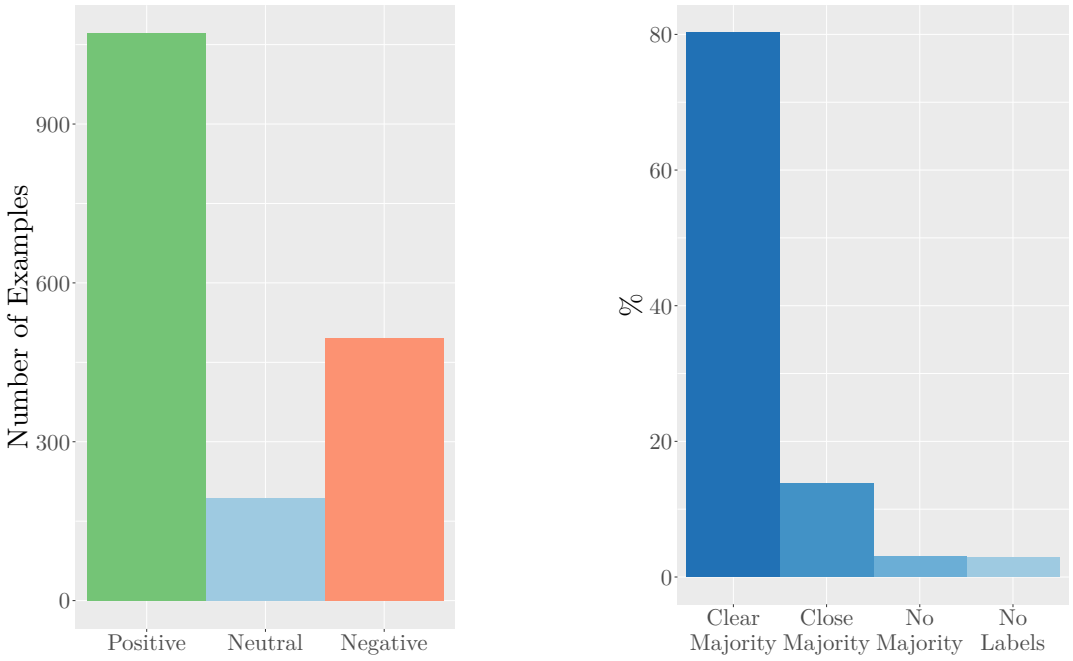

Figure 6: Left: Number of examples per label after heuristically computing them in the SCARE dataset. **Right**: Statistics of label generation. For most of the examples, there was a clear majority decision as to which label should be chosen. *Close Majority* means the majority vote was off by 1. The *No Majority*/*No Labels* portions in the graph were labelled *neutral* by default, while *Clear Majority*/*Close Majority* were labelled according to the majority vote decision.

comes without predefined splits, I split it regularly into a 70% training, 15% development, and 15% test set. As Sänger et al. stated in the passage quoted at the beginning of the description of this section, online reviews tend to be rather short, colloquial text snippets, which is confirmed by plotting the number of BERT subtokens in figure **??**. However, since the texts contain many typos, doublings of characters, and other distortions, the number of "actual" words is considerably lower on average than the number of BERT subtokens: Take a sentence such as "Unterirdisch, laaaaangsaaaaam mit WLAN seid neuestem nur" which would be subtokenized by German BERT as:

["Unter", "##ird", "##isch", ",", "la", "##aa", "##aa", "##n", "##gs", "##aa", "##aa", "##am", "mit", "W", "##LA", "##N", "sei", "##d", "neues", "##tem", "nur"]

— a list of 21 subtokens compared to the de facto 8 tokens present in the sentence.

Sänger et al. [2016] don't predict a sentiment for each instance, but predict fine-grained aspect and subjective phrase spans using a CRF-based model. They report

results for exact matches as well as partial matches. For the aspects, they achieve an exact match F1 score of 62% and 63% for subjective phrases, respectively. Since predicting fine-grained aspect and subjective phrase spans is much more difficult than extrapolating an overall sentiment of the same utterance, a comparison between the outcomes of the two tasks are not really comparable.

3.2.5 XNLI

The name XNLI indicates already what kind of NLU problem this dataset is targeted at: **N**atural **L**anguage **I**nference is the task of determining whether a hypothesis h can reasonably be inferred from a premise p, i.e. if h "follows" from p. It must be emphasized that although the wording suggests it, NLI is quite different from formal logic deduction in that its emphasis "is on informal reasoning, lexical semantic knowledge, and variability of linguistic expression, rather than on long chains of formal reasoning" [MacCartney and Manning, 2009].

Conneau et al. [2018] built the XNLI corpus by employing professional translators to translate 7,500 English sentence pairs from the Multi-Genre Natural Language Inference (MultiNLI) corpus Williams et al. [2017] into fifteen languages. First, they randomly sample 750 examples from each of the ten text source used in MultiNLI, which is in English, and then let the same MultiNLI worker pool generate three hypothesis for each sentence, one for each possible label (*entailment*, *contradicion*, *neutral*). Each sentence pair was then assigned a gold label that was retrieved by carrying out a majority vote between the label that was assigned by the person who created the hypothesis and the labels that were assigned independently to the sentence pair by four other people. Finally, all the sentence pairs were translated into the different languages by translators. In addition, Conneau et al. [2018] carry out some tests to verify that the original gold label still holds in the translated sentences: They recruited two bilingual annotators to reevaluate 100 examples in English and French, i.e. they had to re-assign the labels given the sentence pairs. For the English examples, Conneau et al. find a 85% consensus on the gold labels, and for French a corresponding 83%, from which the authors conclude that the overall semantic relationship between the two languages has been preserved.

Here are some randomly sampled German XNLI hypothesis-premis sentence pairs with the corresponding gold labels.

(3.34) Ich wusste nicht was ich vorhatte oder so, ich musste mich an einen bestimmten Ort in Washington melden.

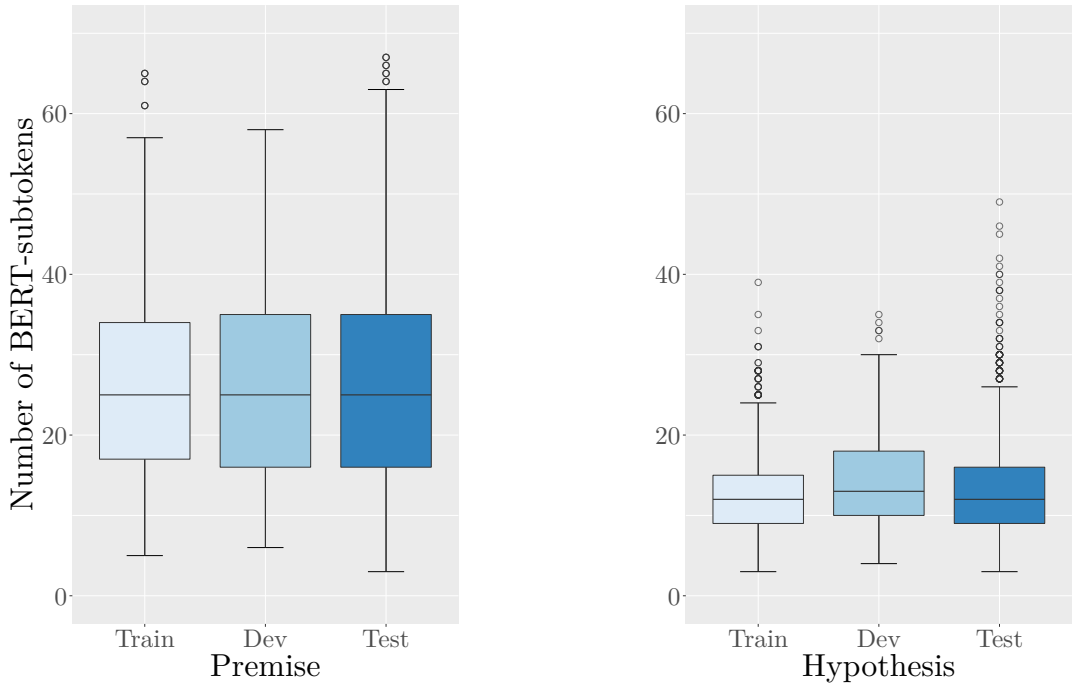

Figure 7: Left: Length of subtokenized XNLI premises. **Right**: Length of subtokenized XNLI hypotheses. While the lengths between the sets do not vary greatly, the hypotheses are significantly shorter than the premises.

Ich war noch nie in Washington, deshalb habe ich mich auf der Suche nach dem Ort verirrt, als ich dahin entsandt wurde.

Neutral

(3.35) Natürlich haben sie mich dort gefragt, warum ich ging.

Sie fragten, warum ich in den Laden ging.

Neutral

(3.36) Und ich dachte OK und das war es dann!

Nachdem ich ja gesagt hatte, endete es.

Entailment

(3.37) John Burke (Alabama) überprüft und analysiert andere zeitgenössische Konten und findet, dass Boswells nicht nur der genaueste ist, sondern er nutzt es, um Johnsons Charakter zu demonstrieren, wobei andere es lediglich als literarischen Geschwätz abstempeln.

John Burke ignoriert Aussagen.

Contradiction

(3.38) Die öffentliche Bibliothek in Greenlee County, Arizona, zeigt die finanziellen und technologischen Probleme von ländlichen Einrichtungen auf.

Greenlee County hat eine öffentliche Bibliothek.

Entailment

As for the datasets before, here are some examples to test for yourself with the answers in this footnote[15].

(3.39) Er kommt aus Griechenland
und er kommt aus einem
kleinen Dorf in Griechenland namens
Tokaleka und er kam nach Amerika
und ich glaube es war 1969 oder
1970 und er heiratete kurz darauf.

Er ist ein griechischer
Mann, der kein Englisch spricht.

(3.40) Suchen Sie nach Emily Dickinson's
kommendem Gedicht, alles
was ich wirklich von Poesie wissen
muss habe ich bein Microsoft gelernt.

Dickinson schrieb Gedichte.

(3.41) „So hat es auch in den kubanischen
Tropen vor Kurzem einen
Tag gegeben, der so schön wie Ruhm
und kalt wie ein Grabstein war."

Es ist immer über 80 in Kuba.

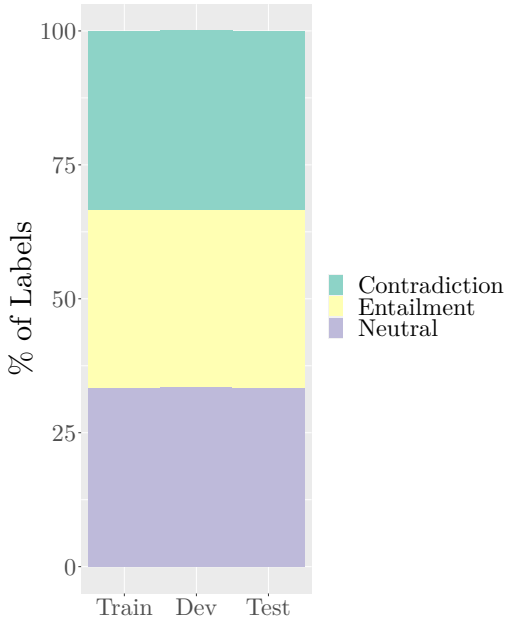

Figure 8: Label distributions of the XNLI data set; apparently, the three classes are very well balanced.

XNLI is a medium sized dataset in the context of GerGLUE: It comprises a total of 7,498 examples, predefined in a development and test set of 2,489 and 5,009 examples, respectively. From the development set, there are a further 2,115 examples split off as training set, leading to a very small dev set of only 374 examples. As will be seen afterwards, this is of course not an ideal set up — therefore I resample the

[15] 3.39: Neutral, 3.40 Entailment, 3.41 Neutral

complete XNLI data set, split it into normal sets (5,248, 1,125, and 1,125 examples, respectively), compute a second round of experiments on this splits and obtain indeed better results.

The lenghts of premises and hypothesis reported in figure 7 confirm the impression from the sample sentences that hypothesis are significantly shorter than premises; this is not such a surprise considering the construction of the dataset where people were asked to produce hypotheses given a certain premise — if given such a task one would come up naturally with a comparatively short, concise question.

The best system Conneau et al. [2018] report for German on their XNLI data set is a model that relies heavily on translation: They train a BiLSTM on the MultiNLI data (432,702 instances) and translate the test set of the given language to English and predict on this data. Employing this strategy, the authors obtain an accuracy on the German test set of 68.7%.

3.2.6 XQuAD

Like MLQA, XQuAD [Artetxe et al., 2019] is a question answering dataset, where an answer span needs to be identified given a context and a corresponding question. In contrast to MLQA, where the contexts are text sequences from Wikipedia articles, XQuAD was constructed by humanly translate 240 paragraphs and 1190 question-answer pairs from SQuAD 1.1 [Rajpurkar et al., 2016].

(3.42) **Context**: Aristoteles lieferte eine philosophische Diskussion über das Konzept einer Kraft als integraler Bestandteil der aristotelischen Kosmologie. Nach Ansicht von Aristoteles enthält die irdische Sphäre vier Elemente, die an verschiedenen „natürlichen Orten“ darin zur Ruhe kommen. Aristoteles glaubte, dass bewegungslose Objekte auf der Erde, die hauptsächlich aus den Elementen Erde und Wasser bestehen, an ihrem natürlichen Ort auf dem Boden liegen und dass sie so bleiben würden, wenn man sie in Ruhe lässt. Er unterschied zwischen der angeborenen Tendenz von Objekten, ihren „natürlichen Ort“ zu finden (z. B. dass schwere Körper fallen), was eine „natürliche Bewegung“ darstellt und unnatürlichen oder erzwungenen Bewegungen, die den fortwährenden Einsatz einer Kraft erfordern. Diese Theorie, die auf der alltäglichen Erfahrung basiert, wie sich Objekte bewegen, wie z. B. die ständige Anwendung einer Kraft, die erforderlich ist, um einen Wagen in Bewegung zu halten, hatte konzeptionelle Schwierigkeiten, das Verhalten von Projektilen, wie beispielsweise den Flug von Pfeilen, zu erklären. Der Ort, an dem der Bogenschütze den Pfeil bewegt, liegt am

Anfang des Fluges und während der Pfeil durch die Luft gleitet, wirkt keine erkennbare effiziente Ursache darauf ein. Aristoteles war sich dieses Problems bewusst und vermutete, dass die durch den Flugweg des Projektils verdrängte Luft das Projektil zu seinem Ziel trägt. Diese Erklärung erfordert ein Kontinuum wie Luft zur Veränderung des Ortes im Allgemeinen.

Question: Wer leitete eine philosophische Diskussion über Kraft?
Answer: Aristoteles

Question: Wovon war das Konzept der Kraft ein integraler Bestandteil?
Answer: aristotelischen Kosmologie

Question: Aus wie vielen Elementen besteht die irdische Sphäre nach Ansicht des Aristoteles?
Answer: vier

Question: Wo vermutete Aristoteles den natürlichen Ort für Erd- und Wasserelemente?
Answer: auf dem Boden

Question: Was bezeichnete Aristoteles als erzwungene Bewegung?
Answer: unnatürlichen

XQuAD is also one of the smaller datasets in the GerGLUE corpus — a total of 1,179 examples, without predefined splits, which results in training, development, and test splits of 820, 181, and 178 examples each. As for MLQA, the answer spans often consists of only a handful of subtokens, due to the fact of many questions asking for only a certain name or place mentioned in the context.

Again,you can test you own performance on this dataset by answering the following questions relating to the given context:

(3.43) **Context**: Da sowohl Präsident Kenyatta als auch Vizepräsident William Ruto 2013 Gerichtstermine vor dem Internationalen Strafgerichtshof in Verbindung mit den Auswirkungen der Wahl von 2007 hatten, entschied sich US-Präsident Barack Obama, das Land während seiner Afrikareise Mitte 2013 nicht zu besuchen. Später in diesem Sommer besuchte Kenyatta auf Einladung von Präsident Xi Jinping China, nachdem er einen Zwischenstopp in Russland eingelegt und die USA als Präsident nicht besucht hatte. Im Juli 2015 besuchte Obama Kenia als erster amerikanischer Präsident, der das Land während seiner Regierungszeit bereiste.

Question: Welches Land besuchte Kenyatta auf Einladung des

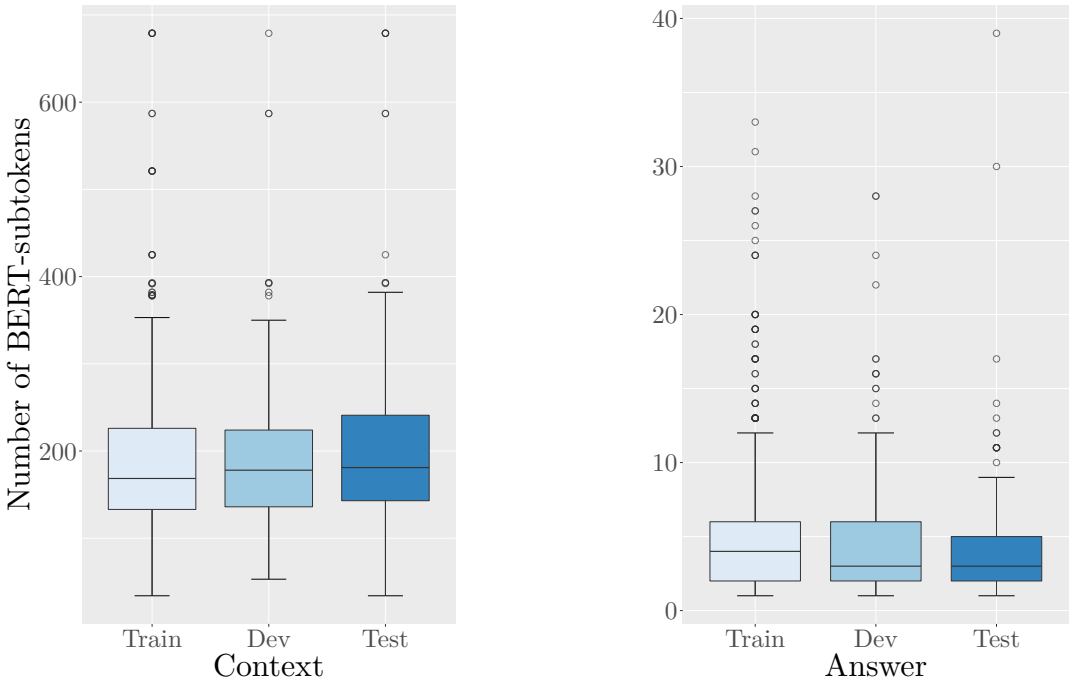

Figure 9: Left: Length of subtokenized XQuAD contexts. **Right**: Length of subtokenized XQuAD answers. Note the difference in y-axis scaling between the two plots.

Präsidenten?[16]

Question: Wann besuchte Obama Kenia schließlich?[17]

Question: Wer entschied sich, das Land 2013 nicht zu besuchen?[18]

Question: Was war das Ergebnis der Wahl von 2007?[19]

As benchamrk results on XQuAD, Artetxe et al. employ a sophisticated technique that consists in re-training a monolingual English BERT model on Wikipedia and transfer it to the target languages, following these steps:

1. Pre-train a monolingual BERT in English with original pretraining objectives
2. Transfer model to new language L_2, but learn only token embeddings new (transformer body is frozen) with original pretraining objectives
3. Fine-tune transformer for downstream task in English (topken embeddings are

[16] China

[17] Juli 2015

[18] US-Präsident Barack Obama

[19] Gerichtstermine vor dem Internationalen Strafgerichtshof

frozen)

4. Zero-shot transfer this model to L_2 by swapping the English token embeddings with the L_2 embeddings

The authors report the following results for the German part of XQuAD: F1: 73.6 Accuracy (exact match): 57.6%

3.2.7 Summary

In the following table 4 an overview of all the datasets is given; the NLP and corresponding ML tasks the dataset is designed for, the numbers of examples of the splits and nature of the splits (predefined or not), the style or register of the text present in the dataset, as well as some important distinctiveness of each dataset. Note that the ordering is not strictly alphabetical, but follows the following grouping: First, come the single sentence classification tasks, then follow the sentence pair classification task, and lastly come the question answering tasks. This order — classification tasks together, question answering tasks together — will due conceptual reasons also be adhered to in the rest of the thesis.

GerGLUE

		NLP Task	ML Task	# Examples Train/Dev/Test	Predefined Splits	Register coll. < mix. < form.	Remarks
		Classification					
Single	deISEAR	Emotion Detection	Multi-Class	700/150/151	-	mixed	Boilerplate text structures ("Ich fühlte [?], als ...")
Single	SCARE	Sentiment Analysis	Multi-Class	1,232/264/264	-	colloquial	Very informal, ungrammatical, and often short text snippets
Pair	PAWS-X	Paraphrase Identification	Binary	48,977/1,932/1,967	Train/Dev/Test	formal	Translation artifact noise, dev/test splits OOD to train
Pair	XNLI	Natural Language Inference	Multi-Class	2,115/374/5,009	Dev/Test	mixed	Translation artifact noise, language from different domains
		Question Answering					
Pair	MLQA	Question Answering	Span Prediction	432/77/4,499	Dev/Test	formal	Highly imbalanced splits regarding number of examples
Pair	XQuAD	Question Answering	Span Prediction	820/181/178	-	formal	Small data set

Table 4: Overview of *GerGLUE* data sets and tasks. The number of examples represent the size of the set splits after preparing the data sets for the experiment; therefore all datasets have all splits. Note that during the preprocessing some examples had to be excluded (see chapter 3 for more details), that's why some numbers do not coincide with those of table 2.

4 Architecture

In this chapter, I describe the different parts of the GliBERT model, how they interact and what a priori decisions were taken during the design of its architecture (e.g. the number of predicate-argument structures considered, etc.).

To eliminate possible misunderstandings and establish a "standard" vocabulary discussing the GliBERT architecture, I define the following terms which will be used in the specified sense throughout this thesis:

- Following Devlin et al. [2018], in this thesis "a 'sentence' can be an arbitrary span of contiguous text, rather than an actual linguistic sentence."
- A "model" or "system" denotes any algorithm that is tailored at a specific task and designed to handle natural language as input.
- As mentioned in the introduction, natural language can be communicated through different channels (speech, singing, text) — I am, however, only concerned with textual representations of language.
- Following Goldberg [2017], I use uppercase letters W, X, Z to represent matrices and lowercase letters b, x, y to represent vectors. Moreover, keeping consistent with Goldberg, vectors are assumed to be row vectors, leading to right-side matrix multiplication: $xW + b$.
- I do assume the reader to be fond of the basic mathematical and theoretical concepts of machine learning, keeping mathematical demonstration to a minimum. For a sound introduction into concepts, terms, and mathematical reasoning behind deep learning in NLP, I can highly recommend "Neural Network Methods for Natural Language Processing" [Goldberg, 2017].

4.1 Overview

GliBERT is an architecture that combines different, pre-existing models and tools to solve the classification or question answering task at hand. The general way an

input sequence is processed by GliBERT is depicted in figure 10:

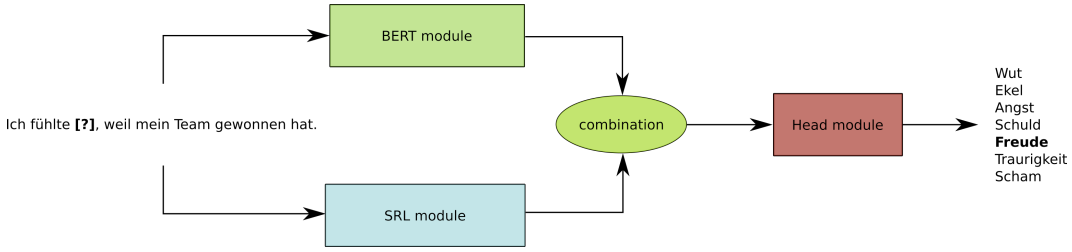

Figure 10: General architecture of GliBERT, exemplified for the deISEAR task.

An input sentence gets processed in parallel by two systems, the vanilla German BERT and the SRL producing and encoding module, to produce two numerical representations of its tokens. These representations get combined and are sent through a head module which produces the actual predictions. The core parts of the model are the following:

BERT module This is the vanilla BERT base model: It tokenizes the input sequence, sends it through its twelve transformer layers and outputs the final hidden states of each (sub-)token.

SRL module This modules actually consists of three submodules: First, the sequence is processed by the ParZu [Sennrich et al., 2009] parser to identify predicates. Secondly, the sequence along the information about which tokens are predicates is handed to the DAMESRL model [Do et al., 2018] which predicts actual SRLs. To ensure there are no tokenization mix-ups between BERT and DAMESRL (because these differences are not reversible as will be seen later), the sequence gets tokenized BERT-style and is passed as this list of tokens to DAMESRL. In a last step, the SRL squence gets numerically encoded, using a bi-directional two-layer GRU.

combination In this step the BERT and SRL representation get combined: to do this, the embeddings need to be processed, i.e. splittet or merged, respectively, so that they can be concatenated. For this, there exist two approaches: (A) Fuse the subtokens of BERT back to tokens, or (B) split the SRLs according to the subtokens of BERT.

Head module At last, the combined token representations of the input is fed through the final network that transforms it to predict task-dependent output. Several architectures can potentially be applied here: FFNNs, GRUs, CNNs, etc.

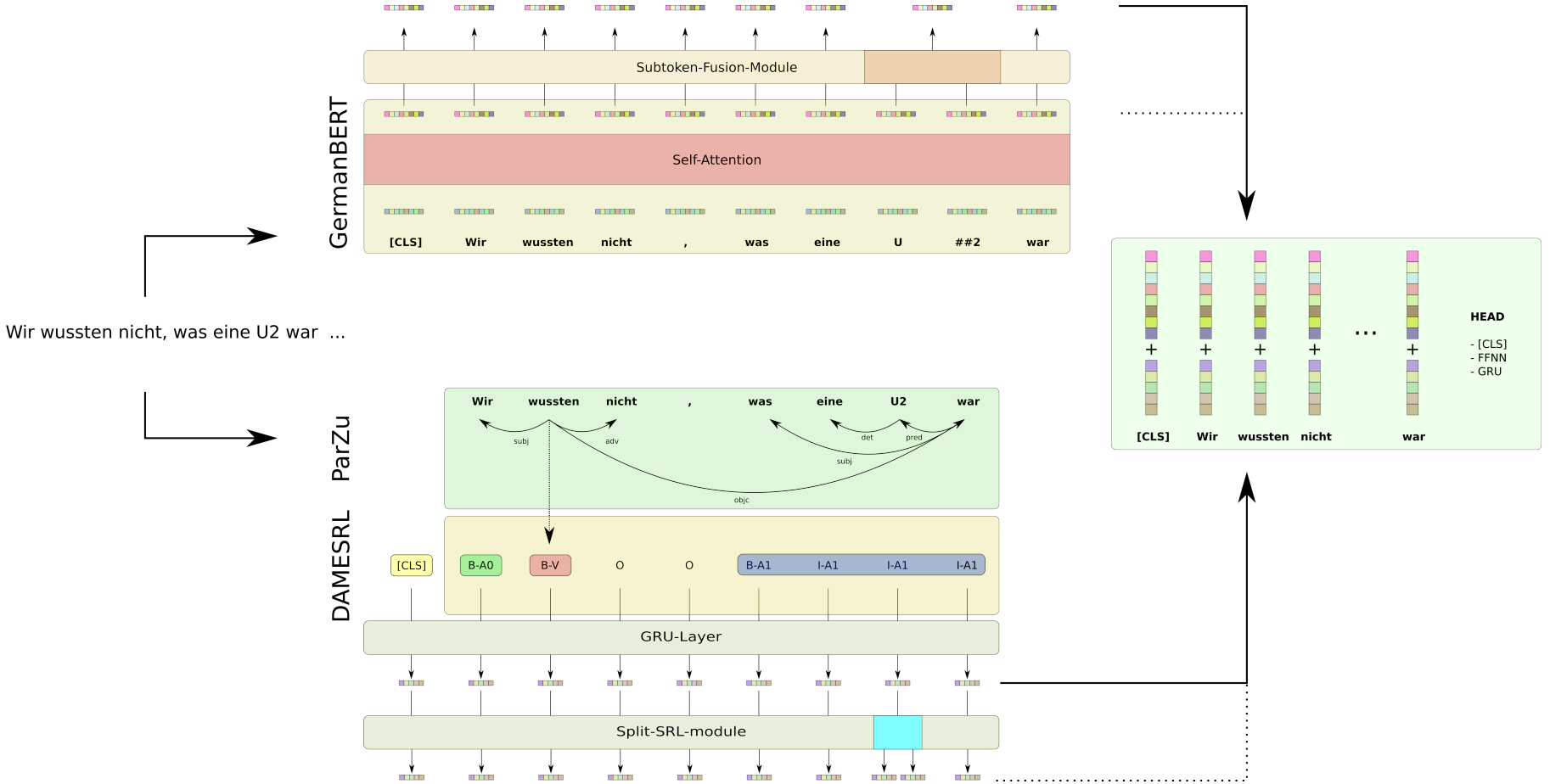

Figure 11: Detailed architecture of GliBERT: On the left, the input sentence is passed through two paths: On top, through the German BERT, with the optional subtoken fusion module on top. On the bottom, through ParZu and DAMESRL, with subsequent embedding via a GRU model; after that an optional split token module follows. The bold arrows on the right side show information flow, if BERT subtokens are fusioned for appending with SRLs. The dotted arrows represent the information flow if the SRLs are split to match with the BERT subtokens.

4.2 BERT Module

Since its publishing three years ago, BERT [Devlin et al., 2018] has often been viewed as a turning-point in NLP: The embeddings it computes by implementing massive self-supervised pre-training proved to be very potent representations of language and were successfully implemented in a wide array of applications addressing downstream tasks via transfer learning (see chapter 2). Pretrained models and APIs for BERT are by now vastly available for a multitude of languages — I chose to use the `bert-base-german-cased` model from deepset which is available in `pyTorch` through the hugging face's `transformer` library Wolf et al. [2019].

While the original BERT was presented in two different sized variants — *base* and *large* — deepset only provides a BERT base model which has the following specifications according to it's configuration file:

Transformer Blocks	12
hidden Size	768
hidden activation function	GeLu
hidden dropout probability	0.1
Attention Heads	12
Vocabulary size	30,000 (cased)
Total Parameters	110 million

Table 5: German BERT hyperparameter configuration.

The handling of the BERT model is straightforward through huggingface's `transformer` library: With a simple function call `BertModel.from_pretrained()` one loads the pretrained BERT, and with another function, `BertTokenizer.from_pretrained()`, one instantiates the BERT tokenizer. After encoding a sentence using the tokenizer's method `.encode_plus()`, the encoded sentence is sent through BERT via its `.forward()`-method — or called implicitly, by passing the sentence to the model — which returns the vectors for all input tokens, which can then be used in downstream tasks. Fine-tuning is carried out by passing the computed loss to the specified optimizer funtion (I use the AdamW optimizer [Loshchilov and Hutter, 2019], a modification of the well-known Adam (Adaptive Moment Estimation) optimizing function [Kingma and Ba, 2014], implementing additional weight decay), which updates BERT's weight matrices.[1]

[1] After wiring all the different parts together, GliBERT is one big model having one loss function, which means that all weights of all layers in the system get update by the AdamW optimizer, not just BERT.

In the GitHub repository, in the file `load_data.py`, the data gets tokenized and loaded, and in the file `gli_bert.py`, the forward pass and weight-updating is defined in the `fine_tune_BERT()` function.

Code

In the configuration file `config.json` you are able to control if the vanilla BERT embeddings should be combined with SRL information via the `combine_SRLs` attribute: If it is set to `true`, the SRL module is activated and the BERT+SRL combined representations are used for the task, if it is set to `false`, only the vanilla BERT embeddings are used.

4.3 SRL Module

A Semantic Role Labeller (SRL) is a system, thaltl assigns automatically Semantic Roles to a given input text.[2]

State-of-the-art SRLs are end-to-end models, in the sense that there is no need of complex pre-processing of the input sentence, such as POS-tagging, syntax parsing, etc. in advance before the actual labeling takes place. However, as is the case for the model I employ, some pre-processing remains. For GliBERT, I implement the DAMESRL, a model presented by Do et al. [2018]. I use their pre-trained German Character-Attention configuration which, according to the authors, achieved an F1 score of 73.5% on the CoNLL'09 task [Hajič et al., 2009]. Despite being characterized as an end-to-end model by the authors, their SRL needs as input not only the tokenized sentence, but also "its predicate w_p as input" [Do et al., 2018]. In other words, DAMESRL expects as input a sentence s as list of tokens $[t_1, t_2, \ldots t_{n-1}, t_n]$, where for each token there is an attribute defined whether it functions as predicate in s or not.

For extracting predicates, I rely on the dependency parse tree the ParZu parser Sennrich et al. [2013] generates for a German sentence. Given the parsed sentence, I need to decide which tokens in it are predicates, and which are not. While this may seem like a straightforward task — just find the verb as in a simple sentence like "He *ate* the apple." —, there are actually a few caveats that need to be considered, e.g. (predicates are emphasized): (1) There may be no predicates at all: "What a day!". (2) There might be more than one predicate: "We *saw* her *leave* the room". (3) Not all verbs might be predicates, e.g. because they play grammatically the role

[2]This may be one or multiple sentences.

of a light verb: "I can *hear* you". In the following section, I will describe how I tackle these problems by making use of the parse tree information of ParZu.

4.3.1 ParZu

Before analyzing which semantic roles are present in a given sentence, one must determine the predicates in this sentence: "First, the predicates which assign semantic roles to the constituents are identified prior to semantic role labeling proper. They are usually identified as the main verbs which head clauses" [Samardzic, 2013, p. 74]. In a dependency framework like the Universal Stanford Dependencies (USD) [De Marneffe et al., 2014], which explicitly sets the content verb as root,[3] identification of the relevant predicate is straightforward: One needs just to look at the dependency parse tree of a given sentence and select the verbal heads — i.e. roots — of the clauses. However, the ParZu parser models not content verbs as heads but function verbs.[4] In other words, in the sentence "He was hit by a ball.", "hit" would be the predicate that assigns the semantic roles of proto-patient "he" and theme, or instrument "a ball". The dependency parse tree produced by implementing the USD framework would analyze the word "hit" as being the root of this sentence, making it easy to forward the such annotated sentence to a semantic role labeler which accepts as input a list of tokens, and marked which ones are the predicates.

Since the parse tree of the German equivalent "Er wurde von einem Ball getroffen" produced by ParZu analyzes the word "wurde" as root of the tree, it does not make sense to forward this as predicate to DAMESRL.

I propose the following algorithm 1 deciding whether a verb in a ParZu-parsed sentence is or is not a predicate using a heuristic, relying on the token's POS tag that the parser predicts. The ParZu parser's default output follows the CoNLL scheme [Buchholz and Marsi, 2006] which means that there are two levels of POS tagging: coarse-grained (CPOSTAG) and fine-grained (POSTAG), where the POSTAG corresponds to the token's STTS tag [Schiller et al., 1999].

Letting this algorithm run on the dependence parse tree depicted in figure 12 of the

[3]Note that is not undisputed:"The parsing scheme that USD advocates takes the division between function word and content word as its guiding principle. One major difficulty with doing this is that the dividing line between function word and content word is often not clear" Groß and Osborne [2015].

[4]This follows general dependency frameworks proposed for German, e.g. Gerdes and Kahane [2001]; Groß and Osborne [2015].

Algorithm 1 Predicate finding algorithm

```
1: for all token t ∈ sentence do
2:   if CPOSTAG t ≠ 'V' then
3:     t ← NOT_PRED
4:   else
5:     if POSTAG t = 'VVFIN' then
6:       t ← PRED
7:     else
8:       FLAG ← True
9:       for all token u ≠ t ∈ subclause where t ∈ subclause do
10:        if CPOSTAG u = 'V' ∧ u dependent on t then
11:          t ← NOT_PRED
12:          FLAG ← False
13:          break
14:        end if
15:      end for
16:      if FLAG = True then
17:        t ← PRED
18:      end if
19:    end if
20:  end if
21: end for
```

sentence

(4.1) Die Klage wurde abgewiesen, was als Sieg beschrieben werden kann.

leads to the correct identification of “abgewiesen” and “beschrieben” as predicates, disregarding the light and modal verbs “wurde”, “werden”, and “kann’.

Basically, the algorithm only takes tokens into account which are verbals (have the CPOSTAG *V*) ; if the POSTAG is *VVFIN*, i.e. if it is a finitely inflected verb form, it is right away considered to be a predicate. Otherwise, if it is a verbal form but not finitely inflected, it is checked whether it is dependent on another verbal element of its subclause, if this is the case, it’s not labeled as predicate, otherwise it is. This leads e.g. to the correct selection of “beschrieben” as predicate in the subclause of sentence 4.1, since it forms the “lowest” verbal element in this clause, only pointing to a light verb which modifies its grammatical function, which in turn points to another light verb which is the head of the subclause.

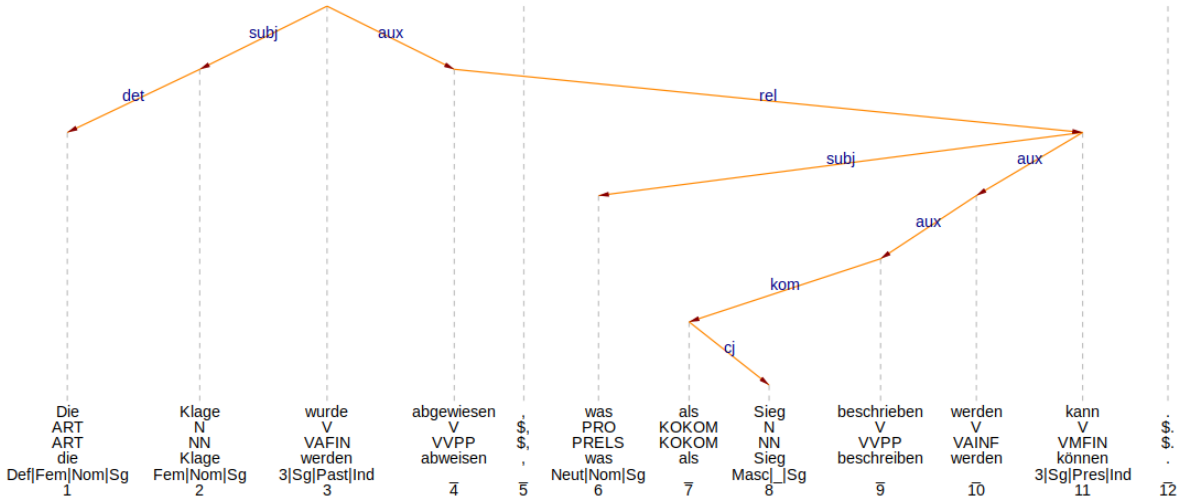

Figure 12: Example ParZu dependency parse tree for a sentence with two predicates out of several verbal forms. Different Information per word is displayed: The normal word form, the CPOSTAG, the POSTAG, the lemma of the word, morphological information, and the index.

4.3.2 Ensuring Tokenization Equivalence

Another difficulty I faced was the tokenization differences between different parsers; which can lead to sitautions where it is actually impossible to correctly automatically align the tokens which two parsers produce for the same sequence. For GliBERT, this problem occurs at the interface where the embeddings of the BERT module and the embeddings for the SRLs, which are based on the ParZu tokenization, need to be aligned. As shown in figure 10, a sentence passes through two pipelines which both apply tokenization to it: The ParZu/DAMESRL/GRU and the BERT pipeline. After both have computed numerical representation for the tokens, these must be combined. However, the tokenization of ParZu (which would get passed on to DAMESRL and then to the SRL-encoder) and the tokenization of BERT differ. ParZu implements the Moses tokenizer, while BERT, in contrast, utilizes an approach called “WordPieces”, which is a rather information processing motivated approach, rather then the linguistically motivated Moses: “Using wordpieces gives a good balance between the flexibility of single characters and the efficiency of full words for decoding, and also sidesteps the need for special treatment of unknown words.” [Wu et al., 2016, p. 2]. As a consequence, aligning the corresponding tokens to guarantee that the correct information pieces get combined is necessary.

To illustrate this, consider the following made-up but not unplausible text snippet:

(4.2) Anstiege um 4° zwischen 1990-2010

The tokenizations produced by ParZu, i.e. Moses, and BERT are depicted side-by-side (note that the BERT tokenization has been merged for better readability):

normal		**aligend**	
`BERT`	`Moses`	`BERT`	`Moses`
Anstiege	Anstiege	Anstiege	Anstiege
um	um	um	um
[UNK]	4	[UNK]	4
zwischen	°	[UNK]	°
1990	zwischen	zwischen	zwischen
-	1990-2021	1990	1990-2021
2021		-	1990-2021
		2021	1990-2021

Table 6: Tokenization aligning difficulties displayed on example sentence 4.2. Note that the BERT subtokens are already merged back to tokens for better readability ("Anstieg", "##e" = "Anstiege"). In this case both tokenization sequences would need duplication (duplicated tokens are highlighted in blue). Achieving this reliably for all potential edge cases is nearly impossible.

In the beginning, I attempted to device an algorithm which would duplicate the respective tokens in the Moses tokenization to lign up with the BERT (sub-)tokenization. However, it soon became clear that this was an endeavor too error prone and meticulous: It is e.g. not a priori clear which tokenization is the shorter one and therefore needs to be duplicated (sometimes, as in table 6, both sequences need duplication); BERT tokenizes tokens which are out-of-vocabulary (OOV) or which contain OOV subtokens as [UNK] further excessively complicates the picture etc. Therefore, I chose the following strategy which avoids all these problems:

(1) Take the ParZu-tokenized and parsed sentence, apply the predicate-finding algorithm to it. Then, (2) tokenize the sentence using the BERT tokenizer, merge subtokenized tokens back to regular tokens, identify the afore detected predicates[5] and hand the BERT tokenized, predicate marked token sequence to DAMESRL.

[5]Since predicates are normally verbal forms without any special characters in them, a string comparison search suffices to achieve this.

4.3.3 DAMESRL

There are only a few end-to-end SOTA SRL frameworks available for German that come with a pre-trained model, especially such ones that can be conveniently integrated in a pipeline of a bigger system.

Do et al. [2018] fill exactly this hole: They introduce DAMESRL, an SRL framework that implements SOTA architecture, namely self-attention mechanisms, similar to BERT's. They report an F1 score of 73.5 for their best model configuration on the German data set of CoNLL '09. This best configuration is based on word embeddings as well as character embeddings, self-attention and a softmax layer on top.

The DAMSRL predictor receives the BERT-tokenized sentence along with the information which tokens in it are predicates (zero or more). For each token labelled as predicate in a sequence it predicts for each other token in the sequence its SRL.

In most cases, a sentences contains not exactly one predicate which distributes semantic roles, but several, especially in longer sentences, — or even none, especially in colloquial, short sentences. Research by Zhang et al. suggests that fixing the number of predicate-argument structures to three yields the best results; so I adopt this number. In other words, if a sentence has more than three argument-predicate structures, I only care about the first three predicates identified (if proceeding from left to right through the sequence), and disregard the others.

	Slot 1	Slot 2	Slot 3		Slot 1	Slot 2	Slot 3
Wir	B-A0	O	O	Wir	B-A0	B-A0	B-A0
wollten	O	O	O	wollten	O	O	O
eine	B-A1	O	O	eine	B-A1	B-A1	B-A1
Sache	I-A1	O	O	Sache	I-A1	I-A1	I-A1
mehr	I-A1	O	O	mehr	I-A1	I-A1	I-A1
retten	B-V	O	O	retten	B-V	B-V	B-V
als	B-C-A1	O	O	als	B-C-A1	B-C-A1	B-C-A1
die	I-C-A1	O	O	die	I-C-A1	I-C-A1	I-C-A1
Restlichen	I-C-A1	O	O	Restlichen	I-C-A1	I-C-A1	I-C-A1
.	O	O	O	.	O	O	O

SRL 4.1: The two strategies for dealing with less than three predicates: **Left**: The open SRL slots get filled with the special SRL *0*. **Right**: The first SRL structure gets duplicated until all slots are filled.

However, if there are fewer than three predicate-argument structures present, I test and report results for two strategies: The first solution (the left sentence in SRL 4.1) lies in filling the "unfilled" predicate-slots with the special "0"-SRL. The second (the right sentence in SRL 4.1) simply copies the first predicate-argument structure to the unfilled slots, thus amplifying the signal from the first predicate-argument structure.

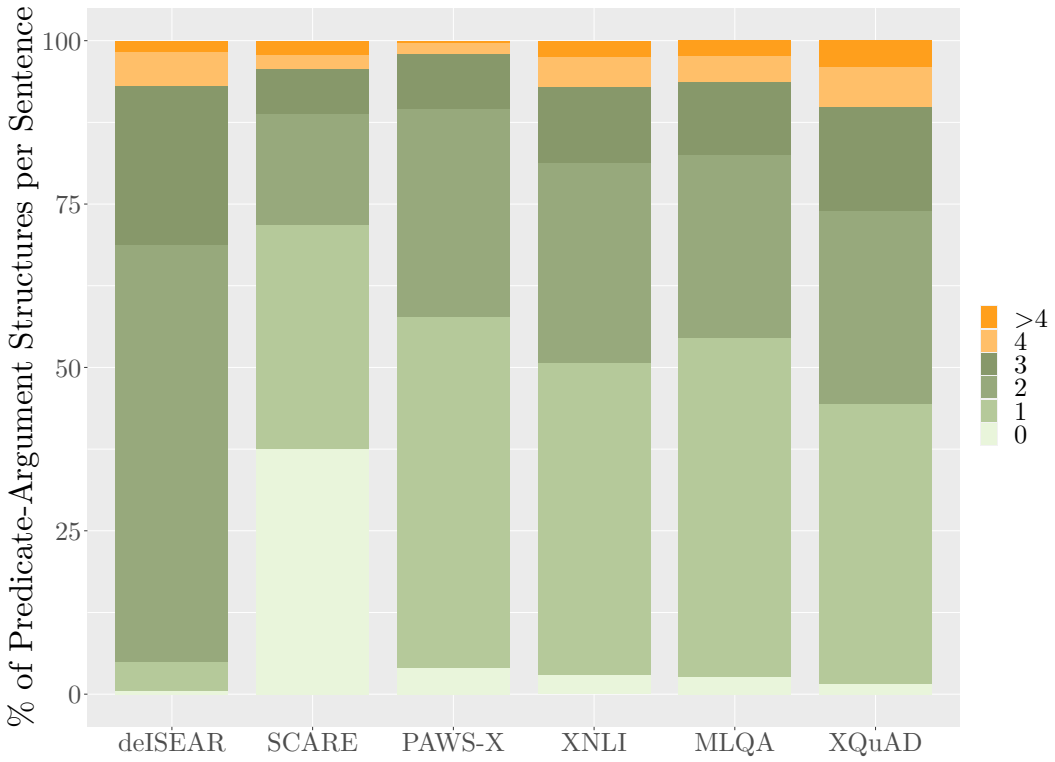

Figure 13: Number of predicate-argument structures in all data sets. Due to it's boilerplate template form, deISEAR shows a peculiar distribution: Since it's examples always begin with "Ich [PRED fühlte] X, als ....", it's guaranteed that at least one predicate is identifed, and it is very probable, because of this sentence structure, that another will occur. The other curious pattern exhibits SCARE: In no other data set the amount of sentences where ParZu couldn't detect any predicates is that high.
Besides the noted peculiarities, it is safe to say that setting the maximum number of predicate-argument structures to three does probably not lead to much information loss; on average over all datasets, 92.73% of all sentences possess three or less such structures.

Code

The predicate-filling mode can be controlled via the `zeros` attribute in the `config.json` file: If it is set to `true`, then the empty predicate-argument slots are filled up with `0`-SRLs, if it is set to `false`, the first predicate-argument structure gets copied to the empty slots.

4.3.4 GRU

Finally, the predicted SRLs need to be encoded in a numeric way so they can be concatenated to the vectors which are computed by BERT. The "classic", pre-

transformer age, way of encoding sequential data would be to employ a recurrent neural network architecture. Typically, one would implement an architecture that counteracts the well-known vanishing / exploding gradients problem of vanilla RNNs (cf. [Bengio et al., 1994]), such as LSTMs [Hochreiter and Schmidhuber, 1997] or GRUs [Cho et al., 2014]. I decided to use GRUs, since they are less computational intensive and research has found both architectures for many tasks and datasets performing on par (cf. Chung et al. [2014]).

For this, the three SRLs for each token get transformed into their numerical representation via look-up in the SRL dictionary of the GRU model, such that each SRL token is a vector $v \in \mathbb{R}^{20}$. Then, the numerical representations for each token get concatenated and those 60-dimensial vectors form then the input for the bi-directional two-layer GRU which computes hidden states for all the inputs.

Because of the subsequent goal of combining BERT embeddings with SRL embeddings, the following considerations need to be taken into account.

BERT adds several special meta tokens to sequences it embeds: (1) At the start of each seqeuence, be it a single sentence or sentence pair one, it inserts a special [CLS] token; CLS standing for classification. (2) At the the end of a sequence and between sentence pairs, BERT inserts a separation token [SEP], which signals the end of a sequence. (3) If the sequence is shorter than the defined maximum length, it gets filled up with [PAD] tokens. An XNLI sentence pair with maximum length 25 woudl thus be represented as tokenizer BERT sequence the following way (for better readability, the BERT subtokens are marked blue):

(4.3) Du musst dort nicht bleiben.
An genau der Stelle musst du stehenbleiben!

[CLS] Du muss ##t dort nicht bleiben . [SEP] An genau der Stelle muss ##t du stehen ##bleiben ! [SEP] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD]

One of the goals of my experiments is to compare the standard BERT implementation with the SRL enriched GLiBERT variant. Since the vanilla classification head predicts only using the last hidden state of the [CLS] token, I need to represent SRL information also on this special token. Therefore, I follow Zhang et al. [2019b] and add a meta-SRL [CLS], which represents the meta semantic role of the [CLS] special token.

Another question that arises concerncs the SRL encoding of multiple sentences: Suppose we have a sequence consisting of several sentences (separated by brackets

and indexed by A, B, and C):

(4.4) [$_A$ Die Ereignisse von Oni finden im oder nach dem Jahr 2032 statt und beschreiben ein dystopisches Zukunftsbild der Erde.] [$_B$ Die Welt ist so verschmutzt, dass nur noch kleine Teile bewohnbar sind.] [$_C$ Um die internationale Wirtschaftskrise zu lösen, haben sich alle Völker unter einer Weltregierung vereinigt.]

Should each sentence A, B, and C be encoded separately and these represenatations than be combined with the aligned BERT subtokens? Or is it more effective to "glue" all sentences together (or rather, the concatenated SRL tokens in those sentences) and embedd this sequence? Further, also meta SRLs for [SEP] and [PAD] tokens could be added — so that also sentence pair tasks are embedded as one long sequence of SRL tokens. All experiments except for XQuAD α were conducted implementing the latter approach, proving that the model profits more from the second architecture.

Code

The hyperparameters for the GRU that embeds the SRLs are specified in the configuration file `config.json`. The following hyperparameter configuration was used in the experiments of this thesis:

GRU_head_hidden_size: 712

GRU_head_dropout: 0.5

SRL_embedding_dim: 20

SRL_hidden_size: 32

SRL_num_layers: 2

SRL_bias: true

SRL_bidirectional: true

SRL_dropout: 0.3

4.4 Combination

The combing of a token's BERT embedding with its corresponding SRL embedding is done straightforwardly; both numerical vectors are concatenated:

Take, the first token t_1 of the sentence in 4.1, "Wir": After being consumed by

the BERT module, it has a BERT embedding vector representation $b \in \mathbb{R}^{768}$. Its SRL representation, let's say the zero-filled variant `B-A0+0+0`, has a GRU-embedded representation $s \in \mathbb{R}^{60}$. The final numerical representation w_1 of the token t_1 "Wir" plus its SRL `B-A0+0+0` is computed as:

$w \in \mathbb{R}^{828} = b \cdot s$

where $\cdot$ denotes the concatenation operation.

Because the tokenization of the SRL module is based on the merged BERT tokenization, there remain two ways of combining BERT embeddings and SRL embeddings, however: (1) The embeddings of subtokenized tokens in BERT are merged back to token level, or (2) SRL embeddings of tokens which were subtokenized in the BERT module get duplicated to align with the BERT embeddings. Consider the following word as an example: "Restlichen" has one SRL in 4.1, `I-C-A1+0+0` (in the duplicated variant), which would lead to one vector s as numerical representation, while it would be tokenized as "Rest", "##lichen" inside BERT, leading to two vectors b_1, b_2 as numerical representation. In the first scenario, b_1 and b_2 would be merged into one vector $b = \frac{b_1+b_2}{2}$; this vector[6] would then be concatenated with s to form the final representation w of "Restlichen". In the second variant, s would be copied once[7] to align with the BERT vectors, which would lead to the final representation of "Restlichen" as two vectors: $w_1 = b_1 \cdot s$ and $w_2 = b_2 \cdot s$.

For all experiments, I run both variants, reporting in chapter 5 the results, which indicate that variant 1, i.e. merging BERT subtokens back to tokens to align with the SRL, leads to information loss, and therefore weakens the general model performance.

Code

The BERT-SRL-combination mode can be controlled via the `merge_subtokens` attribute in the `config.json` file: If it is set to `true`, then the BERT embeddings of subtokens get merged into one token embedding, if it is set to `false`, the SRL embeddings are duplicated to match up with the BERT subtokens.

[6] Of course, this generalizes to all possible number of subtokens: The numerical representation b of a token t consisting of $(t_1, t_2, \dots t_n)$ subtokens corresponding to $(b_1, b_2, \dots b_n)$ vectors, would be computed as $b = \frac{1}{n}\sum_{i=1}^{n} b_i$.

[7] Similarly, the copying of s would be performed $n-1$ times for n BERT subtokens.

4.5 Head Module

Since the vanilla classification and question answering heads differ quite starkly — classification only takes into account the [CLS] token while the start and end index probabilities need to be computed on every token in the sequence — I cover them in separate subsections.

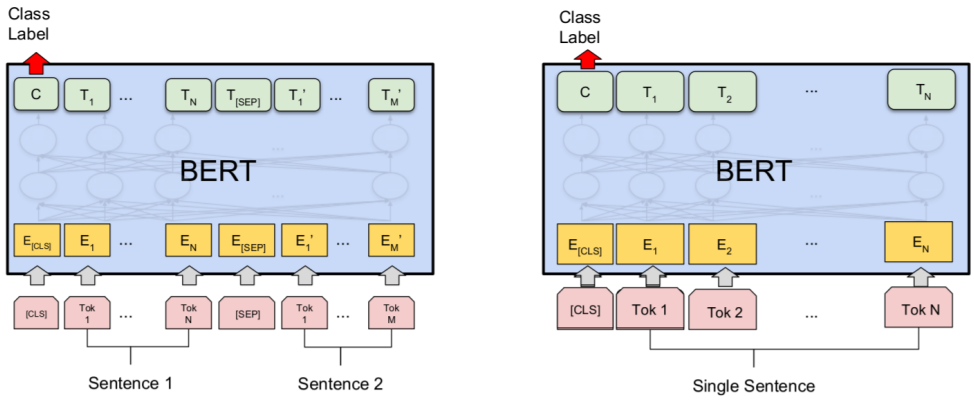

Figure 14: Schema for sentence pair (left) and single sentence (right) classification. Figure taken from [Devlin et al., 2018].

Additionally, for classification tasks, I found more ways in tackling these with different heads, while for the question answering task I only saw one possibility for a GliBERT head which is virtually the vanilla head, except that SRL information may be added to the BERT subtokens.

4.5.1 Classification

For both, single sentence and sentence pair tasks, the vanilla BERT head considers only the last hidden state of the [CLS] token, after the sequence was sent through the transformer blocks and predicts class probabilities only relying on the information present in this vector $w \in \mathbb{R}^{786}$:

In what follows, I present the three classification heads I used for my experiments: The [CLS] head, similar to the vanilla classification head, the FFNN head, which considers the numeric representation of all tokens in the sequence, and the GRU head, which implements a bidirectional recurrent neural network that computes a condensed representation of the whole sequence before predicting the actual class probabilities.[8]

[8]Early in the experimental phase, I also devised a Convolutional Neural Network (CNN) head,

4.5.1.1 [CLS] Head

The vanilla BERT classification approach, as proposed by Devlin et al. [2018], feeds "the `[CLS]` representation [...] into an output layer for classification, such as entailment or sentiment analysis."

In other words, to predict a label in a classification task like sentiment analyis, BERT encodes the whole sequence, i.e. adds its special [CLS], [SEP], and [PAD] tokens, and sends it through its twelve transformer blocks. After that, only the numeric representation of the [CLS] token is fed into a layer, wchich outputs the label probabilities.

The GliBERT [CLS] head operates the same way, except that it is also capable of handling additional SRL input: After adding corresponding SRLs for the special BERT tokens and embedding the SRL sequence, the last hidden state representation for the BERT+SRL combined [CLS] token is selected and softmax-scaled probabilities for all classes are computed. This is also the approach Zhang et al. [2019b] employ in their SemBERT architecture.

Formally, the [CLS] Head is a one-layer Fully connected Feedforward Neural Network (FFNN) without a non-linear activation function. A prediction $\hat{y}$ is the result of sending the embedding of the [CLS] token — or the the embedding of the [CLS]+SRL token, respectively —, i.e. the first token in the sequence, w_0 through the network layer W, normalizing the outputs by applying the $softmax$ function[9] to it, and selecting the dimensionality, i.e. class, with the highest probablity:

$$\hat{y} = max(softmax(Ww_0 + b))$$

where $W^{I \times J}$ is the weight matrix of the FFNN layer and b is a bias term learned during fine-tuning. The dimenionalities (I and J) of W vary according to the setting and dataset: while J is the number of classes for the task — this number ranges from 2 (PAWS-X, *true*, *false*) to 7 (deISEAR, *Angst, Wut, Ekel, Freude, Scham,*

however, the preliminary results were not promising, therefore I concentrated on the other three heads.

[9] The $softmax$ function takes as input a vector $x \in \mathbb{R}^n$ and normalizies it such that all dimensions are positive and add up to 1. In other words, it computes a probability distribution over the input vector's dimension. Mathematically, the $softmax$ function $\sigma : \mathbb{R}^n \to [0,1]^n$ is defined as:

$$\sigma(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

for $i = 1, \ldots, n$.

Schuld, Traurigkeit) —, I is either equal to 768 (no SRL information) or equal to 828 (plus SRL information).

In figure 15, the information flow through GliBERT's network and the final FFNN layer on the [CLS] token is visualized.

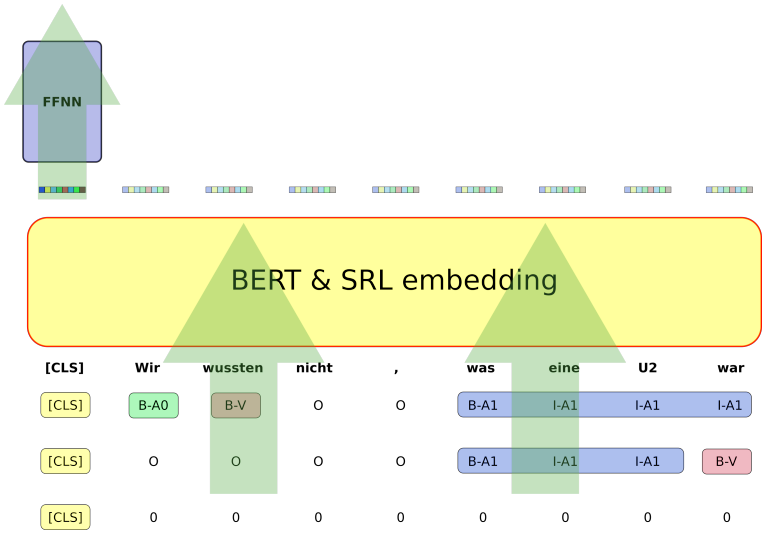

Figure 15: Head with a one-layer **F**eed **F**orward **N**eural **N**etwork on the [CLS]-token. This corresponds to the vanilla BERT classification head.

During training, the loss ℓ is computed implementing the negative log likelihood loss (NLLL)[10] batch-wise and backpropagating it through the network. The NLLLoss is the loss function applied to all heads and will therefore not be mentioned in the following head descriptions.

Intuitively, I expected this head to not show stark differences between the +SRL −SRL settings, since the SRL information, which is essentially a relational, sequential mark-up of the sequence is boiled down into one token. However, as is also confirmed by the results of Zhang et al. [2019b], the [CLS] head definitely profits from SRL information.

[10]Mathematically, the NLLL is defined as

$$\ell(x, y) = \sum_{n=1}^{N} \frac{1}{\sum_{n=1}^{N} w_{y_n}} l_n$$

with $l_n = -w_{y_n} x_{n,y_n}$ where x is the input, y is the target, w is the weight, and N is the batch size (cf. `https://pytorch.org/docs/stable/generated/torch.nn.NLLLoss.html`).

4.5.1.2 FFNN Head

While the [CLS] head computes class predictions based on the weights of the last layer output of the [CLS]-token, this head takes the last layer output of all tokens $w_0, w_1, \ldots, w_n$, concatenates them into one large vector $w = w_0 \cdot w_1 \cdot \ldots \cdot w_n$, and sends this vector w through a one-layer FFNN to predict a class label $\hat{y}$ (depicted in figure 16):

$$\hat{y} = max(softmax(Ww + b))$$

where $W^{I \times J}$ is the weight matrix of the FFNN layer and b is a bias term learned during fine-tuning. While J still is equal to the number of classes for the task at hand, I is the dimensionality of the input vector w: If the setting is such that no SRL information is added, I will amount to $n * 768$, while in the setting where SRL information is added, I equals to $n * (768 + 60)$. n denotes the number of tokens, which means that in a +SRL setting with a maximum length of 100 tokens, $w \in \mathbb{R}^{82800}$ and therefore also $I = 82,800$.

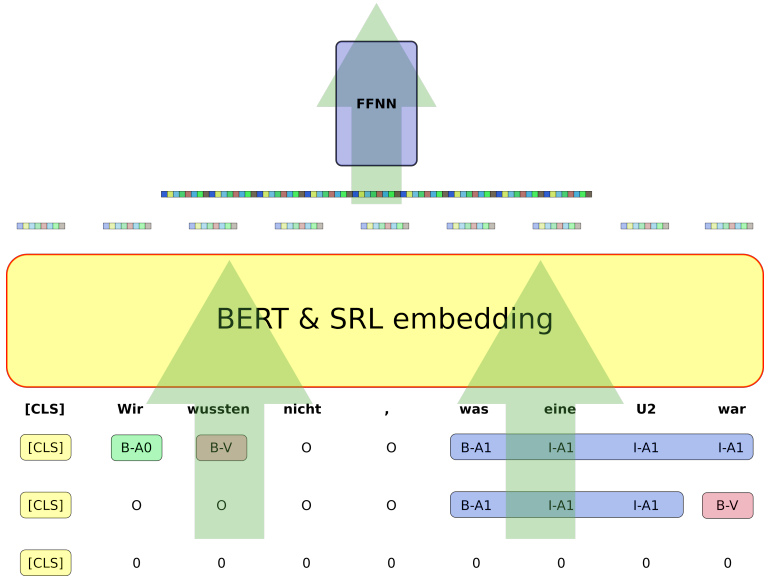

Figure 16: Head with a one-layer **F**eed **F**orward **N**eural **N**etwork on the concatenated token sequence. In difference to the vanilla BERT classification head, not only the embedding of the [CLS] token is used for predicting classes, but the whole sequence of vectors is concatenated and fed into the final head layer.

The motivation for the FFNN head lies in the fact that the SRL information, which is essentially a mark-up of a sequence, is condensed inot one vector for the [CLS] head. Intuitively, I would suspect that it is difficult for the model to extract reliable, useful information out of this single token. Consider the following example (the pseudo SRL [CLS] is added):

$[_{[CLS]}\ ]$ $[_{A\text{-}0}$ The man$]$ $[_{predicate}$ asked$]$ $[_{A\text{-}1}$ his friend$]$.

After the subscripted SRLs were consumed by the BiGRU, there is some information about all SLRs in the hidden states of the [CLS] token. While there may be some information about there being a predicate, an argument zero, and an argument 1 present, it is impossible to determine from which tokens these signals came. Especially in sentence pair tasks, such as paraphrase identification, however, this information is absolutely crucial. By concatenating all embedded tokens into one large vector, the positional information of the SRLs don't get lost.

As has been shown by e.g. Myagmar et al. [2019] for Sentiment Analysis, concatenating BERT embeddings and sending them trough a FFNN produces good results (in fact, it performs the best in the different architectures the authors tested for their task and the dataset on which they evaluated.).[11]

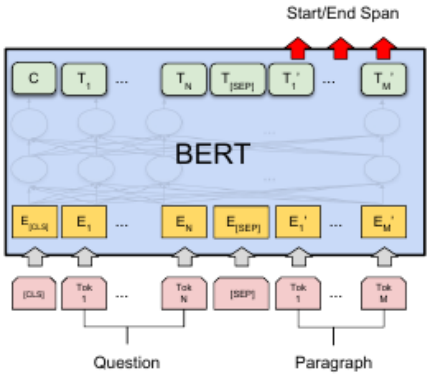

Figure 17: Vanilla BERT question answering head. Figure taken from [Devlin et al., 2018].

4.5.1.3 GRU Head

In the section before, it was assessed that SRLs can be understood as being essentially a token-wise, sequential "mark-up" of a sentence. Therefore, encoding a sequence marked up with SRLs employing an architecture designed for sequential data is not too far off.

The concept of Recurrent Neural Networks (RNNs), i.e. networks whose connections form a directed graph such that output of the model is re-fed as input into itself, has been around since the late 1980s, with [Hopfield, 1982] often being credited as having implemented the first recurrent neural network. RNNs were developed for being able to process sequential data, where not only the information present in the single parts are important, but also the sequential ordering of those parts conveys

[11] In contrast to GliBERT where the FFNN consists of only one layer without activation function, Myagmar et al. employ a two-layered FFNN with the ReLU activation function in between.

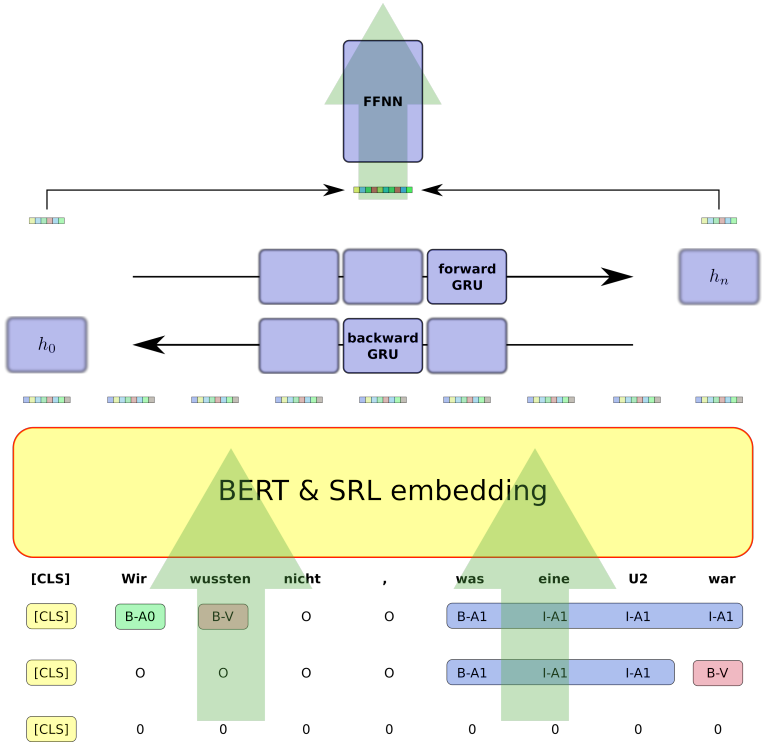

Figure 18: Head with a one-layer **F**eed **F**orward **N**eural **N**etwork on the concatenated last hidden states of the bi-directional GRU.

information.[12] To overcome the problems of the vanishing and exploding gradient problems that vanilla RNNs expose, [Hochreiter and Schmidhuber, 1997] proposed the Long Short-Term Memory LSTM architecture, which implements a special cell architecture which can mathematically "forget" and "memorize" the gradients to which the weights get updated via backpropagation. Before the advent of the transformer architecture, LSTMs with additional attention mechanism where the SOTA architecture for many NLP and NLU applications.

Cho et al. [2014] presented the GRU architecture, which similarly to the LSTM implements a mathematical solution to the vanishing/exploding gradient problem, but being a slimmer architecture. As for the SRL embedding network in the SRL module 4.3, I chose GRU over LSTM because of the leaner architecture.

Formally, the embedded token sequence $w_0, w_1, \ldots, w_n$ gets consumed by a forward and a backward operating GRU, which compute hidden states h_i for each token w_i in the sequence — so, for each token w_i there are two hidden representations: $\overrightarrow{h_i} \in \mathbb{R}^{712}$ for the forward and $\overleftarrow{h_n} \in \mathbb{R}^{712}$ for the backward GRU, respectively. The

[12] Cf. English "The dog bites the cat." vs. "The cat bites the dog."

last hidden states of these networks, $\overrightarrow{h_n}$ and $\overleftarrow{h_n}$ get concatenated, and this vector $w = \overrightarrow{h_n} \cdot \overleftarrow{h_n}$ gets fed into a final dense layer, which computes class probabilities, similar to the [CLS] and FFNN head final layers.[13] This process is illustrated in figure 18.

4.5.2 Question Answering

As was mentioned earlier, there exist several different formulations of question answering tasks (cf. chapter 3, section 3.2.2). However, Question Answering tasks are often formulated as essentially token-wise classification tasks:

Devlin et al. encode the question and context text pair as one sequence separated by the special [SEP] token (as for sentence pair classification tasks). For extracting the answer span, a FFNN predicts for each token i in the sequence its probability for being the start of the answer span, and its probability of being the end of the answer span (see figure 17). After applying a $softmax$ over all start/end probabilities, the token with the highest start probability is selected as beginning of the answer span and the corresponding end token is selected.

4.5.2.1 Span Prediction Head

The standard implementation of the Q&A BERT head consists of one one-layer FFNN which predicts on each input token the probability of it being the start of the answer span and the end of the answer span, respectively (as seen in figure 19. After both probabilities are computed for all tokens, the ones with the highest probability get selected; no further logic is enforced, such as that the index of the start token should be smaller than the one of the end token, or that the answer span may not lie inside the question (i. e. before the first [SEP] token), etc. The computation of the $index_{start}$ and $index_{end}$ of the answer span can be formalized as:

$$index_{start} = max(softmax(\hat{y}_{0_{[0]}}, \hat{y}_{1_{[0]}}, \ldots, \hat{y}_{n_{[0]}})$$

$$index_{end} = max(softmax(\hat{y}_{0_{[1]}}, \hat{y}_{1_{[1]}}, \ldots, \hat{y}_{n_{[1]}})$$

where $\hat{y}_i = Ww_i + b$ for the i-th input token with $\hat{y} \in \mathbb{R}^2$ and the rectangular brackets denote dimension selection.

Since the standard implementation already implements a model that predicts on

[13] Note that the nth hidden state of the backward network is actually the state corresponding to the token t_0, i.e. the first token, in the sequence.

each token, the SRL-embedding enriching is relatively straight forward: First, the input layer of the small head FFNN needs to be adapted to the BERT embedding + SRL dimensionality, which results in the vector representation $\mathbb{R}^{768+60}$ for each SRL-enriched token. Secondly, when the setting of merging the BERT subtokens before adding the SRL embeddings is selected, the start and end span indexes have to be recomputed.

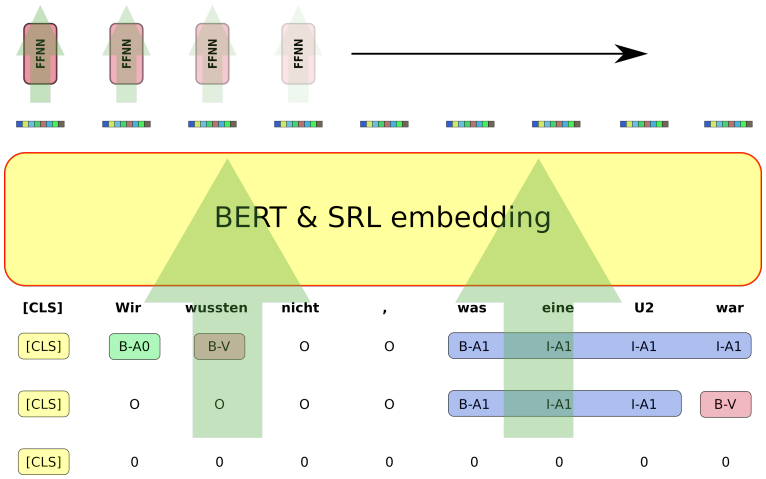

Figure 19: The GliBERT head for span prediction in Question Answering Tasks. After the Tokens and SRLs were consumed by BERT and the SRL embedding module, one FFNN predict on each token in the sequence, how likely it is first and the last token of the answer span. After these predictions have been made for all tokens, the token with the highest value for each position gets selected.

Code

In the file `gli_bert.py`, the head modules are defined as the following classes:

- `GliBERTClassifierCLS`
- `GliBERTClassifierFFNN`
- `GliBERTClassifierGRU`
- `GliBERTSpanPrediction`

In the configuration file `config.json` the head to be used is to be specified under the attribute `bert_head` (remember to pass it as string, e.g. `"GliBERTClassifierCLS"`).

5 Results

In this chapter I report the outcomes of my experiments on the GerGLUE dataset. Because there were several different implementations of various parameters tested against each other (SRLs duplicated vs. SRLs zeroed, BERT tokens merged vs. SRLs splitted, etc.), I provide several aggregated "views" on the plain numerical results, where one or more of these implementations is paid attention to. Further, I document the ourcomings of a human assessment of the quality of the DAMESRL produced SRLs to be able to make propositions about the usefulness of those and their effect on the results. Similar to this, the general data quality of random samples of two datasets is manually reviewed, for the same reasons. Eventually, I carry out a small ablation study, controlling the effect of ghosting out different SRL structures to show that the actual, semanticity providing power of SRLs lies in the combination of these structures.

For each dataset, two rounds of experiments where computed with one hyperparameter difference, resulting in an α and β for each dataset. The hyperparameters which were kept stable over all experiments are listen in table 7.

learning rate	2e-05
SRL embedding dimensions	20
SRL GRU hidden size	32
SRL number of layers	2
SRL bias	True
SRL bidirectional	True
SRL dropout	0.1
# of Epochs	50 (deISEAR, SCARE, XNLI, MLQA, XQuAD), 20 (PAWS-X)
Batch Size	16 (classification datasets) 8 (Q&A datasets)
maximum sequence length	100 (PAWS-X, XNLI) 512 (MLQA, XQuAD)

Table 7: Stable hyperparameter configurations.

In table 8 the different hyperparameter for the two runs is listed for each dataset.

Since the different dataset types were tested with different heads, I group the them the following way: Results for classification datasets (deISEAR, SCARE, PAWS-

	α	β
deISEAR	maximum length: 40	maximum length: 200
SCARE	maximum length: 50	maximum length: 100
PAWS-X	Examples train set: 8,000	Examples train set: 48,977 (all)
XNLI	Predefined splits	Resplitted 70:15:15
MLQA	Predefined splits	Resplitted 70:15:15
XQuAD	SRLs encoded sentence wise	Regular GliBERT SRL encoding

Table 8: Differences between α and β runs for each dataset.

X, XNLI) are reported together, and the results for question answering datasets (MLQA, XQuAD), are grouped together. Since the latter group consists of only two datasets addressing the same task, and bearing in mind that there were only experiments for one head in this group, the Span Prediction Head, the discussion analyzing these results will not be as interesting and substantial as for the first group. Because of the limited scope of this thesis, and for the sake of straightforwardness, only test set results are reported in the tables. However, model selection was always based on development set results as is visualized in figure 20:

Based on the progression of the accuracy value on the development set the epoch with the highest accuracy on the development set was chosen and the corresponding test set accuracy was taken as best performance for this experiment on the dataset. The vertical line in those three example experiments[1] accuracy and loss progression on development and test results compared). marks this epoch, the horizontal drawn through line marks the development accuarcy (the "peak" of the red curve), and the horizontal dotted line marks the correpsonding test accuracy. On the right the corresponding losses for the same epoch are marked in the same fashion.

[1] **deISEAR** β +SRL normal duplicate [CLS] head, **SCARE** β +SRL merge zeros GRU head, **XNLI** α +SRL merge zeros [CLS] head.

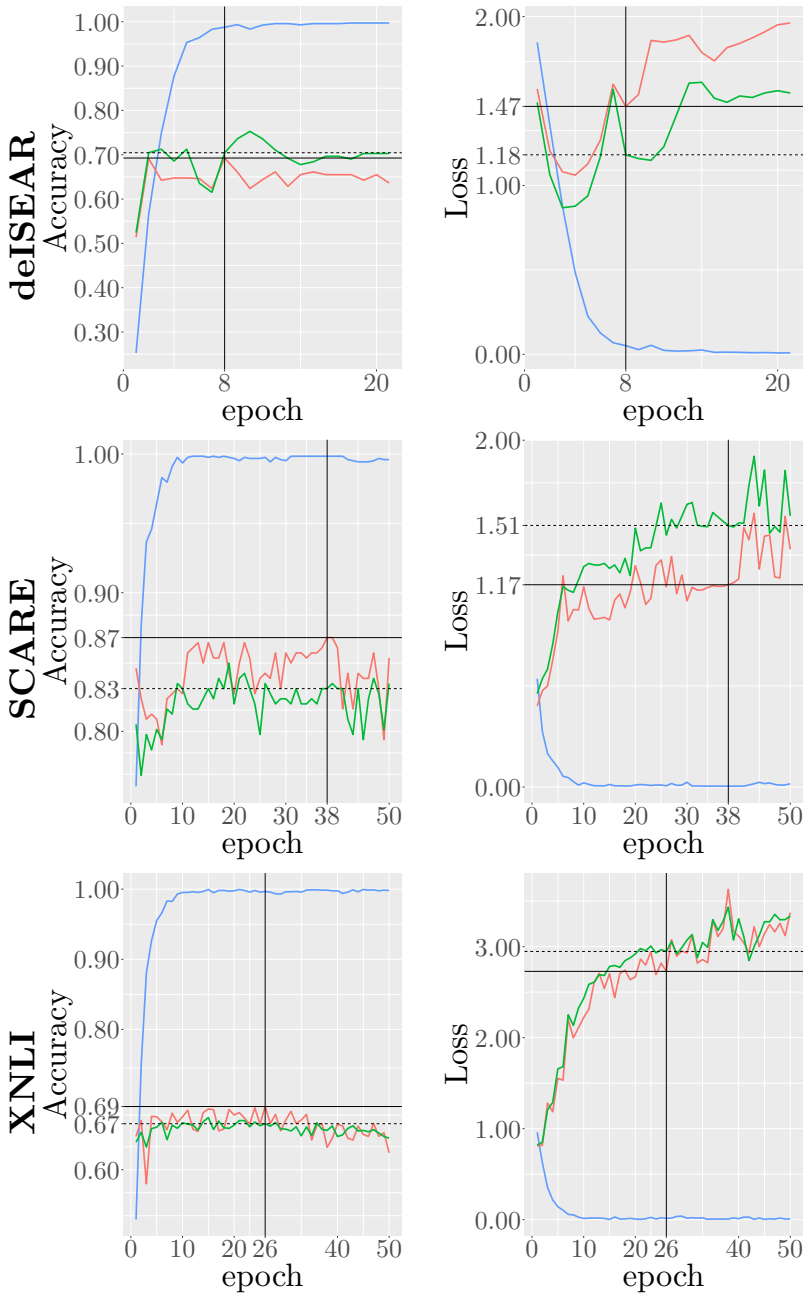

Figure 20: Accuracy and loss plots for three experiments (training, development, and test set): The vertical line marks the epoch with the highest development set accuracy; the test set accurracy for this epoch is taken as peak performance of the model.

In the deISEAR plot, we see that early stopping was triggered after the loss on the development set increased for four contiguous epochs — indicating an overfitting of the model on the training set. Both SCARE and XNLI experiments ran until the maximum number of epochs were reached without triggering early stopping. However, the SCARE accuracy and loss progressions show much more fluctuations and erratic behavior than the XNLI ones, indicating the noisiness of the data and the randomness of the function that needs to be learned by the model.

Code

To evaluate several models in ensemble mode, run the following command (make sure the `conda` environment is activated):

```
$ python ensemble_predict.py -r  \
                    <file_1>.json \
                    <file_2>.json \
                    ...
                    <file_n>.json
```

5.1 Controlling for Statistical Significance

Before reporting the actual results, I briefly elaborate on my strategy for assessing statistical significance, since this is a crucial aspect of empirical analysis in general, and in attributing superiority of one model over the other on the basis of measured performance on a dataset in particular: "if we rely on empirical evaluation to validate our hypotheses and reveal the correct language processing mechanisms, we better be sure that our results are not coincidental" [Dror et al., 2018]. In a formalized way, if we have a performance measure M — in my case accuracy — to evaluate the predictions of a Model A on a dataset X, we want to assess, if this performance was better than the one of a model B:

$\delta(X) = M(A, X) - M(B, X)$

where $\delta(X)$ refers to the test statistic, which will be shortly explained. Thus, the null-hypothesis which needs to be rejected, and the corresponding alternative hypothesis (that model A *is* in fact truly superior to model B) are formulated:

$H_0 : \delta(X) \leq 0$

$H_1 : \delta(X) > 0$

However, it is normally not the case that one is able to answer this question — is the observed performance measurement due to true model improvement or simply due to chance — by simply looking at the pure results; "[b]ut with statistics, one can answer the following proxy question: if the new technique was actually no different than the old technique (the null hypothesis), what is the probability that the results on the test set would be at least this skewed in the new technique's favor?" [Yeh, 2000]

In other words, to be able to reject the above defined null-hypothesis, usually a p-value test is computed, where p is defined "as the probability, under the null hypothesis H_0, of obtaining a result equal to or more extreme than what was actually observe" [Dror et al., 2018, p. 1384]:

$Pr(\delta(X) \geq \delta_{observed}|H_0)$.

There are different approaches as to how this probability can be computed: [Koehn, 2004], e.g., proposes a "paired bootstrap resampling" algorithm he applies to machine translation systems. I follow here Morgan [2005] who argues in favour of an "approximate randomization " approach, which makes less assumptions about the distribution of the sample distributions.[2]

The basic idea is to go through the paired predictions of two models (on the same test set) and randomly (i.e. with a 50% chance) flip the predictions between them. When this has been completed for the whole test set, the difference δ_{perm} of the evaluation metric between the recomputed performance of them is calculated again and it is checked whether $\delta_{perm} \geq \delta_{orig}$. This process is repeated R times and if the number of times δ_{perm} was greater/equal is small enough, in statistics, normally 5%, for a large enough number R, we can reject the null-hypothesis with good confidence. See the following algorithm 2 for a more detailed formulation of this process:

5.2 Classification Dataset Results

To obtain as stable results as possible, I decided to train five models for each architecture and configuration, all initialized with different random seeds. Additionally, I ensembled the five models, achieving a performance gain of several percentage points (see example of PAWS-X, table 10).[3] In table 9 below, the test set ensemble results for each architecture on the classification datasets are reported.

The best, second best and worst performance over all three heads and SRL configurations are marked. In the last row, best and second best results per head and +/-SRL for each subtokenization settings are accumulated. As the Scores indicate, taken the results without controlling for significance,[4] the +SRL configurations

[2] Koehn presupposes that the results to be compared stem from different subsamples of the original test set which get re-drawn repeatedly with replacement. In my case, this does not hold since my results are not obtained from subsamples but the whole test set.

[3] For ensembling, a straightforward majority vote of the five models is implemented.

[4] Here I don't control for statistical significance because it is not clear, against what model(s)

Algorithm 2 Approximate Randomization Algorithm

$p(M, x)$ = prediction of model M on example x
A, B = Two different models
$O = \{x_1, \ldots, x_n\}$ = test set
$O_A = \{p(A, x_1), \ldots, p(A, x_n)\}$
$O_B = \{p(B, x_1), \ldots, p(B, x_n)\}$
O_{gold} = gold labels for $\{x_1, \ldots, x_n\}$
$e(\hat{Y}, Y)$ = evaluation function for gold labels $\hat{Y}$ and predictions Y
$t_{original} = \mid e(O_{gold}, O_A) - e(O_{gold}, O_B) \mid$
$rand()$ = returns 0 or 1, randomly
$swap(x, y)$ = exchanges elements $x \in A, y \in B$ such that $y \in A, x \in B$
$r \leftarrow 0$
$R \leftarrow 0$
$threshold \leftarrow 1,000$
$p \leftarrow 0.05$
while $R < threshold$ **do**
  for all $(a_i, b_i) \in O_A \times O_B \mid i \in I$ **do**
    if $rand() = 0$ **then**
      $swap(a_i, b_i)$
    end if
  end for
  $t_{permute} = \mid e(O_{gold}, O'_A) - e(O_{gold}, O'_B) \mid$
  if $t_{permute} \geq t_{original}$ **then**
    $r \mathrel{+}= 1$
  end if
  $R \mathrel{+}= 1$
end while
if $\frac{r+1}{R+1} < p$ **then**
  system A truly better than system B
end if

achieved 5 times the best, and 10 times the second best results — compared to 4 best and 5 second best results for -SRL. This is a first indicator that adding SRL information seems to be able to support plain BERT embeddings in classification tasks.

The accumulations in roman numerals also indicate a further, observable trend: The GRU head accumulated 5 best and 6 second best results, independent of with or without SRLs, appears to be better suited for classification tasks than the [CLS] (2 best, 6 second best) and FFNN (2 best, 3 second best) heads. Supporting this, the GRU head never was responsible for the worst performance, which were all achieved by the FFNN and [CLS] heads.

In the following table 10, the ensembling of each of the results in the main table is exemplary disaggregated. Each head-SRL configuration is randomly initialized 5 times, resulting in 5 models achieving slightly different performances on the dataset. By ensembling them, a steady gain of, in this case, 1.26% compared to the average of the 5 models is achieved. Similar ensembling improvements where observed in all other datasets.

should a peak performance be controlled for: Take the 77.45% accuracy of the deISEAR α, achieved by the +SRL, BERT tokens subtokenized, FFNN head: Against what should this results be compared? To all other -SRL results? Only to the corresponding -SRL results for the same head?

Classification Datasets

		[CLS] Head						**FFNN Head**						**GRU Head**					
		subtokenized			subtokens merged			subtokenized			subtokens merged			subtokenized			subtokens merged		
		−SRL	+SRL		−SRL	+SRL		−SRL	+SRL		−SRL	+SRL		−SRL	+SRL		−SRL	+SRL	
			zeros	dupl.		zeros	dupl.		zeros	dupl.		zeros	dupl.		zeros	dupl.		zeros	dupl.
deISEAR	α	71.52	72.19	71.52	72.19	*67.55*	72.19	70.86	**77.48**	72.85	74.17	72.85	74.17	70.20	74.83	74.17	73.51	70.20	71.52
	β	71.52	74.83	74.83	70.86	70.86	72.85	74.83	*68.21*	70.20	74.83	73.51	70.86	73.51	74.83	72.19	**76.82**	70.20	74.83
SCARE	α	85.61	82.58	83.71	83.33	83.71	85.61	83.33	83.71	84.09	84.09	*81.44*	84.47	83.71	83.33	84.09	**85.98**	84.09	83.71
	β	86.36	84.85	84.47	85.23	85.23	85.23	**86.74**	85.98	85.23	84.47	*83.33*	84.09	**86.74**	85.98	83.71	86.36	84.09	85.23
PAWS-X	α	80.63	81.60	81.49	*79.92*	80.63	82.51	81.19	80.78	80.07	80.43	80.02	80.68	82.26	82.77	82.77	82.82	82.87	**83.53**
	β	87.49	88.05	**88.21**	87.75	87.24	88.00	86.83	87.39	87.09	87.75	*86.58*	86.68	87.60	87.60	87.90	88.00	88.00	88.05
XNLI	α	67.34	**67.52**	66.64	66.94	66.94	*66.26*	67.20	67.42	67.34	66.38	67.08	66.92	66.68	66.60	67.14	66.42	66.54	66.26
	β	68.09	68.18	66.84	67.82	67.82	68.36	66.31	65.60	66.40	*64.98*	65.51	65.07	66.84	67.82	67.02	67.64	66.31	**68.53**
Scores		II	**II** II			II		**I** I	**I** I		I			**I**	II		**II** I	**II** III	
+SRL	**5** 10																		
−SRL	**4** 5																		

Table 9: Test set accuracy ensemble results (per 5 models) on single sentence and sentence pair tasks. **Bold** font marks the best result per line, underline the second best, and *italics* the poorest. In the *Scores* row, the afore mentioned positive extremes are accumulated for −SRL and +SRL; note that if both +SRL configurations of an architecture achieved an extreme, it is only counted once.
The line marked with light gray — PAWS-X β — is "expanded" in table 10 to illustrate that each result in this table is actually a majority voting out of an ensembeling of fice models.

PAWS-X β

	[CLS] Head						**FFNN Head**						**GRU Head**					
	subtokenized			subtokens merged			subtokenized			subtokens merged			subtokenized			subtokens merged		
	−SRL	+SRL		−SRL	+SRL		−SRL	+SRL		−SRL	+SRL		−SRL	+SRL		−SRL	+SRL	
		zeros	dupl.		zeros	dupl.		zeros	dupl.		zeros	dupl.		zeros	dupl.		zeros	dupl.
Model 1	85.41	85.36	86.02	86.43	86.53	87.04	85.77	85.61	85.77	86.22	84.70	86.53	87.54	87.49	86.43	85.51	86.93	86.53
Model 2	86.07	86.83	86.99	85.87	86.32	86.68	86.17	85.82	87.39	85.92	85.36	85.26	87.04	86.99	85.87	87.19	86.07	87.44
Model 3	86.07	87.49	86.99	87.14	85.26	86.73	86.22	84.90	85.36	85.41	85.31	85.71	86.12	85.66	86.83	86.48	87.09	86.93
Model 4	87.39	86.32	86.58	86.38	84.04	87.65	86.53	86.73	85.61	85.82	85.61	86.38	84.99	86.02	87.70	86.99	86.68	86.88
Model 5	86.63	86.43	86.58	86.99	85.26	85.56	86.18	87.09	84.75	87.04	85.77	85.82	86.12	85.82	86.73	87.34	86.73	86.58
Average	86.31	86.49	86.63	86.56	85.48	86.81	86.22	86.03	*85.78*	86.08	85.35	85.94	86.35	86.40	86.71	86.70	86.70	**86.87**
Ensemble	87.49	88.05	**88.21**	87.75	87.24	88.00	86.83	87.39	87.09	87.75	*86.58*	86.68	87.60	87.60	87.90	88.00	88.00	88.05
Gain	1.18	1.56	1.58	1.19	1.76	1.19	.65	1.36	1.31	1.67	1.23	.74	1.25	1.20	1.19	1.30	1.30	1.18
Average	1.26 (σ 0.28)																	

Table 10: The "expanded" PAWS-X β results. The light gray line corresponds to the one in table 9. As can be seen, the fluctuations between single models is not too big, which is an indicator that the architecture is fairly stable. Ensembeling reliably leads to a 1.26 percentage points gain on average, with a standard deviation of 0.26%.

To see whether there can be seen a tendency as to which SRL-BERT aligning strategy turns out to be more effective, the results of the main table 9 are aggregated in table 11: For each head, the +SRL results are compared, and the differences are reported in the table. For example, we look at the [CLS] head on deISEAR α and ask us, which strategy — merging the BERT subtokens back ot "normal" tokens, or splitting the SRLs up to align with the subtokenized BERT tokens — worked better. For this, the difference between both for +SRL zeros and +SRL duplicate is calculated and controlled for statistical significance between both ensembles. The subtokenizing strategy (yellow) outperformed the merging strategy (purple) 28 times, 15 times statistically significant, while merging was better 20 times, only 15% of these were statistically significant. Therefore, at least for the GerGLUE classification datasets, splitting the SRLs up to align with the BERT subtokens seems more effective than merging the BERT subtokens. This also makes sense intuitively — using the merging strategy of averaging the BERT embeddings of subtokens leads clearly to an information loss or distortion that cannot be fully balanced by the added SRL information.

		[CLS] Head		**FFNN Head**		**GRU Head**	
		zeros	dupl.	zeros	dupl.	zeros	dupl.
deISEAR	α	4.64**	.67	4.63*	1.32	4.63*	2.65
	β	3.97*	1.98	5.30*	.66	4.63*	2.64
SCARE	α	1.13	1.90	2.27*	.38	.76	.38
	β	.38	.76	2.65**	1.14	1.89	1.52
PAWS-X	α	.97*	1.02**	.76	.61	.10	.76
	β	.81**	.21	.81*	.41	.40	.15
XNLI	α	.58**	.38	.34	.42	.06	.88**
	β	.36	1.52*	.09	1.33*	1.51*	1.51*

Table 11: Performance of architectures when BERT subtokenized vs. merged. Both SRL implementations were compared pairwise. Clearly, the subtokenized ensembles showed better classification performances than the merged ones.
Example case, upper left 4.64**: Comparison of deISEAR α, +SRL zero implementation [CLS] Head from table 9: Subtokenized ensemble (72.19% accuracy) and the merged ensemble (67.55% accuracy) — the subtokenized ensemble performed 4.64% better, apparently with quite high significance ($p < 5\%$).

Another accumulating view on the main table 9 is constructed by not just looking for the best results of a whole row, i.e. over all heads, merging strategies, and SRL implementations, but instead record the superiority or inferiority of +SRL compared to −SRL for each pairing: Take for example the deISEAR α subtokenized [CLS] head setting: Without SRLs, the ensemble performance was 71.52%; now the better of the two +SRL implementations of this setting is taken into account, in this case zeroing apparently was better. The difference, .67% is controlled for significance and

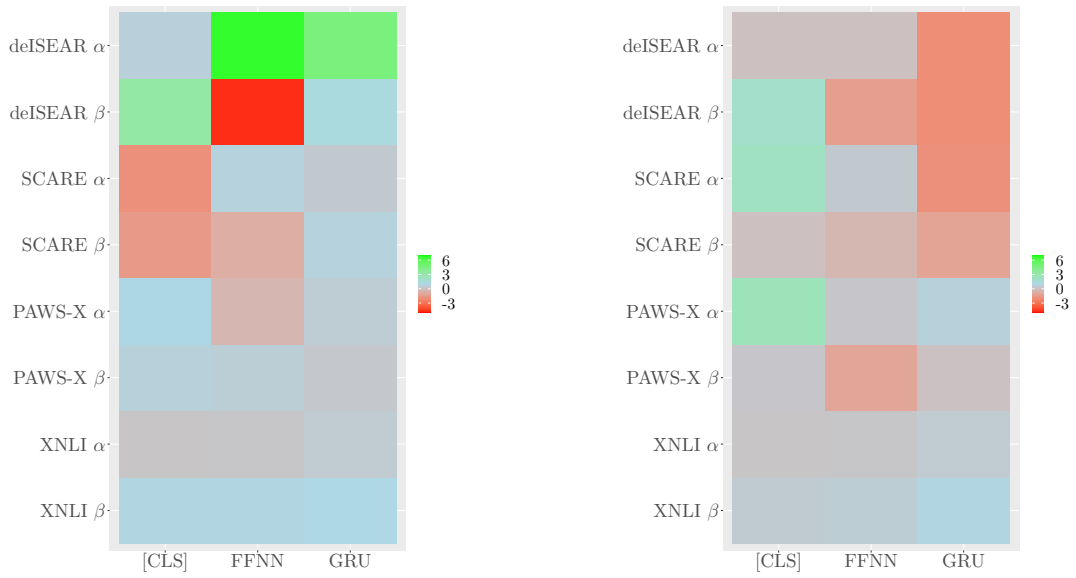

Figure 21: Gain/loss heat maps from the gain/loss values in table 12. **Left**: Subtokenized. **Right**: Merged.

reported in the table. The coloring indicates which SRL implementation performed better and was taken into account; sometimes both implementations performed equally, which is also reflected in coloring. Negative numbers indicate, of course, that the head produced better results when SRL information was *not* added (and this was also controlled for significance). Looking at the such accumulated results (table **??**), no clear picture emerges at first glance; sometimes zeroing SRLs performed better than duplicating SRLs, sometimes not. Sometimes +SRL clearly outperforms −SRL, even highly statistically, sometimes it is the other way around adding SRLs seems to create a disadvantage for the head.

In figure 21 the absolute gains and losses are plotted in two heat maps: In the heat map for the subtokenized settings, the pattern seems to be that for the single sentence classification tasks there is more fluctuation; especially for deISEAR FFNN, where for the α setting, there was the biggest gain in all subtokenized experiments, but also the biggest loss in all subtokenized settings for deISEAR β. The SCARE [CLS] head seems also clearly to suffer from subtokenizing SRLs tokens, while gining in other settings. However, for the sentence pair tasks, subtokenizing mostly adds a smaller, but constant gain, visualized by the uniformly distributed purple shades in this dataset rows.

For the merged settings, the correlation axis appears slightly to change: Before, positive or negative influence of the BERT-SRL combining technique has shown some correlation with datasets (tendency to lead to slight gains in sentence pair

tasks, and irregular behavior on single sentence tasks), now while this still holds slightly, it seems that especially the FFNN and GRU head are negatively affected by merging SRLs, while the [CLS] head shows a trend towards gaining from it. Note also, that the subtokenized settings shows a greater range of gains and losses (from −4.63 to + 6.62) compared to the merging approach (−1.99 to +2.59).

To add one more abstraction layer and see a more complete picture, this table is again summarized in figure 22. Doing this, the image clarifies, allowing to stipulate that adding SRLs on GerGLUE classification tasks leads to a measurable improvement. The total gains clearly outweigh the losses, and also more statistically significant. Regarding the SRL implementation, there is no clear advantage of one method over the other, only a slight tendency of duplicating being more effective than zeroing.

		[CLS] Head		**FFNN Head**		**GRU Head**	
		subtok.	merged	subtok.	merged	subtok.	merged
deISEAR	α	.67	.00	6.62**	.00	4.63**	−1.99
	β	3.31	1.99	−4.63*	−1.32	1.32	−1.99
SCARE	α	−1.90*	2.28*	.76	.38	.38	−1.89
	β	−1.51**	.00	−.76	−.38	.76	−1.13
PAWS-X	α	.97*	2.59***	−.41	.25	.51	.71
	β	.72*	.25	.56	−1.07***	.30	.05
XNLI	α	.18	−.22	.22	.70*	.46	.12
	β	.09	.44	.09	.53	.98	.89

Table 12: Ensemble percentage points gains (positive numbers) / losses (negative numbers) for +SRL over −SRL for each configuration from table 9. The better of the +SRL configurations was taken into account: zeros, duplicate. Light blue denotes that both architectures performed equally (in which case both ensembles were controlled for significance). One asterisk signifies a p-value $< 10\%$, two stand for $p < 5\%$ and three for $p < 1\%$.
Example case upper left 0.67%: The better +SRL ensemble for deISEAR α [CLS] head was the zero implementation (72.19%), surpassing the −SRL ensemble (71.52%) by 0.67 percentage point; but apparently not statistically significant.

As described in chapter 3, the datasets compiled in GerGLUE are a quite heterogeneous assembly: Some datasets like deISEAR and, especially, SCARE comprise rather informal, colloquial tests, while PAWS-X and XNLI consists of more technical, standardized text. Further, as depicted in figure 13, over 30% of the sentences in SCARE don't have any predicate-argument structures — in other words there is no exploitable SRL information for the model.

In figure 23, the same accumulation process as for the general overview can be made dataset-wise: For each double row — the α and β runs —, the gains and losses are

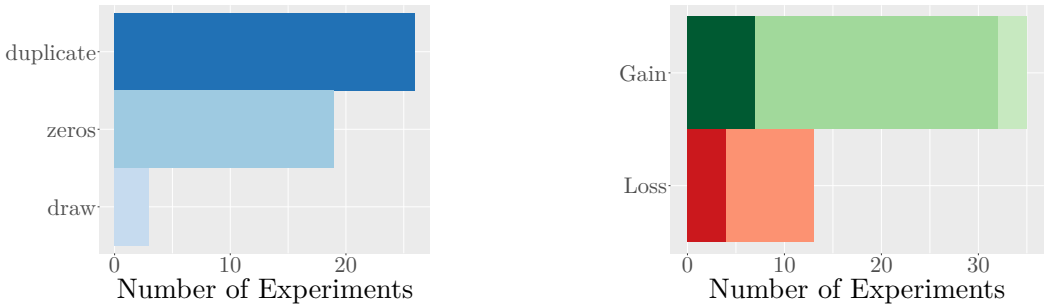

Figure 22: Accumulated scores from table 12. **Left**: Which SRL mode performed stronger out of a total of 48 settings. The bars indicate a slight outperforming of duplicating SRLs instead of adding zeros. **Right**: Counts for Gains and Losses in all 48 settings. Darker shades indicate significant results. The light green tip stands for settings where the gain equals 0.00.

added together, and the significant ones are highlighted.

The positive effect of enriching BERT embeddings with SRLs is especially evident for the sentence pair classification tasks: PAWS-X has the highest number of experiments where a significant improvement over vanilla BERT embeddings could be measured, and has overall clearly gained from adding SRLs. While XNLI has only one significant improved experiment, the overall number of experiments where a positive effect was measured outweighs the negative one by far.

For the single sentence tasks, however, the picture somewhat darkens. While there were some significant improvements for deISEAR, there were also quite some experiments where adding SRLs seems to have weakened the model. Generally, deISEAR is the dataset that shows the strongest fluctuations ranging from 67.55 to 77.48% — a range of staggering 9.93%. It is not clear to me why especially deISEAR shows this behavior, but the relative small amount of examples in the dataset could indicate that it is more prone to unlucky random initializations which could be countered with more examples for the models to extrapolate from. For SCARE, there is no clear positive effect measurable at all — in contrary, the significant deteriorations outnumber the significant gains. However, this is not too surprising taking into account the afore mentioned fact of predicate scarceness in SCARE (or at least, ParZu couldn't detect any). Further, the register is highly colloquial with very frequent deviations from standard orthography etc. which all probably leads to a collapsing of the already not too stable DAMESRL system.

In figure 24 the accumulation of gains and losses is done head-wise; i.e. for both BERT-SRL combination settings — subtokenized and merged —, the gains and losses are added together and the significant ones are indicated. The [CLS] head

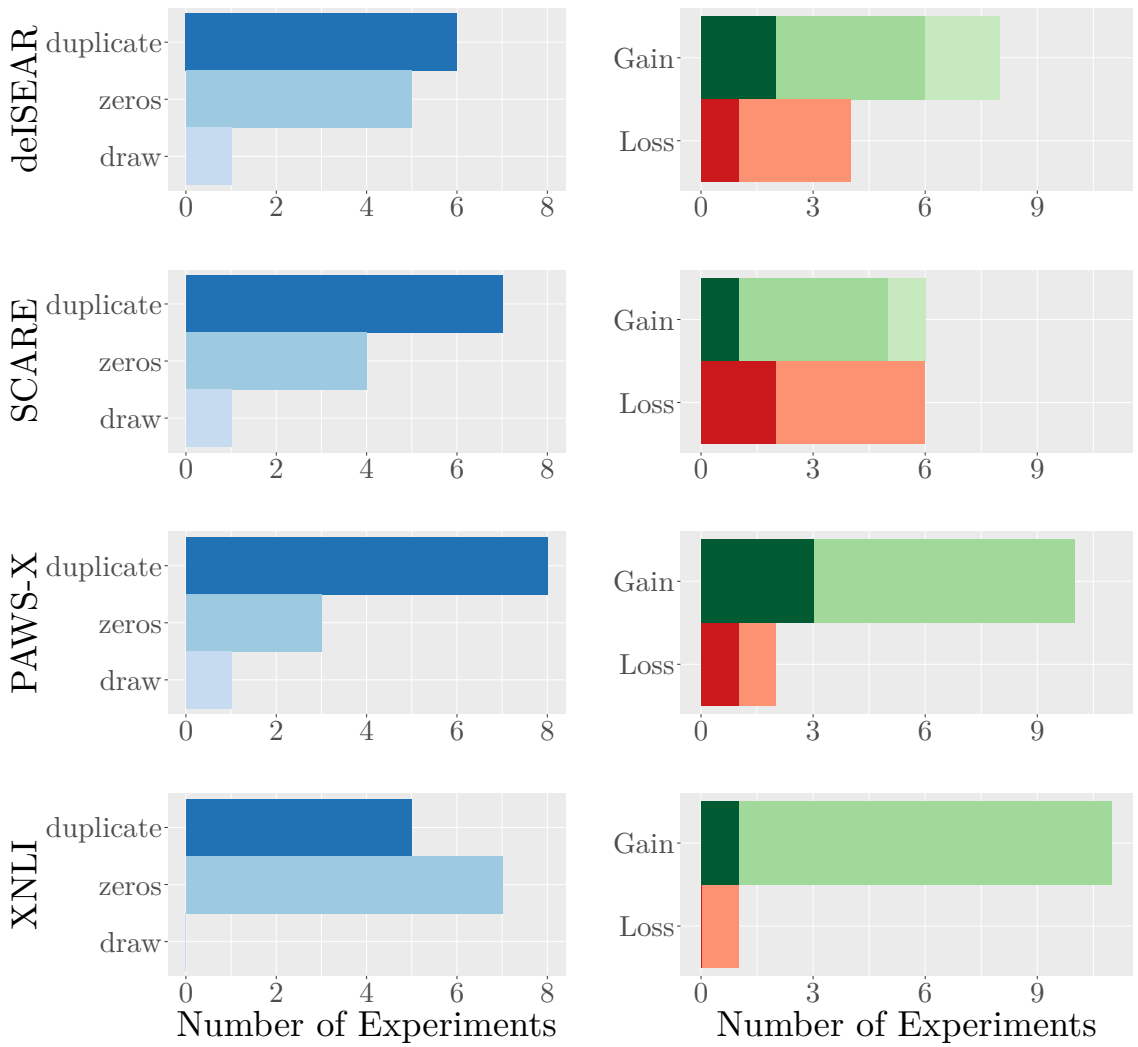

Figure 23: Accumulation of statistics for each classification dataset **Left**: Which SRL architecture performed better. **Right**: Comparison of accuray points gained/lost after adding SRLs. Darker Colors indicate significant runs; light green stands for a neutral run, i.e. where adding SRLs information performed equally to the vanilla architecture

amasses the most significant gains, while at the same time having led to 2 significant losses. Clearly, the FFNN head shows the weakest performance of the tested heads; the overall losses are the heighest of all the heads and the significant gains and losses are on par. The GRU head accumulated the most gains but only one of them being significant.

One might ask if a simple accuracy is the right measure for all datasets, since it is often common to compute other metrics, especially if class imbalance is a known property of datasets. There are two main reasons as to why I chose to stay with simple accuracy: (1) I am not interested in building a classifier which

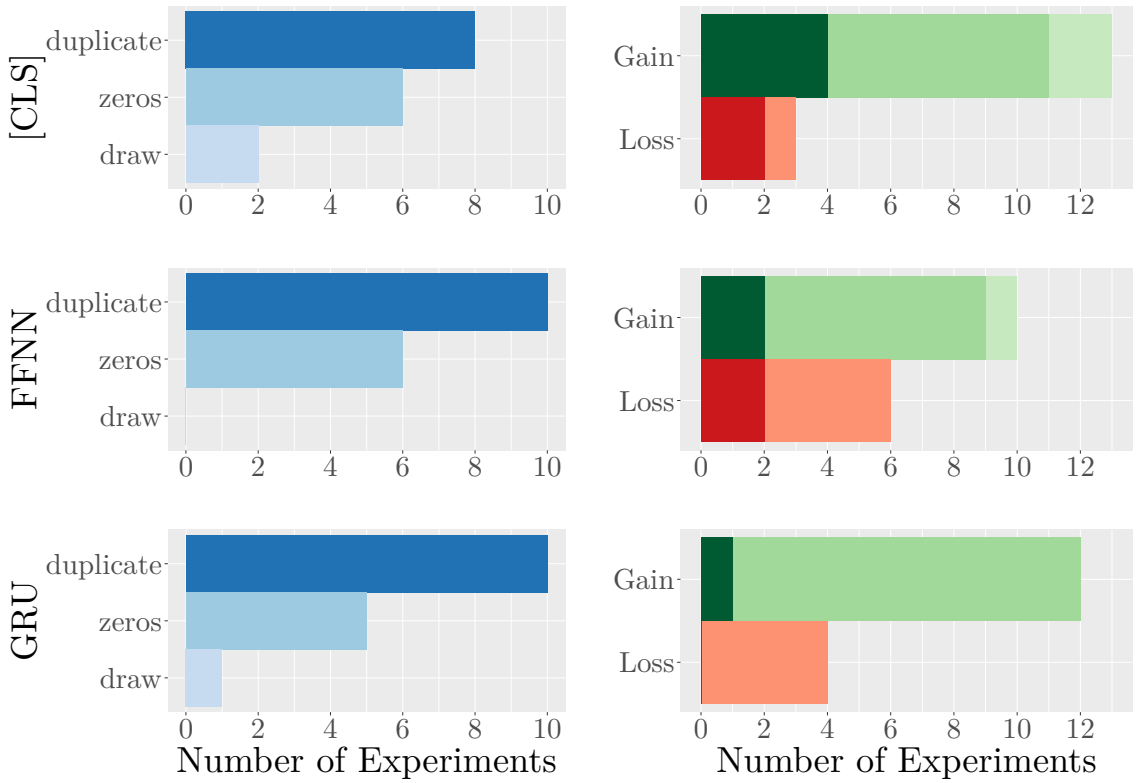

Figure 24: Accumulation of statistics for each head. **Left**: Which SRL architecture performed better. **Right**: Comparison of accuray points gained/lost after adding SRLs. Darker Colors indicate significant runs; light green stands for a neutral run, i.e. where adding SRLs information performed equally to the vanilla architecture

explicitly handles such a problem, e.g. spam detection tasks. In such cases, an F1-score of course is more meaningful because it is more sensible to predictions on underrepresented but task-important classes (such as spam). (2) As reported in chapter 3, most of my datasets do not show a drastic class imbalance. To further substantiate this, I compute some metrics for the most significantly imbalanced dataset, namely SCARE: A brute force "stupid" prediction of the majority class on the test set would results in an accuracy of 59.09% and a macro F1 of 24.76. The example confusion of an actual experiment is shown in table 13: Although the algorithm overpredicts the majority ("Positive") class, it also predicts to some extent the other ones. The measures resulting from this experiment are an accuracy of 83.71% compared to the macro F1 of 73.52. Therefore, although the accuracy may distort the picture to some extent, this is negligible in the context of this thesis.

		Predicted		
		Positive	Neutral	Negative
True	Positive	149	12	5
	Neutral	4	11	6
	Negative	3	8	66

Table 13: Confusion matrix for SCARE α +SRL merged duplicated [CLS] head.

Figure 25: Accumulated scores from table 15. **Left**: Which SRL mode performed stronger out of a total of 8 settings. The bars indicate a slight outperforming of duplicating SRLs instead of adding zeros. **Right**: Counts for Gains and Losses in all 8 settings. Darker shades indicate significant results. The light green tip stands for settings where the gain equals 0.00.

5.3 Question Answering Dataset Results

The reportings of the results for the question answering data sets follows the line that was taken for the classification sets: A general table with the ensemble test set results is reported in 14. Because there is only one head for the question answering task, the table is much smaller than for the classification task; however, the same facts are highlighted: The best, second best, and worst ensemble results in one hyperparameter setting (α or β) over all other configurations (merged/subtokenized, +SRL/−SRL, etc.) are highlighted.

In sharp contrast to the classification task restuls, the verdict here is far more devastating regarding the positive effect of SRLs: 3 out of 4 best results were achieved by models implementing vanilla BERT embeddings and 3 out of 4 times the worst performance was measured for +SRL settings. Only in the XQuAD β setting (regular SRL sequence encoding, see table 8), a +SRL setting performed best — however, when compared to the corresponding −SRL setting, this 0.29% gain is not statistical significant (see table 15).

When accumulated for unfolding subtokenized vs. merged patterns, there emerges

Q&A Datasets

		Span Prediction Head					
		subtokenized			subtokens merged		
		−SRL	+SRL		−SRL	+SRL	
			zeros	dupl.		zeros	dupl.
MLQA	α	**30.69**	29.68	29.68	21.92	21.92	*21.81*
	β	**44.75**	44.55	43.41	*41.66*	41.86	41.79
XQuAD	α	**42.01**	41.42	41.12	37.87	36.98	*35.50*
	β	46.57	45.43	**46.86**	37.43	*37.14*	39.14
Scores		**III** I		**I** III			
+SRL	**1** 3						
−SRL	**3** 1						

Table 14: Test set accuracy ensemble results (per 5 models) on question answering tasks. **Bold** font marks the best result per line, underline the second best, and *italics* the poorest.

a surprisingly clear tendency (see left side of table 15): Splitting SRLs according to the BERT subtokenization always showed stronger results than BERT merging. 5 times out of the total 8 settings the differences were even highly significant with $p < 1\%$ and one time moderately significant with $p < 5\%$.

Also for the Q&A datasets, I compare setting-wise the performance of −SRL against the better +SRL experiment (see right side of table 15): The only significant result is the MLQA α subtokenized setting, where the SRL-enriching led to a loss of 1.01%. The other results are mixed, with −SRL performing 3 times better than +SRL. The only dataset hyperparameter configuration where for both SRL combination methods the +SRL implementation performed better was XQuAD β — however in both cases insignificant.

Therefore, the overall aggregation in figure 25 does not look too promising for the research question of this thesis: For the two question answering datasets in GerGLUE, the effect of enriching BERT embeddings with SRL information seems to rather harm the model instead of adding helpful information. Interestingly, this contrasts with the findings of Zhang et al., which reported a gain of 4.3% in accuracy on the SQuAD 2.0 set for their SemBERT implementation.[5]

[5]However, a "synthetic self training" technique for this results is mentioned which is not elaborated upon any more. For their "regular" SemBERT large implementation the reported gain is 1.9%

		Span Prediction Head			
		Gains/Losses		subtokenized/merged	
		subtok.	merged	zeros	dupl.
MLQA	α	−1.01***	.00	7.76***	7.86***
	β	−.20	.20	2.69	1.62
XQuAD	α	−.59	−.89	4.44**	5.62***
	β	.29	1.71	8.29***	7.72***

Table 15: Left part: Ensemble percentage points gains (positive numbers) / losses (negative numbers) for +SRL over −SRL for the Span Prediction Head from table 14. The better of the +SRL configurations was taken into account: zeros, duplicate. Light blue denotes that both architectures performed equally (in which case both ensembles were controlled for significance). One asterisk signifies a p-value $< 10\%$, two stand for $p < 5\%$ and three for $p < 1\%$. **Right part**: Performance of architectures when BERT subtokenized vs. merged. Both SRL implementations were compared pairwise..

Main conclusions:

- SRLs in tendency helpful
- not for Q&A
- results somewhat unclear

— which still would stand for a positive influence of SRL enriching which is not reflected in my experiments.

6 Discussion

Based on the results presented in the previous chapter 5, A first, preliminary insight can be formulated: Adding numericall embedded SRL information to the German BERT embeddings leads to overall enhancing in classification tasks. For question answering, however, the SRL information leads to a clear decrease in performance. Additionally, while there was no clear advantage of one aligning strategy over the other for the classification tasks, the situation for question answering is crystal clear:

The reason for this rather drastic pattern remains unclear to me; despite tediously controlling the merging algorithm and validating the re-computing of start and end indices, I could not make out any error.

One of the main differences between the classification and question answering datasets is the size of the examples: While for the classifications task the model needs to extract semantic information out of a handful sentences, for the question answering tasks this is rather the exception — normally the context from which the answer span must be extracted is multiple sentences long. My suspicion is that SRL information can highlight some crucial semantic relations present in a sequence which would be hard for the vanilla BERT to extract; maybe because it is in marked passive voice, not obviously negated etc. However, for extracting answer spans out of a long text sequence consisting of several sentences, each having potentially 3 predicate-argument structures present, the SRL information may not display a helpful signal to the head, which then relies more on information present in the vanilla BERT subtokens. This hypothesis is further substantiated by the analysis of BERT-SRL combination strategy superiority, depicted in table 15: Maintaining the BERT inherent subtokenization structure outperforms the merging approach by far, indicating that the head relies heavily on information encoded in the BERT subtokens.

6.1 GliBERT Noise Nuisance Analysis

After reporting the general results, aggregations and the somewhat disenchanting conclusion, I now try to investigate the reasons for this mediocre effect. After

investigating the data and SRL quality, I found that noise on several levels is present which probably led to too much randomness in the complex functions the models needed to approximate. I classify the observed data irregularities into the following **noise** categories:

register The textual styles vary greatly from utilizing complex, hypotactic sentence structures (e.g. XQuAD), to highly informal, elliptic — even erratic — structures (e.g. SCARE).

data set Many of my datasets were constructed either automatically (e.g. scrambling text automatically to create paraphrase pairs) or employing crowd-sourcing techniques. Either way, the process is prone to errors. There are, e.g., 84 sentence pairs in the training set of PAWS-X that are 100% identical, yet labelled as non-paraphrases.

translation Due to the mostly employed semi-automatic translation approach for creating the various datasets, errors have been introduced into the data ranging from typical translation errors (e.g. English "bishop" in the clerical context translated to the German chess figure counterpart "Läufer", not "Bischof") to eventually wrongly copied labels, since the overall meaning changed during the translation process (e.g. a sentence pair is no more contradicting but neutral), thus creating label noise.

SRL The SRLs obtained from DAMESRL are, conservatively formulated, questionable in their quality (e.g. modifiers are completely missing).

While it is normal for data in NLP to possess noise in some way or the other, this usually does not impact the overall performance of a model trained on it in a severe way, since information in language is encoded redundantly, and noise present in one channel can be compensated for by non-corrupted information from another channel.[1] However, if noise is present on too many channels at the same time, reliably recovering structural information may break down to some extent.

[1]For example, information about which constituent takes the syntactic function of the subject is often not marked by using only one possibility, e.g. though a case ending. Often, several strategies are present, as in German: The subject is marked morphological through case endings (which are not always salient), positional (in unmarked contexts), and structurally (in unmarked contexts, normally the proto-agent is realized as subject) (cf. Bussmann [2006]; Jaeger [2010])

6.1.1 Register Noise

German BERT, according to deepset, was pretrained on the German Wikipedia dump, the OpenLegalData dump and news articles. All of this corpora comprise grammatical, to the standards of orthography adhering text belonging to a rather formal, "professional" register. Three of the six datasets in GerGLUE, namely deISEAR, SCARE, and XNLI, however, can be described as representing a rather colloquial register of language, even displaying features of transcribed speech as e.g. in XNLI:

(6.1) Ja und wir haben es irgendwie behalten, äh irgendwie äh, es war eine Überraschungs-Geburtstagsparty für sie, das war es Sie liebte die Überraschungsfeier.

Naturally, since the German BERT module probably wasn't confronted with texts coming from such a register, the BERT embeddings will probably contain some noise regarding the classification task at hand. Also, ParZu is aimed at parsing "clean", correct sentences, showing of course problems at processing examples as the one above, which leads to the known error propagation and amplification through DAMESRL.

In other words, it is expectable for GliBERT to show difficulties on the aforesaid datasets, which was also observed during the analyses of the results in section 5.2.

Interestingly, the discrepancy between the German BERT pretraining text register and the text present in some GerGLUE datasets, the analysis of the German BERT tokenization statistics shows that also in those problematic datasets, there are very little out of vocabulary items, which would be encoded by the catch-all [UNK] BERT token:

6.1.2 Data Set Noise

As [Caswell et al., 2021] point out, apparently there is often a lack of prudence observable when multilingual datasets get compiled — especially for low resource languages they report devastating discoveries of ungrammatical and even non-sense texts for these.

However, especially for sentence pair datasets, one of the sentences needs to be constructed. This can happen by automatically construct these sentences (PAWS-X) or let humans come up with those sentences (as for XNLI). In both cases, there might happen errors, and the label assigned to a sentence pair may be actually

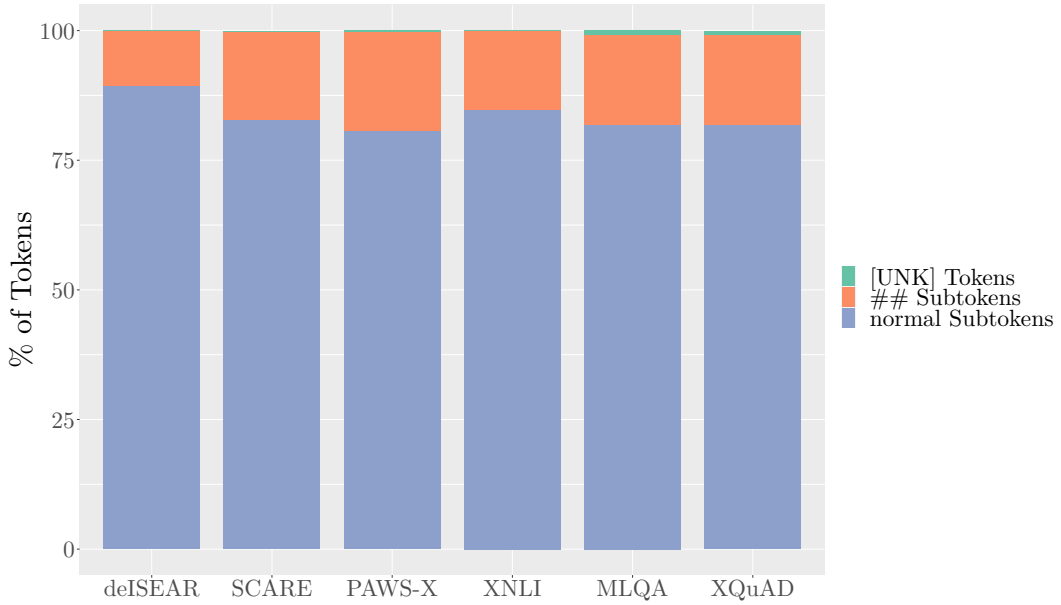

Figure 26: Percentages of token-types in all datasets. ## Subtokens represent the amount of tokens that get re-merged in the merged settings (e.g. "Master" "##arbeit" → "Masterarbeit". The amount of tokens that lie outside of the German BERT vocabulary is in all datasets extremely small (for deISEAR and XNLI there are no [UNK]s at all); the largest shares of such tokens are present in MLQA and XQuAD with .79% and .73%, respectively.

wrong. This is obvious for PAWS-X, where e.g. in the training set, there are 84 identical sentence pairs are labelled as non-paraphrases, which is obviously false, since the other 3,125 exact identical sentence pairs are labelled as paraphrases. E.g. sentence pair number 45061 (in the original English PAWS and the German PAWS-X datasets), both labelled as non-paraphrases:

(6.2) Riverton was a parliamentary electorate in the New Zealand region of Southland .
Riverton was a parliamentary electorate in the New Zealand region of Southland .

(6.3) Riverton war ein Parlamentswähler in der neuseeländischen Region Southland.
Riverton war ein Parlamentswähler in der neuseeländischen Region Southland.

But also for human created sentence pairs, there is a chance that the gold labels are not that clear. Further, through the creation of the German datasets by means of semi-automatically translating English datasets, additional translation noise might

additionally pollute the data.

To examine this more systematically for XNLI and PAWS-X, I conduct a re-labeling of 20 random examples each where the predictions of the best model contradicted the gold labels in the dataset.

6.1.2.1 Re-annotation

PAWS-X

For the 20 re-annotated examples in PAWS-X the Fleiss' κ of 0.68 indicates "substantial" agreement between the two human annotators; however, if the coefficient is computed between the two human annotators and the gold labels of those 20 examples, the value drops to 0.35, indicating substantive disagreement as to what sentence pairs are to be considered paraphrases and which don't.

Even more interestingly, in 30% of the considered cases, the model and the human annotators unanimously disagree with the gold labels present in PAWS-X:

(6.4) Der NVIDIA TITAN V wurde von Nvidia am 7. Dezember 2017 offiziell angekündigt.
Am 07. Dezember 2017, verkündete NVIDIA offiziell Nvidia TITAN V.

humans & model: False, Gold: True

(6.5) Die Schäfte sind sehr kurz oder oft nicht vorhanden.
Es sind entweder wenig Landschaften vorhanden oder sie fehlen in den meisten Fällen.

humans & model: False, Gold: True

(6.6) 1963 trat Roy der Kommunistischen Partei Indiens bei und leitete Gewerkschaftsbewegungen in Bansdroni in Kalkutta.
Roy trat 1963 der Kommunistischen Partei Indiens bei und leitete Gewerkschaftsbewegungen im Kolkata-Gebiet von Bansdroni.

humans & model: True, Gold:False

(6.7) Der Kanal ist einer der ältesten schiffbaren Kanäle Europas und sogar Belgiens.
Der Kanal ist einer der ältesten befahrbaren Kanäle in Belgien und Europa.

humans & model: True, Gold:False

(6.8) Propilidium pelseneeri ist eine Art der Meeresschnecken, eine wahre Napfschnecke und Gastropoden-Mollusk in der Familie der Lepetidae.
Propilidium pelseneeri ist eine Art der Meeresschnecken, eine wahre Napfschnecke und Meeres-Gastropoden-Mollusk der Familie der Lepetidae.

humans & model: True, Gold:False

(6.9) Die Chicago Bears sanken auf die Giants 27:21, und verloren 0:6 zum ersten Mal seit 1976.
Die Chicago Bears verloren 21:27 gegen die Bears und standen erstmals seit 1976 bei 0:6.

humans & model: False, Gold:True

In all of this cases it seems to be the case that the given gold labels actually are false, leading to the conclusion that the evaluation and measured performance on such a dataset must be taken with caution, and not just blindly accepted as ultimate truth.

XNLI

For XNLI, where in contrast to PAWS-X each example can have one of three labels instead of binary classification, the findings differ to some extent: While the agreement only between the two re-annotators is measured as a low Fleiss' κ of 0.36, it raises to 0.48 when the gold labels are also taken into account, indicating "moderate agreement". For one example the humans and model were on the same page regarding the label but disagreed with the gold label:

(6.10) Bato ist ein Jahrhunderte altes Wort, das man als Kerl oder Kumpel übersetzen kann.
Bato (oder Vato) ist ein spanisches Wort, das Typ oder Typ bedeutet.

humans & model: neutral, Gold: Entailment

Since the specification of "spanisch" is only encountered in the hypothesis but missing from the premise, I would argue that the humans and the model are right about the "neutral" label and the gold label is false. In another example, all three "agents" — the two humans, the model, and the gold labels — disagreed:

(6.11) Oh, ich sehe oh der Staat braucht es nicht gut, das ist eher das, das ist eher ungewöhnlich, nicht wahr?
Das macht Sinn, dass der Staat es benötigt.

humans: neutral, model: entailment, Gold: contradiction

The latter is also a further exemplification of the register diversity in XNLI: The premise above seems as if it was a transcription of an utterance torn out of the context an actual conversation; because of that for a human reader there is essential information missing, making it a rather strange and vague case for attributing a specific entailment regarding any hypothesis.

Both re-labelings have led to the finding that the quality of semi-automatically created datasets is often doubtful: (1) As the many "neutral" re-labelings of XNLI examples by the human annotators indicate, often there seems not be a clear preference towards a certain label — rather, as in the examples above, there is too much ambiguity and lacking of needed information around to make a clear decision of the relation between those sentences. (2) Further, the predefined labels in such datasets are by no means to be taken as undoubtedly indicating true classifications.

6.1.3 Translation Noise

Automatic translation has come a long way and systems nowadays produces often qualitatively high results — at least for translation between high resource languages, such as English and German. But still, especially for delicate, subtle expressions, errors can happen and a (partial) mistranslation is produced. While this potentially leads to translation artifacts, which may confuse the model, there also exists the possibility of additionally introducing label noise.

Take for example the following XNLI English original premise-hypothesis pair and its aromatically translated German counterparts:

(6.12) and that's a lot of it is due to the fact that the mothers are on drugs
The mothers take drugs.

(6.13) Und vieles davon liegt daran, dass die Mütter Medikamente nehmen.
Die Mütter nehmen Drogen.

The English term "drug" indeed is an ambiguous word, which either can refer to a pharmaceutical product that is used as medicine or to an abusively taken substance (which may be of the first type).[2] However, since the label of the original sentence pair is *entailment*, the term would have needed to be translated in the same meaning both times. However, this did not happen, and thus a wrongly labelled training

[2] `https://dictionary.cambridge.org/dictionary/english/drug`

instance was created.

6.1.4 SRL Noise

A major question arising in the context of using automatically assigned Semantic Roles in downstream tasks, is how precise these predicted Semantic Role are. Since there is no gold standard available for Semantic Role Labels for the datasets I use in my experiments, there is no straight-forward way to evaluate their quality automatically. In contrast to other tagging tasks like POS prediction or NER, Semantic Roles are not as black and white: While it is relatively easy to decide if a predicted POS tag is correct or incorrect, it is more a scale concerning SRLs.

As for the data quality / label noise assessment, I conduct an human evaluation study for determining the SRL quality: Three people evaluated the SRL quality of examples regarding whether they estimate them to be helpful, neutral, or harmful to the model for solving the task at hand.[3]

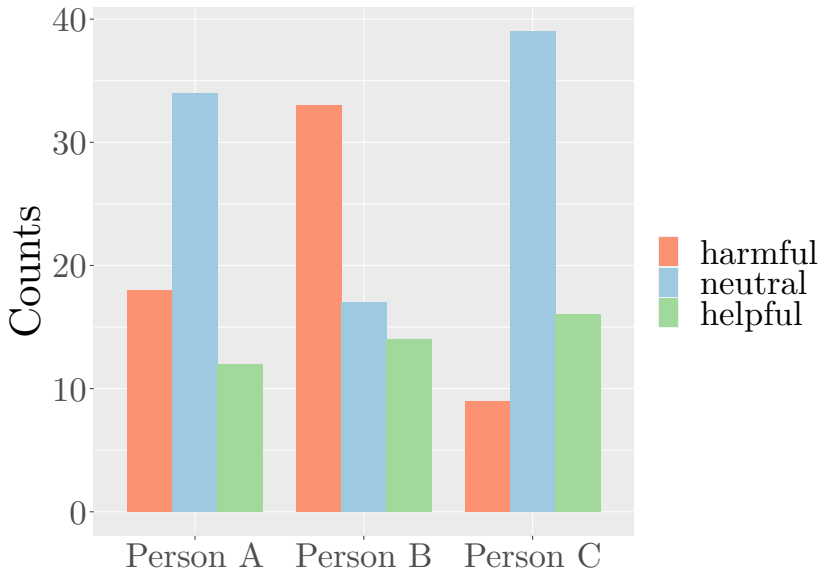

Figure 27: Independent evaluation of SRL quality by three people. Regardless of the label attributed to each example, it is obvious, that the total amount of sentences for which the annotators evaluated the corresponding semanti roles as *helpful*, is relatively stable.

In figure 27 the aggregated evaluations per human are summed up: Evidently, the predicted SRLs were mostly estimated to be neither of great help to the model nor

[3]For single sentence tasks, 20 examples were randomly sampled, for sentence pair tasks 10 examples, and for question answering tasks 2 examples (since the context comprises often umpteen sentences.

severely distract it. It has to be noted, however, that the inter-annotator agreement between the the three people was very low, the computed Fleiss' κ was only 0.20 — indicating an agreement slightly above randomness.

If the annotations are summed up over each dataset, as in figure 28, this impression gets tightened: Only a small subset of the usefulness attributions were unisono, i.e. all annotators picked the same label (indicated by darker color shades).

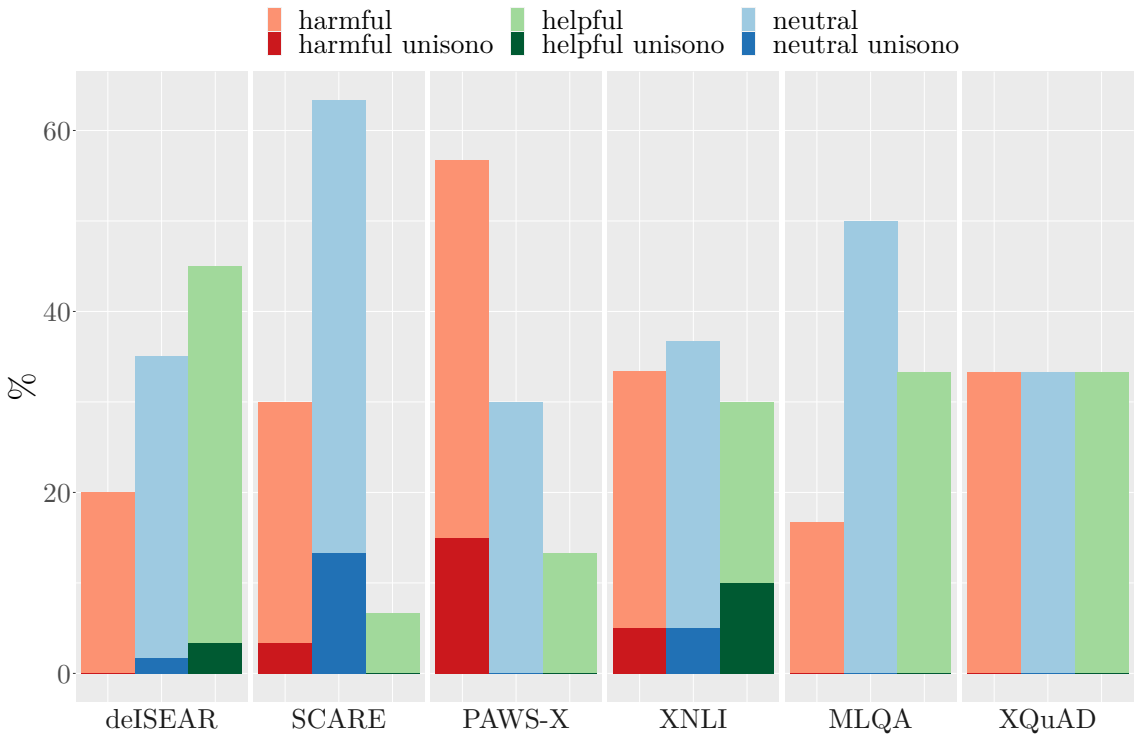

Figure 28: Estimated quality of SRLs per dataset.

Interestingly, for PAWS-X was comparatively high agreement between the annotators that the SRLs would rather harm the model, however, as is shown in the right figure of 22, PAWS-X is one of the datasets were the performance gain from adding SRL information was rather clear.

6.2 Ablation study

To be able to make substantial claims about the positive influence about a new algorithm over an established one, it is common ground to conduct an ablation study. In such a study, one tries to determine which aspects of the proposed architecture contribute how much to the overall performance gain (or loss, respectively).

Ich	B-A0	O
weiß	B-V	O
nicht	O	O
ob	B-A1	O
er	I-A1	B-A0
danach	I-A1	O
in	I-A1	B-A1
Augusta	I-A1	I-A1
geblieben	I-A1	B-V
ist	I-A1	O
.	O	O
==========	==========	==========
Er	B-A0	
wohnte	B-V	
weiterhin	O	
in	B-A1	
Augusta	I-A1	
.	O	

SRL 6.1: Normal SRLs.

In my case, i.e. the attempt to improve the performance of BERT regarding NLU tasks, the following question would need some ablation experiments to be answered: What part of the SRLs is most responsible for the performance boost? To be able to formulate this in a matter which can be experimentally tested, I identify two easily separable and testable aspects of SRLs, take for example the sentence depicted in figure 6.1: Firstly, the information what parts of a sentence are the predicates. The intuition behind this is that maybe the head relies mostly on the information as to which tokens carry information about the events that happen in a given sentence. To test this, I drop the information about all SRLs, except the information that a token is a predicate (left side of figure 6.2). In the second case, the hypothesis is reversed: Maybe the head is able to get the most useful hints about the information which indicates what role certain token groups play in a given sentence. To test for this all information about predicates is dropped and only information about arguments is preserved (right side of figure 6.2).

SRL 6.2: Left: Only predicate SRLs. **Right**: Only argument SRLs.

To see the influence of these SRL ghosting techniques, for each classification dataset, I take a configuration from table 9 where +SRL showed positive influence and re-computed 5 models for each configuration (preserving only predicate information vs. only argument information). In table **??** I report the results from this ablation experiments.

As can be seen, SRL information only displays its full positive effect when both information parts are combined: Providing the model only with predicate informa-

Ich	O	O
weiß	B-V	O
nicht	O	O
ob	O	O
er	O	O
danach	O	O
in	O	O
Augusta	O	O
geblieben	O	B-V
ist	O	O
.	O	O
Er	O	
wohnte	B-V	
weiterhin	O	
in	O	
Augusta	O	
.	O	

Ich	B-A0	O
weiß	O	O
nicht	O	O
ob	B-A1	O
er	I-A1	B-A0
danach	I-A1	O
in	I-A1	B-A1
Augusta	I-A1	I-A1
geblieben	I-A1	O
ist	I-A1	O
.	O	O
Er	B-A0	
wohnte	O	
weiterhin	O	
in	B-A1	
Augusta	I-A1	
.	O	

		−SRL	+SRL		
			only PREDs	only ARGs	normal
deISEAR α	FFNN Head subtok. zeros	*70.86*	72.19	75.50**	**77.48****
SCARE α	[CLS] Head merged duplicate	*83.33*	84.47	85.23	**85.61***
PAWS-X β	[CLS] Head merged duplicate	*79.92*	80.53	80.68	**82.51*****
XNLI β	GRU Head subtok. zeros	*66.84*	67.02	**68.00**	67.82

Table 16: Ablation on effect of PREDs and ARGs information isolated. Note that mostly only the combination of both information structures leads to a significant improvement over vanilly BERT embeddings (−SRL)

tion already leads to a slight, but insignificant improvement in all experiments. By telling the model exclusively about where arguments are located in sentences leads to a greater advancement, but mostly also insignificant. For three out of four datasets, informing the model about both SRL information structures in combination leads to a further significant increase of accuracy.

Apparently, the crucial aspect of information the GliBERT heads improve from having knowledge about semantic roles lies in its relational character: It is not so much an "annotation" of certain parts of sentences — as for e.g. named entities —, but the indication of the relationship in which those sequences stand to each other.

6.3 Summary

General effects of SRL information on BERT embeddings

For classification datasets, a modest, but clear gain of 0.8% in overall accuracy was determined.[4] For question answering datasets no positive, measurable effect could be asserted.

Zhang et al. [2019b], who implemented SRL-enriching for the English BERT model, report an overall performance increase of SemBERT of 2.4% on the GLUE classification datasets. Since the authors do not report significance and the model selection approach that led to these results, it is difficult to explain the performance gap between the two models. However, as is indicated by the results of the CoNLL '09 [Hajič et al., 2009] , English Semantic Role Labelling is significantly better than for German; which would partly explain the superiority of SemBERT over GliBERT.

Further, GliBERT showed varying performance according to the task at hand: The most reliable gains were found on sentence pair classification data sets, followed by single-sentence classification tasks. For question answering tasks no positive effect of SRL information could be measured.

However, not only the tasks at hand influences the effectiveness of SRL information, also the quality of the data has a decisive influence of the performance of the combined representations: Especially datasets that were created by (semi-)automatically translate English corpora into a target language expose translation noise, sometimes leading to wrong gold labels, which hinder reliable model fitting and selection.

Differences in head architectures

The BERT vanilla head for classification adapted to the GLiBERT [CLS] head showed mixed results: While producing 11 improvements out of 16 experiments, 4 of them significantly, there were also 3 instances where adding SRL information led to a performance decrease, where in two thirds of the cases, this decrease was statistically significant. The [CLS] head therefore is a relatively stable head, however, if the random initialization of the weights is not optimal or it gets stuck in a non-optimal minimum of the objective function, it produces severe negative results.

The FFNN head, where all token embeddings get concatenated is clearly the worst

[4] Only the significant results from table 12 were taken into account

classification head: The positive and negative effects of adding SRL information almost balance each other out — there is a slight bias for positive results.

Processing the token embeddings by a recurrent architecture is the most reliable approach: The GRU head improves in 75% of all experiments when SRL information is added. Often these gains are not statistically significant when considered in isolation, but the overall tendency of the GRU head is clearly suggests the NLU supporting character of SRLs.

For question answering, the GliBERT adapted vanilla head does not benefit from embedded SRL information — all improvements were statistically insignificant and in the majority of the experiments, adding SRL information led to a performance decrease.

Differences of SRL implementation modes

For merging the SRL embeddings with the BERT embeddings, two modes were proposed: Merging BERT subtokens back to token level by averaging the respective vectors, or duplicating the SRLs to match up with the BERT subtokens. Over all datasets, and for question answering datasets in particular, duplicating the SRLs and preserving the BERT subtoken embeddings clearly led to better results than the latter approach. Presumably, this has to do with the inevitable information loss during the subtokens avering process.

When a sentence has less than three predicate-argument structures, there were two approaches presented as well: Duplicating the first predicate-argument structure to fill the empty slots versus filling the empty predicate-argument structures with a special `O`-SRL. Here, duplicating the first predicate-argument structure has shown to be slightly more effective than the zero-filling approach on most datasets: Only for XNLI and the question answering datasets the duplicating strategy showed a weaker performance in tendency compared to the zero-filling approach.

7 Conclusion

The core contributions of this thesis consist mainly of three components: (1) The compilation of a German NLU dataset, *GerGLUE*, incorporating different tasks and modalities, as well as including domain-specific language, ranging from colloquial to highly stylized texts. (2) A pipeline that brings all data sets in the same workable format, computes SRLs by implementing two freely available tools (ParZu, DAMESRL) and several instruments for training and analyzing models. (3) *GliBERT*, a BERT-based architecture which combines vanilla BERT-embeddings with embedded SRLs during fine-tuning, comprising several heads, and convenient hyperparameter adjusting.[1]

Extensive experimentation and subsequent result analysis indicate that providing NLU-targeted models with contextualized word embeddings from German BERT merged with additional, numerically encoded SRL information leads to performance gains compared to the vanilla representations on most tested datasets. These results imply that self-supervised pre-trained representations learned from unstructured, raw text data generally profit from additional, explicit linguistic information made available during fine-tuning.

Further research elucidated the importance of the relational nature of SRLs: The core features of SRLs are (1) the marking of the predicate and (2) the highlighting of its arguments and their respective roles — while each of these two basic characteristics in isolation does moderately leverage the model's performance, only the combination of them leads to reliable significant gains.

Another key insight of this thesis is the confirmation of the well-known dependence of data-driven machine learning models on data quality: Several noise sources in the data of this thesis that hinder reliable weight parametrization were identified and analyzed. Especially the soundness of the automatically predicted, explicit linguistic information is crucial: The predicted SRLs of the implemented system

[1] All code relating to this thesis is availabel under `https://github.com/JonathanSchaber/Masterarbeit`.

are of questionable grade, e.g. completely lacking the ability of detecting modifiers and negation particles. A conceivable improvement in the predicted SRLs would presumably lead to a substantial overall NLU capability enhancement.

Outlook / Future Work

During the course of the experimental phase of this thesis, several ideas and possible paths disclosed itself which could not be pursuited due to the limited scope of this thesis. In the following list I scetch loosely those which seem to me worthy of investigation:

Better SRL-predicting system Re-run experiments with an off-th-shelf SRL-predictor that reliably predicts SRLs and modifiers, especially negation. My expectations are that the performance boost would be remarkably higher, given the poor quality of the used SRLs.

Manually add negation particle The negation particles could be detected in the ParZu parse tree and added manually as `MOD-NEG` to the SRLs.

Serious hyper-parameter search For my experiments, I tested several different hyperparameter settings (such as the SRL embedding size, learning rate, etc.) and implemented the ones that showed the best preliminary performance. Executing systematic hyperparameter space searching may lead to additional increases in performance.

Phenomenon-targeted data set creation For confidently assessing the counteracting of semanticity weak spots of BERT through SRL information, datasets compiled at targeting those would be of more explanatory power than general performance on regular NLU datasets.

Model weights analysis To further assess that the model relies on the additional SRL information for making better predictions, analyses of the actual weight matrices of the GliBERT model could be carried out.

Other word embeddings / linguistic information In this thesis the general hypothesis of providing contextualized word embeddings with explicit linguistic information was demonstrated for BERT and SRLs. It would be interesting, if the positive results could be replicated for different word embeddings like ELMo and other explicit linguistic information like e.g. knowledge graphs.

Reassured by the outcome of this thesis, I am convinced that future work in NLU

and NLP in general will see a comeback of linguistic instrumentary into the field: In order to reasonably handle the highly variable input of natural language, future NLP models will probably continue to implement some form of weight parametrization that will be streamlined data-driven by learning from real-world data. However, a complete averting of implementing linguistically motivated instruments is clearly not advisable, as the various results of analyzing the errors such models make unambiguously demonstrate. I see GliBERT as an exemplification of the fertile combination of deep learning techniques with structured linguistic knowledge representations.

References

M. Artetxe, S. Ruder, and D. Yogatama. On the cross-lingual transferability of monolingual representations. *arXiv preprint arXiv:1910.11856*, 2019.

C. F. Baker, C. J. Fillmore, and J. B. Lowe. The berkeley framenet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, 1998.

E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623, 2021.

Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2): 157–166, 1994.

Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *The journal of machine learning research*, 3:1137–1155, 2003.

R. C. Berwick, N. Chomsky, and M. Piattelli-Palmarini. Poverty of the stimulus stands: Why recent challenges fail. *Rich languages from poor inputs*, pages 19–42, 2013.

Y. Bisk, A. Holtzman, J. Thomason, J. Andreas, Y. Bengio, J. Chai, M. Lapata, A. Lazaridou, J. May, A. Nisnevich, et al. Experience grounds language. *arXiv preprint arXiv:2004.10151*, 2020.

C. Bonial, J. Hwang, J. Bonn, K. Conger, O. Babko-Malaya, and M. Palmer. English propbank annotation guidelines. *Center for Computational Language and Education Research Institute of Cognitive Science University of Colorado at Boulder*, 48, 2012.

S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning. The snli corpus. 2015.

E. Brill and R. J. Mooney. An overview of empirical natural language processing. *AI magazine*, 18(4):13–13, 1997.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

S. Buchholz and E. Marsi. Conll-x shared task on multilingual dependency parsing. In *Proceedings of the tenth conference on computational natural language learning (CoNLL-X)*, pages 149–164, 2006.

H. Bussmann. *Routledge dictionary of language and linguistics*. Routledge, 2006.

E. Cambria, D. Das, S. Bandyopadhyay, and A. Feraco. Affective computing and sentiment analysis. In *A practical guide to sentiment analysis*, pages 1–10. Springer, 2017.

I. Caswell, J. Kreutzer, L. Wang, A. Wahab, D. van Esch, N. Ulzii-Orshikh, A. Tapo, N. Subramani, A. Sokolov, C. Sikasote, M. Setyawan, S. Sarin, S. Samb, B. Sagot, C. Rivera, A. Rios, I. Papadimitriou, S. Osei, P. J. O. Suárez, I. Orife, K. Ogueji, R. A. Niyongabo, T. Q. Nguyen, M. Müller, A. Müller, S. H. Muhammad, N. Muhammad, A. Mnyakeni, J. Mirzakhalov, T. Matangira, C. Leong, N. Lawson, S. Kudugunta, Y. Jernite, M. Jenny, O. Firat, B. F. P. Dossou, S. Dlamini, N. de Silva, S. Çabuk Ballı, S. Biderman, A. Battisti, A. Baruwa, A. Bapna, P. Baljekar, I. A. Azime, A. Awokoya, D. Ataman, O. Ahia, O. Ahia, S. Agrawal, and M. Adeyemi. Quality at a glance: An audit of web-crawled multilingual datasets, 2021.

I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos. Legal-bert: The muppets straight out of law school, 2020.

D. Chen and W.-t. Yih. Open-domain question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 34–37, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-tutorials.8. URL `https://www.aclweb.org/anthology/2020.acl-tutorials.8`.

K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

N. Chomsky. On cognitive structures and their development: A reply to piaget. *Language and Learning: The debate between Jean Piaget and Noam Chomsky*, 1980.

N. Chomsky. *Syntactic structures*. De Gruyter Mouton, 2009.

N. Chomsky. *Aspects of the Theory of Syntax*, volume 11. MIT press, 2014.

J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

K. Clark, U. Khandelwal, O. Levy, and C. D. Manning. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*, 2019.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(ARTICLE):2493–2537, 2011.

A. Conneau, G. Lample, R. Rinott, A. Williams, S. R. Bowman, H. Schwenk, and V. Stoyanov. Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*, 2018.

M.-C. De Marneffe, T. Dozat, N. Silveira, K. Haverinen, F. Ginter, J. Nivre, and C. D. Manning. Universal stanford dependencies: A cross-linguistic typology. In *LREC*, volume 14, pages 4585–4592, 2014.

F. De Saussure. *Cours de linguistique générale*, volume 1. Otto Harrassowitz Verlag, 1989.

W. de Vries, A. van Cranenburgh, A. Bisazza, T. Caselli, G. van Noord, and M. Nissim. Bertje: A dutch bert model. *arXiv preprint arXiv:1912.09582*, 2019.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Q. N. T. Do, A. Leeuwenberg, G. Heyman, and M. F. Moens. A flexible and easy-to-use semantic role labeling framework for different languages. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 161–165, 2018.

D. Dowty. Thematic proto-roles and argument selection. *language*, 67(3):547–619, 1991.

R. Dror, G. Baumer, S. Shlomov, and R. Reichart. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, 2018.

A. Ettinger. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48, 2020.

C. Fillmore. The case for case. 1967.

L. Floridi and M. Chiriatti. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30(4):681–694, 2020.

G. Furnas et al. Using latent semantic analysis to improve information retrieval. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 281–285. ACM Press, 1988.

K. Gerdes and S. Kahane. Word order in german: A formal dependency grammar using a topological hierarchy. In *Proceedings of the 39th annual meeting of the Association for Computational Linguistics*, pages 220–227, 2001.

D. Gildea and D. Jurafsky. Automatic labeling of semantic roles. *Computational linguistics*, 28(3):245–288, 2002.

A. Go, R. Bhayani, and L. Huang. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, 1(12):2009, 2009.

Y. Goldberg. Neural network methods for natural language processing. *Synthesis lectures on human language technologies*, 10(1):1–309, 2017.

T. Groß and T. Osborne. The dependency status of function words: Auxiliaries. In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*, pages 111–120, 2015.

J. Hajič, M. Ciaramita, R. Johansson, D. Kawahara, M. A. Martí, L. Màrquez, A. Meyers, J. Nivre, S. Padó, J. Štepánek, et al. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. 2009.

B. Hamp and H. Feldweg. Germanet-a lexical-semantic net for german. In *Automatic information extraction and building of lexical semantic resources for NLP applications*, 1997.

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

P. He, X. Liu, J. Gao, and W. Chen. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*, 2020.

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

J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8): 2554–2558, 1982.

J. Hutchins. The history of machine translation in a nutshell. *Retrieved December*, 20(2009):1–1, 2005.

R. S. Jackendoff. Semantic interpretation in generative grammar. 1972.

T. F. Jaeger. Redundancy and reduction: Speakers manage syntactic information density. *Cognitive psychology*, 61(1):23–62, 2010.

Y. Ji, Z. Zhou, H. Liu, and R. V. Davuluri. Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *bioRxiv*, 2020.

N. Jiang and M.-C. de Marneffe. Evaluating bert for natural language inference: A case study on the commitmentbank. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 6088–6093, 2019.

D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025, 2020.

M. Johnson. How the statistical revolution changes (computational) linguistics. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, pages 3–11, 2009.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

P. Koehn. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 388–395, 2004.

D. Kumawat and V. Jain. Pos tagging approaches: A comparison. *International Journal of Computer Applications*, 118(6), 2015.

J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

J. D. Lewis and J. L. Elman. Learnability and the statistical structure of language: Poverty of stimulus arguments revisited. In *Proceedings of the 26th annual Boston University conference on language development*, volume 1, pages 359–370. Citeseer, 2001.

P. Lewis, B. Oğuz, R. Rinott, S. Riedel, and H. Schwenk. Mlqa: Evaluating cross-lingual extractive question answering. *arXiv preprint arXiv:1910.07475*, 2019.

Z. Li, X. Ding, and T. Liu. Transbert: A three-stage pre-training technology for story-ending prediction. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 20(1):1–20, 2021.

E. D. Liddy. Natural language processing. 2001.

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

I. Loshchilov and F. Hutter. Decoupled weight decay regularization, 2019.

D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

Z. Lu, P. Du, and J.-Y. Nie. Vgcn-bert: augmenting bert with graph embedding for text classification. In *European Conference on Information Retrieval*, pages 369–382. Springer, 2020.

B. C. Lust. *Child language: Acquisition and growth*. Cambridge University Press, 2006.

B. MacCartney and C. D. Manning. *Natural language inference*. Citeseer, 2009.

C. D. Manning. Computational linguistics and deep learning. *Computational Linguistics*, 41(4):701–707, 2015.

L. Martin, B. Muller, P. J. O. Suárez, Y. Dupont, L. Romary, É. V. de la Clergerie, D. Seddah, and B. Sagot. Camembert: a tasty french language model. *arXiv preprint arXiv:1911.03894*, 2019.

K. R. McKeown. Paraphrasing using given and new information in a question-answer system. *Technical Reports (CIS)*, page 723, 1980.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

W. Morgan. Statistical hypothesis tests for nlp, 2005.

B. Myagmar, J. Li, and S. Kimura. Transferable high-level representations of bert for cross-domain sentiment classification. In *Proceedings on the International Conference on Artificial Intelligence (ICAI)*, pages 135–141. The Steering Committee of The World Congress in Computer Science, Computer . . . , 2019.

M. Palmer, D. Gildea, and P. Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106, 2005.

M. Palmer, D. Gildea, and N. Xue. Semantic role labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103, 2010.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

Y. Peng, S. Yan, and Z. Lu. Transfer learning in biomedical natural language processing: an evaluation of bert and elmo on ten benchmarking datasets. *arXiv preprint arXiv:1906.05474*, 2019.

J. B. Plotkin and M. A. Nowak. Language evolution and information theory. *Journal of Theoretical Biology*, 205(1):147–159, 2000.

M. Polignano, P. Basile, M. De Gemmis, G. Semeraro, and V. Basile. Alberto: Italian bert language understanding model for nlp challenging tasks based on tweets. In *6th Italian Conference on Computational Linguistics, CLiC-it 2019*, volume 2481, pages 1–6. CEUR, 2019.

A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding with unsupervised learning. 2018.

P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

L. A. Ramshaw and M. P. Marcus. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer, 1999.

A. Rogers, O. Kovaleva, and A. Rumshisky. A primer in bertology: What we know about how bert works, 2020.

J. R. Saffran, R. N. Aslin, and E. L. Newport. Statistical learning by 8-month-old infants. *Science*, 274(5294):1926–1928, 1996.

M. Sahlgren and F. Carlsson. The singleton fallacy: Why current critiques of language models miss the point. *arXiv preprint arXiv:2102.04310*, 2021.

T. Samardzic. *Dynamics, causation, duration in the predicate-argument structure of verbs: a computational approach based on parallel corpora.* PhD thesis, University of Geneva, 2013.

G. Sampson. Writing systems. *London, UK: H utchinson*, 1985.

M. Sänger, U. Leser, S. Kemmerer, P. Adolphs, and R. Klinger. Scare—the sentiment corpus of app reviews with fine-grained annotations in german. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1114–1121, 2016.

V. Sanh, L. Debut, J. Chaumond, and T. Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

K. R. Scherer and H. G. Wallbott. Evidence for universality and cultural variation of differential emotion response patterning. *Journal of personality and social psychology*, 66(2):310, 1994.

A. Schiller, S. Teufel, C. Stöckert, and C. Thielen. Guidelines für das tagging deutscher textcorpora. *University of Stuttgart/University of Tübingen*, 1999.

R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni. Green ai. *arXiv preprint arXiv:1907.10597*, 2019.

R. Sennrich, G. Schneider, M. Volk, and M. Warin. A new hybrid dependency parser for german. *Proceedings of the German Society for Computational Linguistics and Language Technology*, pages 115–124, 2009.

R. Sennrich, M. Volk, and G. Schneider. Exploiting synergies between open resources for german dependency parsing, pos-tagging, and morphological analysis. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, pages 601–609, 2013.

O. Sharir, B. Peleg, and Y. Shoham. The cost of training nlp models: A concise overview. *arXiv preprint arXiv:2004.08900*, 2020.

R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

C. Sun, A. Myers, C. Vondrick, K. Murphy, and C. Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7464–7473, 2019a.

Y. Sun, S. Wang, Y. Li, S. Feng, X. Chen, H. Zhang, X. Tian, D. Zhu, H. Tian, and H. Wu. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*, 2019b.

M. Tomasello and M. Carpenter. Shared intentionality. *Developmental science*, 10 (1):121–125, 2007.

E. Troiano, S. Padó, and R. Klinger. Crowdsourcing and validating event-focused emotion corpora for german and english. *arXiv preprint arXiv:1905.13618*, 2019.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

J. Welbl, N. F. Liu, and M. Gardner. Crowdsourcing multiple choice science questions, 2017.

A. Williams, N. Nangia, and S. R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, et al. Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Y. Yang, Y. Zhang, C. Tar, and J. Baldridge. Paws-x: A cross-lingual adversarial dataset for paraphrase identification. *arXiv preprint arXiv:1908.11828*, 2019.

A. Yeh. More accurate tests for the statistical significance of result differences. *arXiv preprint cs/0008005*, 2000.

Y. Zhang, J. Baldridge, and L. He. Paws: Paraphrase adversaries from word scrambling. *arXiv preprint arXiv:1904.01130*, 2019a.

Z. Zhang, Y. Wu, H. Zhao, Z. Li, S. Zhang, X. Zhou, and X. Zhou. Semantics-aware bert for language understanding. *arXiv preprint arXiv:1909.02209*, 2019b.

Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.