Chair of Data Science

Extracting Definition and Subsumption from German Law

Master-Thesis of

Stefanie Urchs

1. Supervisor
Prof. Dr. Michael Granitzer

2. Supervisor
Prof. Dr. Harald Kosch

Advisor: Dr. Jelena Mitrović

July 6, 2020

Contents

Abstract

German legal argumentation is bound by writing styles. The two main styles *Urteilsstil* and *Gutachtenstil* share the components definition and subsumption. A classification of these components in law texts can be useful for the law education.

In 2009 Mochales Palau and Moens [PM09] used decisions of the European Court of Human Rights to detect argumentative text, classify the argumentative propositions contained in the argumentative parts and detect the relations between the arguments. The detection of argumentative text and the classification of premise and claim from [PM09] is adapted to detecting text written in *Urteilsstil* and classifying definitions and subsumptions. A model trained with this approach is subsequently used to detect subsumptions and definitions in *Gutachtenstil*.

Furthermore, this thesis presents two novel corpora of German law. The first corpus contains 31,334 court decisions and corresponding meta data. The second one consists of 200 annotated judgments and is used for the classification task.

Unfortunately, a reproduction of [PM09] was not possible due to changes in the corpus, the features and the implementation of the classifier. However, a simple support vector machine trained with tf-idf vectors leads to promising results on the judgment corpus. This simple model is not usable for a domain transfer from *Urteilsstil* to *Gutachtenstil*.

Acknowledgments

I would like to thank my thesis advisor Dr. Jelena Mitrović and my supervisors Prof. Dr. Michael Granitzer and Prof. Dr. Harald Kosch for the support and supervision of my thesis.

Furthermore, I would like to thank Prof. Dr. Urs Kramer for providing extremely valuable legal feedback and for providing exercise cases and the corresponding solutions.

Additionally, I would like to thank the Forschungsstelle "FREDI" and Prof. Dr. Kai von Lewinski for enabling the annotation of the judgment corpus. This thesis would not have been possible without their support.

Finally, I would also like to thank Dr. Richard Rotermund for his continuous support and help throughout the whole thesis process and his valuable legal feedback.

List of Figures

List of Tables

1 Introduction

Germany is a country of poets and thinkers; legal professionals take this to heart, as their language is their main professional tool. However, official legal texts like court decisions cannot be written as pure prose, the writing must be regulated. One way of legal text regulation are the writing styles used in legal texts. In Germany two main writing styles exit: *Urteilsstil* (judgement style) and *Gutachtenstil* (appraisal style) The first style consists of roughly three components: conclusion, definition and subsumption [Dan05] and is used in legal judgements. The second style contains roughly the same parts, however, the conclusion is written in a fundamentally different style [Web18]. An automatic detection of the components of these styles can be used for legal teaching. Detecting these components is no easy task. The components can be nested and sometimes omitted or combined with other components if deemed too trivial. To handle these problems argumentation mining techniques can be explored. In 2009 Mochales Palau and Moens [PM09] used decisions of the European Court of Human Rights to detect argumentative text, classify the argumentative propositions contained in the argumentative parts and detect the relations between the arguments. The detection of argumentative text and the following classification of propositions forms the baseline for the detection of definitions and subsumptions in judgements. A model trained on *Urteilsstil* should subsequently be able to detect definitions and subsumption in *Gutachtenstil* as these components should have the same form in both writing styles.

No corpora for this kind of detection are publicly available. Therefore, this thesis presents a novel corpus for detecting definitions and subsumptions in German legal judgements. Additionally, a bigger corpus with a huge variety of German court decisions and the corresponding meta data is introduced.

A position paper about the topic of this thesis was accepted on the 10th International Conference on Advanced Computer Information Technologies (ACIT 2020). The paper

"Towards Classifying Parts of German Legal Writing Styles in German Legal Judgments" [UMG] will be published in the IEEEXplore digital library after the conference in September 2020.

1.1 Objective

This thesis answers the following research questions:

1. Is it possible to automatically recognise the definition part of the *Urteilsstil* in a legal decision, using argumentation mining techniques?
2. Is it possible to automatically recognise the subsumption part of the *Urteilsstil* in a legal decision, using argumentation mining techniques?
3. Is it possible to train a model on legal decisions in *Urteilsstil* and use it on exercise case solutions in *Gutachtenstil*?

To answer these questions a novel annotated corpus of German legal judgements is created. Using this corpus a model is trained on judgements on *Urteilsstil*, answering question one and two. Two answer question three the trained model is used on an annotated corpus of exercise case solutions, written in *Gutachtenstil*. Furthermore, an not annotated corpus, enriched with metadata is presented.

This thesis uses the approach and results of [PM09] as baseline.

1.2 Structure of the Thesis

The remaining parts of this thesis are structured as follows. Chapter 2 introduces international work of computer scientists on legal texts. In chapter 3 the basics of argumentation and argumentation mining are discussed, leading to a presentation of written legal argumentation in chapter 4. Subsequently, the corpora of this thesis are presented in chapter 5, followed by the description of the learning on legal decisions. The model developed in chapter 6 is used on exercise cases in chapter 7. Finally the thesis is concluded in chapter 8.

2 Legal Tech

Kind et al. [KFP19] define legal tech as every technical solution that is used by law professionals. This definition is really broad, though, the field is too. It is not possible to explore the whole of legal tech in one single thesis chapter. Therefore, his chapter focuses on legal tech that is working with text.

The presentation of the field starts with legal information retrieval, followed by legal reasoning systems. Subsequently work in the field of legal text classification and legal text corpora are presented.

2.1 Legal Information Retrieval

Classical information retrieval (IR) relies on external knowledge (e. g. thesauri and classification schemes) and an accurate indexing of documents, that is mostly manually done. To reduce the manual labour and increase the accuracy of the retrieval methods the following fields should be explored: learning of the information needs, information extraction, text classification / clustering and text summarisation [Moe01].

Especially the legal information retrieval is blessed and cursed with plenty text-based literature about law. Not only the raw materials about law are text-based, also almost everything produced in law leads to new text. Already in the 1960s companies like *Lexis* introduced systems to retrieve legal information. The introduction of the internet triggered the development of a variety of new IR solutions [Wid02].

Schweighofer and Geist [SG07] introduce one of these new solutions. The authors combine legal ontologies and search context to perform query expansions for legal IR. In later work Geist [Gei08] includes citation analysis to further improve legal IR.

Maxwell and Schafer [MS08] identify two approaches to do IR. Firstly, a manual knowledge engineering (KE) approach, like manual document indexing mentioned in [Moe01] or systems to retrieve legal information mentioned in [Wid02]. According to the authors the KE approach is based on artificial intelligence and case-based reasoning. Secondly, they identify a natural language processing (NLP) approach which is linked to open domain statistical retrieval. The authors argue that the KE based approach was suitable in the past. However, in the future the KE based approach should be replaced by a NLP based approach. Furthermore, the authors propose a recall in the context of precision instead of the overall recall to measure the performance of an IR system. By using this contextual recall low quality opinions can be excluded from the IR results, saving the valuable time of law professionals. In a different paper Maxwell et al. [MOC13] advocate for a feature-based selection of dependency paths in IR. Dependency parse trees are able to capture syntactic and semantic relations between words. However, parsing large collections of documents is computationally expensive. To solve this challenge the authors use *catenae* which detect prominent semantic content and present it in an underspecified manner. These *catenae* should identify the most important information in a text.

Landthaler et al. [Lan+16] explore possibilities to go beyond pure exact matches in legal IR. By utilising word embeddings the authors enable IR systems to also find similar matches to the initial search term. Researchers of the same group apply rule-based text analysis methods to extract semantic information from legal documents [Wal+17a]. In addition to the semantic information the authors extract legal definitions from laws and how terms are used in a defining context. In a later paper Waltl et al. [WBM], who belong to the same research group as the authors of the last two papers, argue that rule-based IR is superior to statistical, machine learning based approaches, thus, disagreeing with [MS08]. The authors explain, that most industry solutions are based on rule-based IR, only academic solutions are based on machine learning. Using domain specific languages for pattern annotation on textual data should help to overcome the shortcomings of rule-based IR.

2.2 Legal Reasoning Systems

Already in 1987 Rissland and Ashley [RA87] presented a first legal reasoning tool called HYPO that is able to compare and contrast legal cases. The system is based on cases from American trade secret law which are organised in a case knowledge base. The cases are indexed in so called *dimensions* to improve case retrieval. The retrieval is trigged after a analysis of the *current fact situation*, and retrieved cases are organised in a *claim lattice* according to the *current fact situation*. In the same year Bench-Capon [Ben87] published a paper about a system that provides large legislation based organisations in the UK with knowledge based decision support. The system models the knowledge of policy makers in a knowledge base by defining (logical) rules between attributes. Using these rule policy makers can examine the knowledge base using the following facilities: assert, examine, why, what and which.

Based on HYPO Rissland and Skalak developed CABARET [RS91] in 1991, a reasoning tool which performs *statutory* interpretation tasks and interpret the meaning of a legal rule. This process is quite difficult, because critical statutory terms are not well defined in the regulation or statute. Therefore, knowledge linked to these, like cases or historical background, has to be found and interpreted. For this system the case-based reasoning of HYPO is extended with a rule-based reasoning approach. Ashley used HYPO in the same year, independently from Rissland, to develop the law learning environment CATO [AA91]. This system supports students to make arguments with cases by teaching them to test their theories and argument with given cases. The system generates arguments for each side of a claim by altering existing cases. It works with 26 *Factors* that are either present in a case or not and represent stereotypical strength and weaknesses of a case. These Factors are organised in a *Factor-Hierarchy* that represents legal knowledge of a higher level that helps to organise arguments and to reason with similar cases. Also in 1991 Branting adapted the case-based reasoning, presented in the HYPO paper, and developed the GREBE [Bra91] system. GREBE uses portions of precedents to analyse legal documents in the domain of Texas worker's compensation law. He focuses on determining the relevant similarities and differences between analysed cases. In the same year researches from the Netherlands presented the findings of their PROLEXIS [Wal+91] project, a toolkit for building a legal expert system and a prototype based on Dutch landlord-tenant law.

Australian researchers extended CABARET and PROLEXIS in 1993 to the IKBALS [ZVH93] system. The authors combine case-based reasoning, rule-based reasoning and case based retrieval. The system induces rules from cases and uses these rules as indices for the case based retrieval. Also in 1993, but in a different legal system, researchers from Japan introduced HELIC-11 [Oht+93], a legal reasoning system. It draws conclusions by combining statutory law and judicial precedents. All possible conclusions are drawn and presented in a inference tree.

In 1995 Zeleznikow and Stranieri improve the rule-based approach used in IKBALS, by introducing Split-Up [ZS95]. A system that predicts the outcome of property disputes in the domain of Australian family law by utilising argument structures and neural networks.

Buidling on HYPO and CABARET Rissland et al. develop BankXX [RSF96] in 1996. A system to peruse and gather information for arguments in a heuristic *best-first* fashion. BankXX relies on a huge heterogeneous and highly interconnected network of domain knowledge. The heart of the system is a semantic network, the nodes of this network consist of cases and legal theories and the links between the nodes are interconnections of cases and theories. On this graph the *heuristic best-first search* is performed. A year later the same research group presented SPIRE [DR97], a system to find passages with relevant information in legal cases. It operates in two stages. Firstly a small amount of relevant cases are extracted, then the similarity of these cases to the new problem is determined and the cases are arranged in a standard claim lattice. The best cases from the knowledge base form the so called *relevance feedback case-knowledge-base*. With help of this *relevance feedback case-knowledge-base* similar cases are extracted. Secondly the system extracts the relevant passages from the extracted cases.

Loui et al. presented in 1997 Room 5 [Lou+97] a web-based legal argumentation game. In this game users either argue pro-petitioner or pro-respondent in disputes taken from the US Supreme Court cases. By playing this game users construct an ontology for US federal law and a data base of semi-structured arguments.

Based on CATO Brüninghaus and Ashley developed SMILE [BA99] in 1999, a system that is able to find abstract fact situations in legal texts with a classification approach. SMILE works with a ID3 (decision tree) algorithm to classify to which *factor* a sentence in a legal text belongs. Over years the system evolved to use a nearest neighbour approach for text classification [AB09].

Greek researchers worked in the 2014 project NOMAD [Kio+14] in a similar direction as Bench-Capon in 1987. They present a system to help policy formulate and validate policies by utilising non-moderated crowd sourcing. The system gives policy makers insights into the perspective of citizens on different policies by analysing contributions in forums, social networks, clogs, wikis and newsgroups.

A German approach to legal reasoning systems is the 2016 presented LEXIA [Wal+16] by Waltl et al.. The authors develop a data science environment for the German legal domain. The data model of the system is able to represent German legal texts (e.g. documents from legislation and jurisdiction, contracts and patents) and their nested referential structure. As a base line for the architecture of their text mining engine Apache UIMA is used. The rule-based script language Apace Ruta enables the collaborative specification of linguistic and semantic structures.

2.3 Text Classification in the Legal Realm

As shown in chapter 2.2 the USA pioneered legal reasoning systems by utilising text classification. However, American researchers are also researching legal text classification outside of reasoning systems. Thompson [Tho01] uses a nearest neighbour algorithm, a decision tree and a machine learning rule induction algorithm to categorise case law cases into topics. A more recent work predicts the behaviour of the Supreme Court of the Unites States [KBB17]. The authors train a time-evolving random forest classifier to predict how the court will vote on cases.

In Australia researchers classify text in Australian contracts [CM10] with a variety of classifiers. Random forests are best suited to detect structural elements and meta data. Contracts are also of interest for Indian researchers. Indukuri and Krishna [IK10] use a binary support vector machine (SVM) to extract clauses from e-contracts. These clauses are subsequently classified into payment related and non-payment related.

EU legislation is also subject to classification. Mencía and Fürnkranz [MF10] compare a multinomial Naïve Bayes with a multi-class multi-lable perceptron for classify legal documents of the European Union. Aletras et al. [Ale+16] chose decisions of the European Court of Human Rights to train a SVM to predict if an article of the convention of human rights is violated. In the UK financial regulations are used for legal text

classification. Asooja et al. [Aso+15] use a SVM in a one vs. rest strategy to classify anti money-laundering provision types. By using this strategy the authors are able to handle the multi-label classification problem. O'Neill et al. [Nei+17] use a deep learning approach on UK financial regulation to classify modality in sentences. In Portugal Gonçales and Quaresma [GQ03] use Portuguese juridical documents for a multi-label classification problem. They train a SVM and compare the results to a C4.5 decision tree and a Naïve Bayes algorithm. The SVM outperforms the Naïve Bayes and is faster in the model building than the decision tree. The authors extend this work with linguistic features in a later paper [GQ05]. Naïve Bayes and multi-class SVM are also used in Italy. Francesconi and Passerini [FP07] classify provisions in Italian legislative texts. To improve text classification Opsomer et al. [Ops+09] exploit properties of legal text. The authors train a linear SVM on Flemish, Belgian and European environmental and energy legislation to detect the subject of a legal document. The Dutch compare knowledge-based classification of Dutch legal texts with a machine learning based approach. The authors use a SVM for the machine learning approach.

In Germany Waltl et al. [Wal+17c] predict the outcome of cases in German tax law. They train a Naïve Bayes on 44,285 judgements of German fiscal courts. Waltl et al. [Wal+17b] further use active machine learning to classify German legal norms. The work is based on the Dutch work presented above. The researchers revisit the legal norm in German law in a later paper to classify semantic types [Wal+19]. The authors compare how multinomial Naïve Bayes, logistic regression, SVM, random forests and multilayer perceptrons perform at classifying the legal norms in one of seven semantic types. Based on this work Glaser et al. [GSM18] experiment on the portability of machine learning models between different document types in the legal domain. They show that it is possible to train a model on tenancy law ot the German Civil Code and use it on a rental agreement data set.

Around the world researchers look at legal documents in different granularity and from different perspectives. However, it seems that no one focused on legal writing styles.

2.4 Legal Text Corpora

In 2005 Reed et al. [Ree06] presented the first corpus containing analysed argumentation. This corpus contains text from various domains, including judicial summaries and discussion. Due to a security breach this corpus is no longer online available. Researchers from the same group presented another legal corpus: a collection of 47 documents from the European Court of Human Rights [PM09; MM11], annotated with argumentation components and argumentation schemes. Unfortunately, they did not publish this corpus.

The US courts publish their cases online. Sugathadasa et al. [Sug+17] compiled 22,776 cases from the United States supreme court into a corpus and published it in combination with other legal cases. Judicial decisions from the UK form the basis of the corpus of Rizzo et al.[RP12]. This corpus was created and published, because most legal corpora are too small too act as normative reference or not public available. The authors cite the corpus of Favretti et al. [RTM01] as one of the few corpora comparable to their work. It contains European legislation in English and Italian. A corpus consisting Italian legislation in Italian and German is presented by Gamper [Gam00]. However, the paper does not mention a publication of this corpus.

Like the US courts European and especially German courts publish their cases free of charge for the public. These are some of the documents the internatinal research group computer assisted legal linguistics (CAL^2)[1] uses to compile legal corpora, containing legislation from around the world. Unfortunately, these corpora are not publicly available.

[1] `https://cal2.eu/index.php`

3 Argumentation Mining

Argumentation Mining is a relatively new field spanning between natural language processing, argumentation theory and information retrieval [PM09]. This section explores the basics of argumentation before moving to argumentative zoning, the ancestor of argumentation mining. In the end argumentation mining is discussed in detail.

3.1 Argumentation Theory

Argumentation exists in all areas of our lives. It is most evident in formal political debates, the exchange of scientific opinions or even in the legal environment. However, informal discussions in the family and at work are also full of argumentation. In general, argumentation can be defined as the representation of a standpoint, of a certain opinion, towards another opinion. It is not possible to make a relevant argumentation without having a standpoint of one's own. Furthermore, an argumentation without a real or imagined counterpart does not make sense. If all parties share the same opinion about a standpoint, there is no need to argue about it [Eem+14].

Argumentation is more than a structure, it is a *communicative act complex*, that consist of a combination of communicational moves. These moves can be verbal and also non-verbal (visual). Furthermore, argumentation is a *interactional act complex*, differentiating it from a pure monologue. It is part of a dialogue in which the person bringing fourth the argumentation tries to convince the other party of their standpoint, or make them at least accept it as a valid opinion. The arguing person can be *held accountable for* the constellations of propositions they are stating. These constellation include the propositions themselves and the communicative structure they have in the discussion. Lastly, argumentation tries to convince the opposite party of the acceptability of the standpoint by addressing them as *a rational judge who judges reasonably*. In

other words someone who is able to understand the argumentation and willing to do so [Eem+14].

These properties of argumentation lead to the following definition [Eem+14]:

> *"Argumentation is a communicative and interactional act complex aimed at resolving a difference of opinion with the addressee by putting forward a constellation of propositions the arguer can be held accountable for to make the standpoint at issue acceptable to a rational judge who judges reasonably."*

This subchapter introduces the fundamentals of argumentation theory. Starting with the logic foundation followed by three schools of argumentation and ending with a description of argumentation schemes.

3.1.1 Logic

The study of logic reaches back to the Greek philosophers. It defines the basic units of thinking as concepts and **propositions**. These units are strongly interwoven, propositions consist of concepts, but concepts on their own are meaningless [Fog55].

Propositions are the meaning of a sentence. Several sentences can have the same meaning, therefore, they have the same proposition. For example the sentences "Two and two is four." and "Zwei und Zwei ist Vier." convey the same meaning and have the same proposition. The basic structure of logic sentences is **subject(S) est predicate(P)**, four basic kinds of sentences are known to logic [Ajd58]:

1. All S are P. *(SaP)* → There is no S that is not P.
2. Some S are P. *(SiP)* → There are S that are P.
3. No S is P. *(SeP)* → There is no S that is P.
4. Some S are not P. *(SoP)* → There are S that are not P.

SaP contradicts *SoP* and *SeP* contradicts *SiP*. *SaP* excludes *SeP*, whereas *SiP* and *SoP* complement each other. Leading to the logical square depicted in figure 3.1 [Ajd58].

This logical square helps to identify if a sentence is just a negation of another one, leading to the same proposition, or a new and different sentence that has a new and different

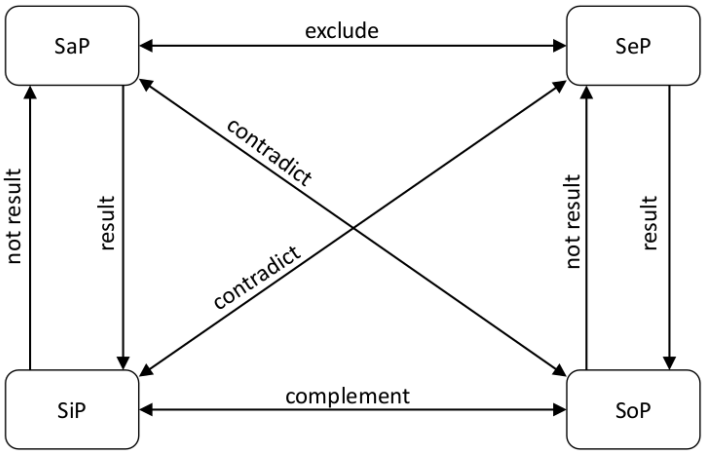


Figure 3.1: Logical square showing the relations between the four categorical logical sentences [Ajd58].

proposition. Furthermore, the logical square enables the reader to draw conclusions with just two sentences. For example: "All students studied for a test." is a sentence with the form *SaP*. From the logical square can be concluded that there can not be a *SeP* or *SoP*, leading to the following sentence "There is no student who did not studied for a test.". However, from "Some students studied for a test." (*SiP*) can be concluded that "Some students did not study for a test." (*SoP*). These conclusions, formed with only two sentences, are called immediate conclusion [Ajd58].

Besides the immediate conclusions, there are also indirect conclusions. The formal logic knows two kinds of conclusions, the inductive and the deductive conclusion. The inductive conclusion infers from a special case to a general principle, the deductive conclusion works the other way round. The best known type of deductive conclusion is the so-called **syllogism** [Ajd58]. The syllogism was introduced by Aristotle, it consists of two propositions that are concluded into a third one. Table 3.1 shows a simple syllogism [Fog55].

(1)	All fish live in water.
(2)	Sharks are fish.
(3)	Sharks live in water.

Table 3.1: Example of a syllogism.

The proposition (1) and (2) are called **premise** and (3) **conclusion**. The first premise is called major premise, the second minor premise [Fog55]. In argumentation theory, a set of premises together with a conclusion form an **argument**. Sometimes the conclusion is also called a **claim** [Kel13].

In normal use of language, a syllogism is not always used in its pure form. It occurs occasionally that one of the premises or even the conclusion is omitted, since these can be deduced from what has previously been stated. This kind of shortened syllogism is called enthymeme [Fog55]. Enthymemes pose a problem for automated argument recognition because they require a lot of domain knowledge.

3.1.2 Toulmin's Argumentation Theory

Toulmin defines **argumentation** as "*the activity of making **claims**, challenging them, **backing** them up by producing **reasons**, criticizing those reasons, **rebutting** those criticism, and so on*". He proceeds to define **reasoning** as "*central activity for of presenting the reasons in support of a claim, so as to show how those reasons succeed in giving strength to the claim*". In contrast to the logical definition of **argument**, Toulmin defines them as follows: "*sequence of interlinked claims and reasons that, between them, establish the content and force of the position for which a particular speaker is arguing*" [Tou84].

According to Toulmin every "*wholly explicit argument*" contains the following elements [Tou84]:

1. Claims
2. Grounds
3. Warrants
4. Backing

The **claim** of an argument is the destination of the argument. It is the standpoint that is discussed. The **grounds** are the foundation that support the claim. The **warrants** should support the grounds and reinforce the foundation of the claim. These warrants have to be checked and verified, this is done by the **backing** [Tou84].

In addition to the four elements of an argument Toulmin identifies **rebuttals** in argumentation. The claim of an argument can be presented as less certain using a rebuttal. By using a rebuttal exceptional circumstances that undermine the argument can be presented, thus anticipating the possible counter of an addressee [Tou84].

Figure 3.2 shows all elements of Toulmins argumentation theory in an example.

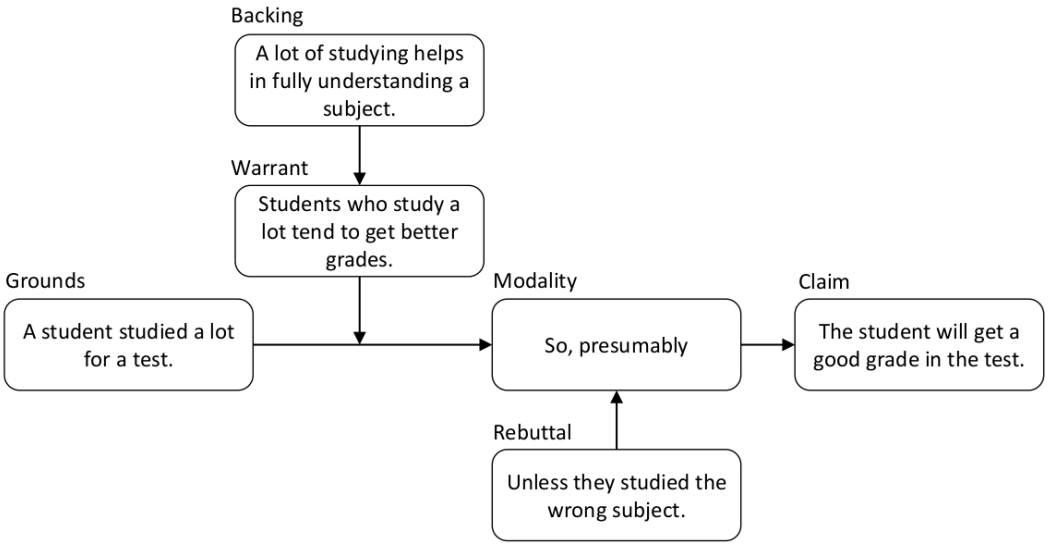


Figure 3.2: Graphical representation of Toulmins argumentation theory.

The main argumentation line is, that a student who studied a lot for a test will get a good grade in it. This main line is supported by the warrant that students who study a lot tend to get better grates. This warrant is backed by the fact that increased studying also increases understanding. However, using a modality one could bring forth the rebuttal, that the student could have studied the wrong subject, which would not help them to pass the test.

3.1.3 The New Rethoric

Perelman and Olbrechts-Tyteca define the *New Rhetoric* in their book with the same title [Ch 73] as an extension to logic. For the authors the main focus of argumentation is to convince the **auditorium**. To effectively convince an auditorium the arguer has to define their auditorium first and then tailor their argumentation to this specified auditorium. An argument that is convincing in regards to a certain auditorium can be totally irrelevant or even repulsive for a different one.

The authors differentiate between two classes of argumentation schemes: **association** and **dissociation**. Schemes that use association bring together distinct elements to a whole and form a unity among them. The schemes using dissociation do the exact opposite, they split up groups of unified elements.

Associative argumentation schemes are further divided into *"quasi-logical arguments"*, *"arguments based on the structure of reality"* and *"relations establishing the structure of reality"*. *"Quasi-logical arguments"* are similar to formal reasoning or mathematical logic. At first a general overview of the argument is given and subsequentially the overview is filled with data that supports the argument. Arguments that are based on the structure of reality also use a logical structure as basis. However, theses arguments bring fourth similarities between already accepted claims and new, not yet accepted, ones. The last category *"relations establishing the structure of reality"* uses argumentation by examples, including illustrations, analogies and metaphors.

3.1.4 Pragma-Dialectics

Building upon the theories of Toulmin and the New Rhetoric Eemeren and Grootendorst defined the pragma-dialectical theory of argumentation that combines pragmatic insights from speech theory and discourse analysis with dialectical insights from critical rationalism and formal dialectic approaches [Eem+14].

Eemeren and Grootendorst differentiate between argumentative discourse and discursive text. The former always involves two parties whereas the later only implicitly includes a second party and can, therefore, be considered as a monologue. According to the authors argumentation consists of **arguments** that relate to a **standpoint**. In the most basic version of argumentation one argument, containing one premise, relates to one standpoint, also called single argumentation. Complex argumentation consists of several interwoven single argumentations. The authors define three distinct types of complex argumentation: **multiple argumentation**, **coordinatively compound argumentation** and **subordinatively compound argumentation** [EG+92]. Figure 3.3 shows the three types of complex argumentation.

When using multiple argumentation an arguer brings forward at least two different arguments relating to their standpoint. In coordinatively compound argumentation at least two arguments are presented as a union to relate to the standpoint. The third kind of complex argumentation, subordinatively compound argumentation, is a chain of arguments, where the standpoint of the previous argument leads to the next argument. The chain of arguments is closed when the listener regards the sum of sub-standpoints as acceptable defence for the primary standpoint [EG+92].

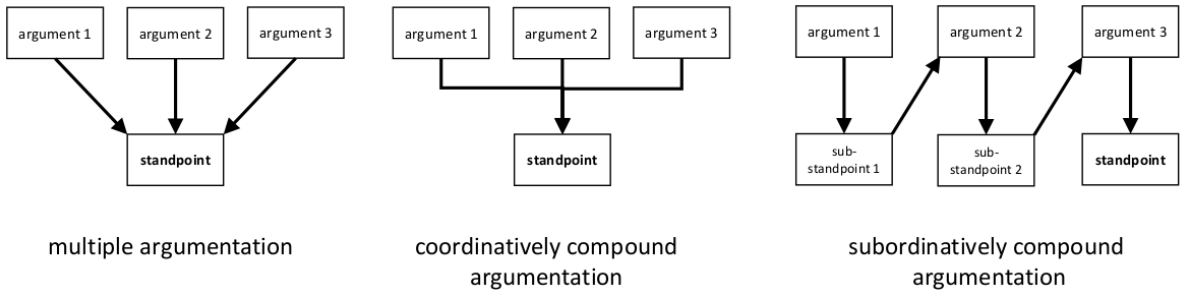


Figure 3.3: Graphical representation of complex argumentation types [EG+92].

3.1.5 Walton's Argumentation Schemes

According to Walton et al. [WRM08] **argumentation schemes** are particular patterns of reasoning. The classification of these patterns enables a probing for missing premises and a deterministic way to analyse and evaluate arguments. By pairing argumentation schemes with critical questions it is possible to evaluate an argument in the context of the dialogue in which it occurred. Only when all premises are supported by some weight of evidence the conclusion should be acceptable. The authors differentiate between schemes for *"argument from analogy, classification and precedent"*, schemes for *"knowledge-related, practical an other schemes"*, schemes for *"arguments from generally accepted opinions, commitment and character"* and *"casual argumentation schemes"*. The list of proposed schemes is not conclusive.

3.2 Argumentative Zoning

Simone Teufel [Teu99] presents Argumentative Zoning for generating user- and task-tailored summaries. By segmenting texts into rhetoric zones it is possible to identify specific parts of the text. E.g. scientific papers can be segmented in sections that cover the related work or scientific background, the aims of the paper, the organisation of the paper and many other zones. These zones can be used to summarise the text.

As a first step to automatically structure and categorise legal text Moens and Uyttendaele [MU97] use argumentative zoning. The authors segment legal cases into nine zones: superscription, identification of the victim, identification of the accused, alleged offences, transition formulation, opinion of the court, legal foundations, verdict and conclusion. Using these zones the authors represent Belgian criminal cases as knowledge graph. The

resulting graphs are used in the SALOMON [RA87] tool to automatically summarise the criminal cases. In a later work Hachey and Grover [HG05] use Argumentative Zoning to segment judgments of the UK House of Lords into facts, proceedings, background, framing, disposal, textual and other. These zones are subsequently used to summarise the judgments.

3.3 Argumentation Mining

The aim of Argumentation Mining is to automatically detect argumentation in free text. This includes the detection of all arguments that are part of the argumentation, the structure in these arguments (relation between premisses and claim) and the relations between these arguments [MM11].

In the following the tasks of argumentation mining are introduced, subsequently corpora for argumentation mining in general and on legal text are presented and in the end the current state of the art is discussed.

3.3.1 Tasks of Argumentation Mining

As outlined above argumentation mining starts with detecting argumentative text in free text. In this argumentative text the internal parts of an argument, namely premisses and claims, have to be classified. As soon as the arguments and their parts are known it is possible to detect the interactions between various arguments. Therefore the following three tasks of argumentation mining can be identified [PM09]:

1. Identification of arguments in free text.
2. Identification of the internal structure of an argument.
3. Identification of interaction between arguments.

The first task is essential for argumentation mining, because not all propositions in a text belong to an argument. For this classification very simple features like n-grams, part of speech, word couples, text statistics and keywords can be used. Furthermore, it is possible to look into the parse tree of a sentence [PM09].

In the second task boundaries of arguments are clarified by classifying the propositions for example into premises and claims. Features like the location of a sentence, the sentence length, tense or type of specific words and rhetorical or argumentative patters can be used. Depending on the underlying text specific features might be interesting too. For example in a law text a reference to an article of law might indicate a premise [PM09]. Instead of premises and claims it is possible to classify the four elements of an argument as defined by Toulmin.

Only when the boundaries between arguments are clear it is possible to detect the interaction between them. In other words to detect how arguments relate to each other. Furthermore, the utilised argumentation schemes could be detected. To perform this task a context-free grammar, based on the model of argumentative parsing for part of speech tagging, can be used [PM09].

3.3.2 Argumentation Mining Corpora

Argumentation Mining is a classification task and, therefore, a supervised machine learning task, which is in need of annotated corpora. In [Sta17; HEG14] the authors present an overview of currently available annotated argumentation mining corpora. To not simply repeat the work done by the authors only some corpora related to this thesis are presented.

AIFdb [Law+12] is a database solution for the so-called *Argument Web*. The *Argument Web* is a vision of an interconnected network of arguments posted on the internet. The website `http://corpora.aifdb.org/`[1] contains about 200 argumentation corpora with varying level of quality. Everyone can create an new corpus on the website. The most well known corpus on this website is AraucariaDB by Reed et al. [Ree+08]. The corpus was created in 2003 and contains texts from newspaper editorials, parliamentary records, judicial summaries and discussion boards from different regions of the world. All texts are written in English. After an extension in 2004 the corpus contains over 660 mapped arguments. The annotation includes the argument scheme and the relation between propositions.

[1] accessed on 2020-06-29

In 2009 Mochales and Ieven [MI09] presented a corpus from decisions of the European Court of Human Rights. The corpus consists of 45 judgments and decisions in English. The annotators were ask to annotate the argumentation structure according to the Pragma-Dialectics, argumentation schemes and premises & claims.

As mentioned in 2.4 the international research group computer assisted legal linguistics develops a German reference corpus (JuReKo) that contains "*all statutes of national law (legislation, recorded at one time); decisions and opinions of all federal courts and of a selection of courts at different instances (case law); commentaries, legal papers and articles of academic legal discourse, published in the most important and high ranked law journals*"[2]. Unfortunately, the corpus is not publicly available.

3.3.3 State of the Art in Legal Argumentation Mining

The line between legal reasoning system as described in 2.2 and argumentation mining gets thinner over time. In 2007 one of the first works dedicated to argumentation mining was published [Moe+07]. The authors use legal text to detect arguments. Based on this work Mochales Palau and Moens [PM09] detect, classify and structure arguments in legal texts. This work forms the baseline of this thesis.

Also in 2009 Gordon and Walton [GW09] detect argumentation schemes in legal text. The authors present *Carneade* a software architecture that enables users to construct and search for arguments by using *"diverse computational models of argumentation schemes"*.

ARGUMENTUM [Hou+13] is a German project, that developed a prototype for automatic detection, analysis as well as recommendation of argumentation structures in court decisions. The prototype is based on decisions of the German federal constitutional court. The prototype operates in six phases:

1. detecting argumentative text
2. separate text into smaller units
3. map paragraphs into a vector space

[2] `https://cal2.eu/core-projects-and-associated-projects/german-legal-reference-corpus-jureko-en` accessed on 2020-02-14

4. sentiment analysis for each paragraph
5. classification of propositions and relations between arguments
6. saving and structuring gathered information

No argumentation mining project is known to the author that focuses on the recognition of German legal writing styles.

4 Legal Argumentation

According to Alexy [Ale83] legal argumentation has two aspects: the internal justification and the external justification. The internal justification examines whether premises lead to claims in a logical way. The external justification checks the correctness of premisses.

This thesis discusses the internal justification of legal argumentation, also called the legal syllogism. Table 4.1 shows that the legal syllogism is formed like the classical/logical syllogism introduced in 3.1.1. From general norms (major premise) and the concrete problem (minor premise) a conclusion is logically deduced [Web18].

General norm/law	Major premise/ first premise	If rain is falling streets are wet.
Concrete problem	Minor premise/ second premise	Rain is falling.
Final sentence	Conclusion	Therefore, the street is wet.

Table 4.1: Example of legal deduction with a syllogistic inferring method [Web18].

In contrast to logical reasoning, legal reasoning requires a strict definition of concepts e.g. what is rain in a legal sense? Using these defined concepts the concrete facts are subsumed, leading to a conclusion [Web18].

The German legal education teaches law students to work on legal problems with two distinct styles. First the Gutachtenstil (appraisal style), which is used almost exclusively until the first state exam. After the first state exam the Urteilsstil (judgment style) is introduced.

4.1 Gutachtenstil

Appraisals written in *Gutachtenstil* do not, in contrast to common language, start with the solution to the question of the appraisal. It starts with a hypothesis and ends with the concluded result. Table 4.2 illustrates the *Gutachtenstil* [Web18]:

Hypothesis	*The weather might be nice.*
Definition	*The weather is always nice, if the sun is shining.*
Subsumption	*Today the sun is shining.*
Conclusion	*Therefore, the weather is nice.*

Table 4.2: Example of *Gutachtenstil* with named subcomponents [Web18].

Hypothesis: The hypothesis is formulated in a top sentence (the major premise). This major premise has to be in line with the problem at hand. Only if this is the case the examination can yield the result of the problem. Part of the hypothesis is to cite all norms/laws that are related to the hypothesis [Web18].

Definition: Subsequently the criteria that have to be examined are determined, one after another. The fulfilment conditions for this criteria are abstractly described. It is necessary to interpret the vague law term by using the law or common definitions [Web18].

Subsumption: At this stage the step from the abstract norm/law to the concrete problem is performed. It is examined whether the the problem suites the law or not. Therefore, the subsumption consults the described problem and compares it to the definition from above. It is assessed if the requirements of a norm/law are met or not. The reader of the appraisal has to be persuaded with a good argumentation. Therefore, all important information of the described problem have to be used. It is not enough to state a fact, also the types of violations, their severity and frequency have to be taken into account [Web18].

Conclusion: In the end a sentence concludes the result of the subsumption into a final result [Web18].

All criteria that have to be examined have to be defined and subsumed. In the end all parts are concluded together [Web18].

Definition, subsumption and conclusions can be combined in trivial cases [Web18].

Example:

V is reading a book, then O come suddenly from behind and shoots V in the head, killing them. Has O made themself punishable under §211 StGB?

Hypothesis: O may have made themself punishable according to §211 StGB.

Definition(0): A murderer under this provision is any person who kills a person for pleasure, for sexual gratification, out of greed or otherwise base motives, by stealth or cruelly or by means that pose a danger to the public or in order to facilitate or to cover up another offence.[1]

Definition(1): Stealth in the the sense of the law means intentionally exploiting the innocence and defencelessness of a victim.[2]

Definition(2): A victim is defenceless if their defence possibilities are restricted due to their innocence/being unsuspecting.[3]

Definition(3): Unsuspecting/innocent is anyone who, at the start of the attempted killing, does not expect any significant physical attack on their life or physical integrity.[4]

Subsumption(3): V is reading a book and can not expect to be harmed.

Conclusion(3): Consequently, V is unsuspecting/innocent.

Subsumption(2): V can not defend themself because they is innocently reading a book.

Conclusion(2): Because of that, V is defenceless.

Subsumption(1): O intentionally exploits the defencelessness of V who is innocently reading a book.

Conclusion(1): Therefore, O acts stealthy.

Subsumption(0): O kills V, who is a human, in stealth.

[1] `https://www.gesetze-im-internet.de/englisch_stgb/englisch_stgb.html#p1808` accessed on 2020-01-06
[2] BGHSt 9, 385 (390)
[3] Lackner/Kühl, 28. Auflage München 2014, § 211 Rdn. 8
[4] Rengier, StrafR BT II, 15. Auflage München 2014, § 4 Rdn. 24; Lackner/Kühl, 27. Auflage München 2011, § 211 Rdn. 7

Conclusion(0): Therefore, O is punishable by §211 StGB.

This is a highly artificial example, which is shortened for the sake of this thesis. In a real appraisal Definition 2 and 3 would be treated as one. The subsumption and conclusion of 2 would be condensed to one sentence, because of the triviality of this part.

4.2 Urteilsstil

In contrast to *Gutachtenstil*, *Urteilsstil* starts with the determination of the judgment, the reasoning follows. The reasoning consist of part-results. Another difference to *Gutachtenstil* is that the *Urteilsstil* does not consider all possible points of view and disputes. Only explanations that are relevant to the result are discussed [Dan05].

The most basic version of the *Urteilsstil* consists of three stages: the concrete legal consequences followed by the abstract legal facts and consequences (aka exact wording of the law) ending in the concrete facts. Between the abstract legal facts and consequences and the concrete facts a *Feststellungssatz* (determination sentence, the result of the subsumption) states if the legal requirements are fulfilled or not [Dan05].

The determination sentence and the concrete facts together form the subsumption.

This leads to the following structure:

Conclusion	Overall Result	The weather today is nice.
Definition	Abstract Legal Facts and Consequences	The weather is always nice, if the sun is shining.
Subsumption	Determination Sentence Concrete Facts	The conditions are met. The sun is shining.

Table 4.3: Example of *Urteilsstil* with named subcomponents and a mapping to *Gutachtenstil*.

Table 4.3 shows that the *Urteilsstil* and *Gutachtenstil* contain the same three writing style parts. However, the position of the conclusion is fundamentally different. Therefore, although these two conclusions are logically the same they differ on a semantic level.

Example [Dan05]:

Overall Result: *The claim is well justified. The defendant owes the plaintiff 1500€ in damages.*

Abstract Legal Consequences (definition): *If an object is damaged by an animal, then according to §833 S 1 BGB the person who keeps the animal is obliged to compensate the injured person for the resulting damage.*

Determination Sentence: *These conditions are fulfilled here.*

Concrete Facts: *The defendant's dog scratched the paint of the plaintiff's car. The repainting caused costs of 1500€.*

This most basic schema is mostly extended in practice. As in the *Gutachtenstil*, norms may refer to other regulations. This results in a nested structure.

5 Corpus

In 2009 Mochales and Ieven [MI09] created an argumentation corpus of 45 judgements and decisions of the European Court of Human Rights(ECHR), gathered in the time from August 2006 to December 2006. The authors decided to work on the English version of the documents instead of the French translation.

The chosen documents are structured into a header, that provides information about the case, summaries on the main facts of the case, previous decisions of the ECHR on the current case, previous steps of the applicant before they took the case to the ECHR and complaints, followed by the decision process of the court and finalised with the decision of the court. The ECHR is subject to case law, therefore, the discourse structure begins with the applicant making their case, by listing articles and facts on which the case is based and argues why the facts violate the listed articles. Subsequently, the defendant defends themself by offering their interpretation of the facts and how these relate to the listed laws. In the end the court analyses the provided complaints and arguments provided by both sides and offers their independent argumentation and decision.

The corpus is annotated by two sets of annotators each consisting of two persons and a "judge" to solve disagreements between annotators. The first set of annotators is composed of two lawyers who still studied in their masters degree, at the time of the annotation. One of the lawyers is European the other one is not from Europe. The annotators of the second set have no law background, but are trained on argumentation theories, argumentation analysis and the type of argumentation done in the ECHR corpus. The "judge" worked for more than four years on analysing ECHR cases and have a profound knowledge on argumentation theories, especially argumentation schemes.

The annotators task is to detect argumentation schemes, the components of the detected arguments and the relations between these components. Annotation of the corpus proved to be difficult. The annotators had problems on deciding which kind of argumentation to mark. Different understanding of law lead to disagreements in the first annotator

set. The boarders of arguments are hard to detect, because premises that lead to a final conclusion are scattered in the text. This made the premises hard to detect. In the end the authors conclude that using (well-known) argumentation theories for the annotation of a real-word law corpus leads to huge problems.

The presented corpus was reworked and used in [PM09], the paper forming the baseline for chapter 6.

The corpus presented above is comparable to the corpus needed in this thesis. However, this thesis deals with German law. This means on the one hand, that the used language is German and not English. On the other hand German law is civil/statute law. This different kind of law follows different argumentation guidelines and thinking principles. Case law argues with pre-existing cases, whereas statute law argues with the law written in statutes. Furthermore, in the presented corpus argumentation is annotated according to well-known argumentation schemes. The elementary units of argumentation are annotated as premise and claim. To detect German legal writing styles it is necessary to deviate from common argumentation theory. A corpus providing annotations for writing style parts is essential to train a classification model that can automatically detect these writing style parts. At the time of writing this thesis no publicly available corpus containing annotations of German legal writing styles exists. Therefore, a new corpus is presented in this thesis.

The remainder of this chapter is structured as follows: at first the data basis for the corpus is described, then the scraping and crawling of the data is discussed followed by a discussion about the annotation of the data. In the end the annotated corpus is presented.

5.1 Basis

Like [MI09] this thesis focuses on legal judgements. In Germany different sources offer German court decisions. For this thesis the court decisions of the Bavarian courts form the basis of the corpus. The website `www.gesetze-bayern.de`[1] offers 31,334 court decisions from 131 Bavarian courts, dating back to 2015 (as of 2020-01-24). The decisions are provided from the Bavarian state after the courts agreed to a publication. All decisions are processed by the publisher C.H.BECK, commissioned by the Bavarian state. This processing includes anonymisation, key-wording and adding of editorial guidelines to the decisions. Furthermore, the website contains laws, norms and additional law resources that are added to the website a few days after their official proclamation. In addition to the court decisions and other law resources the website provides the electronic version of the *Bayrisches Gesetz- und Verordnungsblatt* (Bavarian law and regulation journal) where newly issued laws and the most important regulations, of Bavarian State Parliament and the Bavarian Government, are published.

The decisions consist of 22 different types, mostly resolutions (*Beschluss*, 15,946 (51%)), judgments (*Urteil*, 11,530 (37%)) or end-judgments (*Endurteil*, 2,592 (8.2%)). From the 131 courts the most decisions are made by VG Munich (7839 (25%)), VGH Munich (6819 (22%)) and OLG Munich (2221 (7%)). The courts of the Bavarian capital are generally the busiest ones. The VGH Munich is the *Verwaltungsgerichtshof* (Higher Administrative Court) of Bavaria, the highest instance for all topics regarding the public administration. Below the VGH exist six VG (*Verwaltungsgericht*/ Administrative Court) almost all administrative regions have their own VG, except for lower Bavaria, which fall under the jurisdiction of VG Regensburg. These type of courts issue the most decisions in the corpus (21,243(68%)). Figure 5.1 shows that the courts issue more decisions in the first seven months of the year. The first three months are the busiest of the year.

In addition to the annotated corpus, that is presented in detail later on, a not annotated corpus containing all 31,334 decisions will be published on zenodo[2]. The decisions are enriched with metadata and should enable researchers to gain more insights into Bavarian courts.

[1] `https://www.gesetze-bayern.de/` accessed and crawled on 2020-01-24
[2] `https://zenodo.org/`

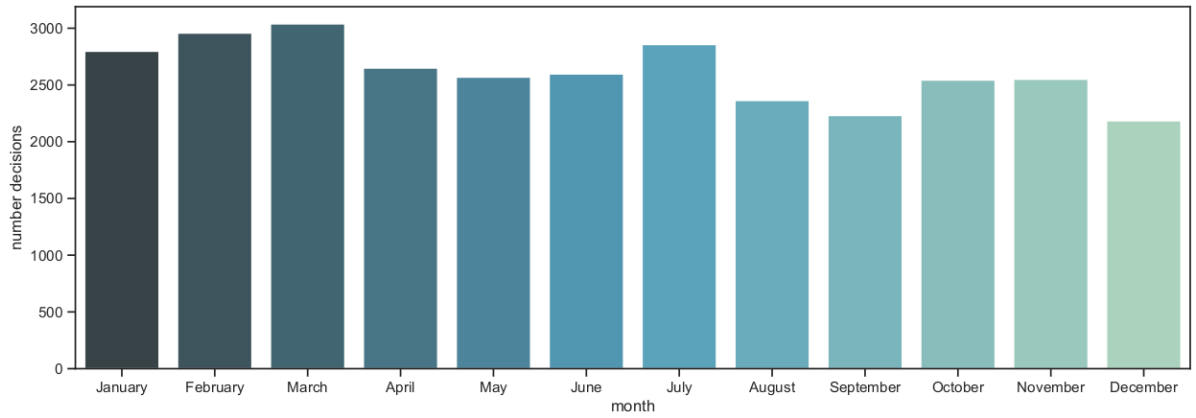


Figure 5.1: Number of issued decisions over all courts by month.

5.2 Crawling/Scraping

The website `www.gesetze-bayern.de` is crawled for legal documents using the python framework scrapy[3]. Navigating websites and scraping huge amounts of data is easy due to the build in functions of scrapy. Legal judgements on this website have the following structure:

- document title (figure B.1)
- box with metadata containing (figure B.2) (not all metadata is always present in all decisions):
  - title of the judgment
  - chains of norms
    List of norms that are important for this judgment.
  - guiding principles
    Numbered list of the guiding principles of the judgment.
  - keywords
  - lower court
  - additional facts
  - decision reference
    Reference to where this judgment is officially published.

[3] `https://scrapy.org/`

- *Tenor* (purport) (figure B.3)
  Most important results of the legal judgment numbed with Roman numerals or Arabic numbers.

- Offence (figure B.4)
  Detailed description of the offence with sometimes numbered paragraphs. The website also numbers the paragraphs, but separates paragraphs differently than the original numbering.

- Grounds for the decision (figure B.5, B.6, B.7, B.8, B.9)
  Detailed description of the grounds for the decision. The website numbers paragraphs. Additionally, a variety of structuring formats exists.

In contrast to other decision types only judgements always contain the text part *Tenor* and Offence.

This complex document structure is further complicated by an even more complex HTML structure of the website. To attain fast results the website is crawled two times. At the first time a simple data set is extracted that only contains the name of the document, the name of the legal judgment and the plain text of the grounds for the decision. In the second scraping iteration as much metadata as possible is extracted. This includes the explicit metadata listed above and the implicit metadata conveyed in the paragraph structure.

Scrapy utilises so called *spiders* that go through websites and scrape the contents. The spider for the judgments starts at the site `https://www.gesetze-bayern.de/Search/Filter/DOKTYP/rspr`. This site is the first page of the list of all available judgments. On this page the spider is searching for all `div` that have the class `hltitel` and follows the links contained in this `div`s, if the item contains "*Urteil*". The spider parses the page that lies behind the link. First the name of the judgment is extracted, then the title of the document. Sometimes the title contains a "/". This is substituted with a "$" so the document can be saved under its title. Finally the most important part is extracted: the text of the decision reasons. To do so the spider searches for a `h2` that contains the character "*ründe*". Inside of the "*ründe*" only the `div` with the class `absatz gruende` are scraped. This excludes the paragraph numbers provided by the website. The result is a list of all paragraphs of the "*Entscheidungsgründe*". This list is joined to a string using new lines. The parts are combined into a JSON and saved into a directory. After scraping all links of one site the spider navigates to the next site by selecting the last

item of the pagination on the page. The full listing for the judgment data set can be found in listing A.1.

The full data set spider also starts at the site `https://www.gesetze-bayern.de/Search/Filter/DOKTYP/rspr`. In contrast to the judgment spider the full spider visits every link on its way. After it followed a link the page is parsed. At first the *meta title* is extracted it consists of the court, the decision style, the decision date and the file number. The full *meta title* is used for saving the file, "/" are exchanged for "$" for saving purposes. Second, the metadata from the `rsprbox` is parsed. This box contains additional information provided by the website: norm chains, decision guideline, keywords, lower court(s), additional information and a decision reference. Decision guidelines have a special `div` class, which can be addressed for scraping. Keywords, additional information and decision references are stored in a single `div`. Therefore, it is possible to search for a `div` with the class `rsprboxueber`, in which all headlines are stored, and take following `rsprboxzeile`. Norm chains and lower courts can consist of several `rsprboxzeile`. Therefore, all `rsprboxzeile` between the `rsprboxueber` containing *"menkette"* or *"instan"* and the possible following `rsprboxueber` are scraped. Thirdly, the decision content is parsed. Purport, legal facts and decision reasons can be addressed via their `div` class. To preserve the structure provided by the website, the content is saved in a list of strings. Each string corresponds to one `div` element in the website. See listing A.3 for the full listing of the full data set.

The scraping saves the decisions in the following JSON format:

```
{
   "meta": {
      "meta_title": "",
      "court": "",
      "decision_style": "",
      "date": "",
      "file_number":"",
      "title": "",
      "norm_chains": ["", ""],
      "decision_guidelines": ["", ""],
      "keywords": "",
      "lower_court": ["", ""],
      "additional_information": "",
      "decision_reference": ""},
   "decision_text":{
      "tenor": ["",""],
      "legal_facts": ["",""],
      "decision_reasons": ["",""]
   }
}
```

The structure of the metadata should be conserved as best as possible. Therefore, norm chains, decision guidelines and lower court are saved as a list of strings, conserving the `div` structure. Keywords, additional information and decision reference are saved in one `div` and, therefore, saved to one string. The purport, the legal facts and the decision reasons are saved into a list of stings, that referrers to the `div` structure of the website. In the annotated corpus the decision reasons are enriched with labels. Leading to the following form:

```
{
   "meta": {
   "meta_title": "",
   "court": "",
   "decision_style": "",
   "date": "",
   "file_number":"",
   "title": "",
   "norm_chains": ["", ""],
   "decision_guidelines": ["", ""],
   "keywords": "",
   "lower_court": ["", ""],
   "additional_information": "",
   "decision_reference": ""},
   },
   "decision_text":{
      "tenor": ["",""],
      "legal_facts": ["",""],
      "decision_reasons": [
         [
            [text, label],
            [text, label]
         ],
         [
            [text, label]
         ]
      ]
   }
}
```

The decision reasons are a list of paragraph lists. These paragraph lists contain a list for every sentence in this paragraph. The sentence list contains the text of the sentence on position 0 and the corresponding label on position 1. Three different labels exist: `definition`, `subsumption` and `other`. All all sentences in every paragraph are labelled with one of these labels.

5.3 Annotation

From the 11,530 scraped legal judgements 200 are randomly chosen for manual annotation. See listing A.2 for the selection process. The annotation is done by a legal expert, that holds a first legal state exam. Law students first make contact with the *Urteilsstil* used in legal judgements after the first state exam.

To create a gold standard corpus the annotation of several expert annotators, having a high inter annotator agreement is neccessary [Wis+14]. If non-expert annotators are utilised at least four annotators are needed to replace one expert annotator. Furthermore, a sufficiently high annotator agreement between these non-experts has to be reached. To reduce annotation time and costs the corpus for this thesis is annotated by the legal expert introduced above. The resulting corpus is not a gold-standard corpus but sufficient for this thesis.

To ensure a common understanding of the *Urteilsstil* between the legal expert and the thesis author, the expert is provided with an annotation-guide. Additionally, the annotation task is discussed in detail before the start of the annotation. The annotation guide and the detailed discussion also entailed an in-depth description of the used annotation tool.

5.3.1 Annotation Guide

The annotation guide starts with an explanation of the annotation tool. The annotator is guided through the tool and the basic functionalities that are relevant for the annotation are explained. Since the annotator works remotely a personal introduction into the tool is not possible. The system description in the guide enables the annotator to look up functionalities if they forgot how to use the tool.

The second part of the guide describes the definition of the *Urteilsstil* as outlined in background chapter 4.2. In addition to general style description the guide contains the following remarks:

- Nested definitions and subsumptions should be marked.
- Only full sentences should be marked, if a subsumption and a definition are contained in the same sentence, this sentence should not be marked.

- Only text the annotator is sure about should be marked.
- It is possible that a document does not contain any clearly recognisable definitions and/or subsumption.

See C for full a guide in German, as provided in pdf format to the annotator. In addition to the pdf version of the guide the annotation tool contains a HTML version of the annotation guide. The HTML version excludes the system description, it is focused on the description of the *Urteilsstil* and the remarks mentioned above. The HTML of the guide version is embedded into the annotation tool and easily available from every annotation task.

5.3.2 Annotation Tool

The annotation platform doccano[4] is used for the annotation task. This is an open source text annotation tool for human use. The fully dockerised tool is available on Github and can be deployed by simply cloning the repository. The developers also offer one-click deployment via Amazon Web Services, Microsoft Azure, Google Cloud Platform, Heroku.

doccano offers three kind of projects: text classification, sequence to sequence tasks and sequence labelling. In the first project type, the text classification users can label texts with one label for the whole text. These kind of projects are useful to develop corpora like 20 Newsgroups or Reuters News Dataset where documents are labelled with their news category. In sequence to sequence users can annotate text sequences with other text sequences. This kind of project is very useful for translation tasks. A text sequence in one language is labelled with a text sequence in another language. The third kind of project type, the sequence labelling enables users to mark sequences in text and put a predefined label on this sequence. Such a project opens up a lot of possibilities. It allows for, named entity labelling, part-of-speech tagging, semantic role labelling and also annotating parts of legal writing styles in legal text.

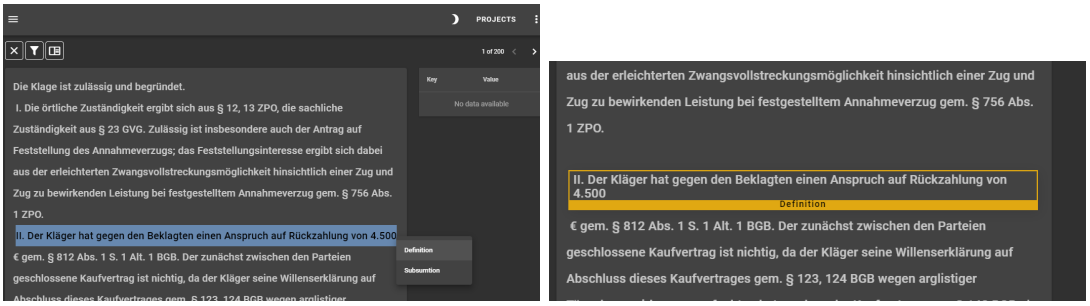
The system provides four user levels, an *active user*, *staff status user*, *superuser* and *inactive user*. *Active user* are normal user who are able to log into the annotation tool and perform assigned tasks. *Staff users* can additionally log into the administration

[4] `https://github.com/doccano/doccano`

site of doccano and create/edit/delete projects, users and labels if they get these rights assigned. Furthermore, the *staff user* can change user permissions, if allowed to do so. *Superusers* have all these permissions without explicitly getting them assigned. *Inactive users* are users who no longer have access to the system. It is advisable to not delete these users because the users are linked to the annotations they have done.

Upon creating a new project *staff users* or *superusers* decide on a project name, a short description, the project type and the users that are allowed to access the project. In addition annotation guidelines can be added in a simple HTML format. It is also possible to randomise the document order and to allow collaborative annotation. If collaborative annotation is enabled users can access the same document at the same time. However, they can only edit and delete their own annotations.

The annotation process is quite intuitive as shown in Figure 5.2. The user marks a text sequence by clicking the left mouse button and dragging the cursor over the desired text, like in any other program containing text. After releasing the left mouse button the user is prompted to chose from the predefined labels, they can also use a keyboerd shorcut to chose the label. After a label is chosen the text sequence gets surrounded by a box in the colour of the label. If the user the hovers the mouse over the annotation-box a "X" appears in the upper left corner of the box. By clicking this "X" a label can be deleted.



(a) Selection of a label in doccano (b) Annotated sequence in doccano.

Figure 5.2: Annotation process in doccano.

Staff users and *superusers* can also look into statistics about the annotation project. These statistics include how many documents are labelled, how many labels are issued per label and which user assigned how many labels.

Datasets can be uploaded either as plain text, JSON or CoNLL with a limit of 50MB per upload. The annotated data can be exported as JSON or as special JSON text label format.

Two well known tools in the text annotation realm are GATE[5] and brat[6]. GATE (general architecture for text engineering) offers a overwhelmingly huge variety of services for scientists, educators and businesses. It is an infrastructure that enables scientists and developers to develop and deploy software components that process human language. To be able to generate projects GATE has to be locally installed. Annotations can be shared via the web with a plug-in called CREOLE. However, no online annotation is possible. Data resources are stored in specific data stores and have to be loaded into the project. Due to the offered variety of possibilities GATE is hard to understand and to use. Furthermore, the user interface of GATE is scientific and loaded with information. This loaded user interface can not be used intuitively. The complexity of the tool, the requirement of local working and the unintuitive interface lead to a decision against GATE. A main requirement for the annotation tool is to be able to do annotation from everywhere, which requires a web platform. Additionally, the annotator is no (computer) scientist that is used to work with complex tools. The installation of the tool and the unintuitive user interface would add unnecessary difficulties to an already complex annotation task.

brat is also a well known tool in the scientific community. Unlike GATE, brat is a web-based tool. It is designed for structured annotations like marking named entities or relations between these entities. Documents can be loaded into the system without storing them in a specific data store. The labelling process is similar to doccano, the user marks a sequence and is subsequently prompted to chose a label. However, to provide a brat server the administrator has to first install brat and then a CGI-capable web server. This setup process is far more complicated than the dockerised deployment of doccano. Setting up projects is controlled in four different files, whereas doccano utilises an administrative website. The most important reason for not using brat is the user interface. brats user interface is not overloaded with functionality like the one of GATE. However, the user interface is simplistic in a scientific way. In other words the user interface of doccano is visually more pleasing.

In a 2002 article Norman [Nor02] stated that *"Attractive things work better"*, explaining that positive emotions positively enhance creative breadth-first thinking. People under stress, e.g. while doing a difficult and boring annotation task, tend to be less able to cope with difficulties and less flexible in their problem solving approach. However, positive

[5] `https://gate.ac.uk/teamware/`
[6] `https://brat.nlplab.org/index.html`

affect counters these problems and makes people more able to cope with difficulties makes them and more flexible. Aesthetic design leads to a more relaxed usage and, therefore, increases the usability of the tool. Due to this psychological phenomena it is crucial to choose a system with a pleasing user interface. doccano provides the most intuitive and visual pleasing interface of all three tools. In addition to this crucial advantage the easy out of the box deployment stirred the tool decision towards doccano.

5.4 Annotated Corpus

As stated above the corpus consist of 200 randomly chosen judgments, that are annotated by an legal expert. All sentences in the judgments are tagged with either `definition`, `subsumption` or `other`. After thesis completion the corpus will be published on zenodo[7].

Overall 25,198 sentences are annotated. 21% (5,331) of these sentences are marked as `definition`, 53% (13,386) are marked as `subsumptions` and the remaining 26% (6,481) as `other`. Figure 5.3 shows over all judgements how many sentences are labelled with a certain label and how many sentences judgments contain.

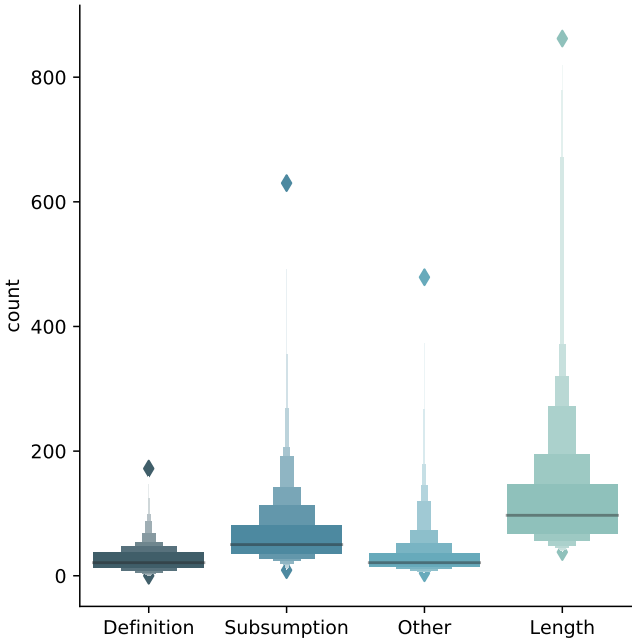


Figure 5.3: Number of sentences for all labels and all judgements.

[7] `https://zenodo.org/`

The length of judgments in sentences ranges from 38 to 862 sentences. In the median judgments have 97 sentences, most judgements are on the shorter side. Judgements are mostly written to explain the ruling of the court to the loosing side [Dan05]. People involved in court cases are for the most part no law experts. For that reason the judgements should be as brief as possible, as reflected by the corpus.

The amount of marked `definitions` is low in all documents. Furthermore, the length of a judgment only slightly increases the number of sentences marked as `definition` in a judgement. `Subsumptions` contrast this behaviour. The number of sentences labelled with `subsumption` correlates strongly with the number of sentences in a judgement. The median of sentences labelled with `definition` is 21 per judgement and for `subsumption` 50 per judgement. This shows that independent of the length of a judgement the description of the abstract law is short. Facts are stated without much detour. However, on average law professionals use more than twice as many sentences to subsume the case. When looking back to chapter 4.2 this seems reasonable. The subsumption is the place to argue and combine the pure listing of the law from the definition with the real world case. Law professionals can show their handiwork in this component. The remaining sentences are labelled as `other`. These sentences include the conclusions and text that does not belong to one of the writing style components. Additionally, the annotator is instructed to only label whole sentences as `definition` or `subsumption`. If one sentence contains both it is labelled as other. Therefore, it is hard to draw conclusions about the component `conclusion` from the sentences labelled as `other`. In general only a median of 21 sentences per judgement is labelled as `other`. Meaning that on average less sentences form conclusions than sentences form definitions. As conclusions are mostly short statements this observation seems reasonable. However, without a clear labelling of `conclusion` any statements about conclusions in German legal judgements are mere guessing.

Figure 5.4 shows that judgements from 22 of the 131 courts are selected for the corpus. Most judgements originate from the *VG Augsburg* (59 / 30%) followed by the *VG Ansbach* (39 / 20%) and *LSG Munich* (33 / 17%). For most court one to four judgements are selected. However, the analysis of the base data already reviled that *Verwaltungsgerichte* (VG) contribute more to the corpus than other courts. The *LSG Munich* is on place four for contributing judgements to the base corpus. Therefore, the random selection of judgements seems to reflect the underlying distribution of decision by courts well.

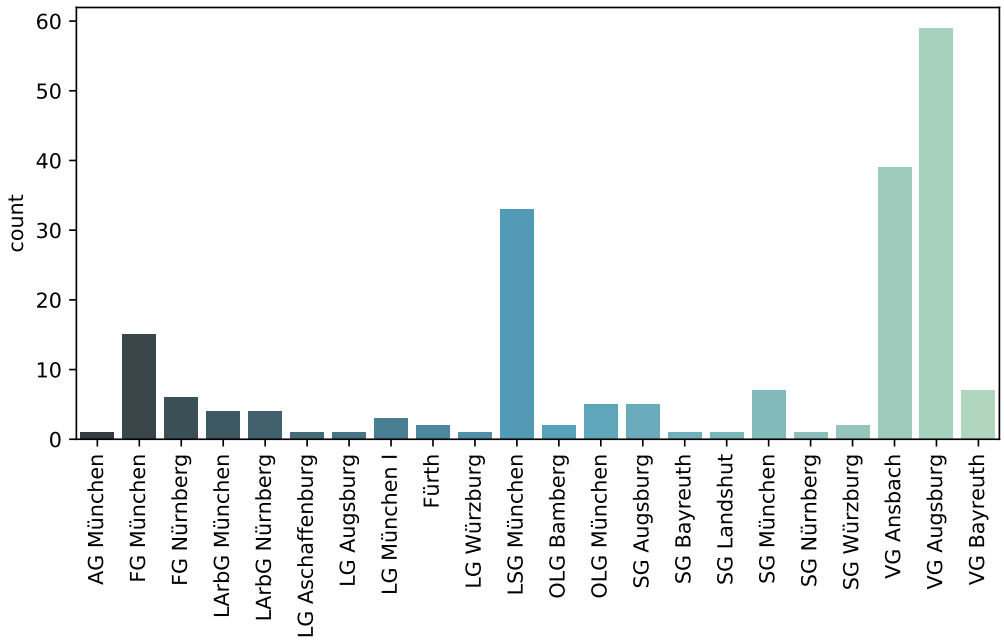


Figure 5.4: Distribution of judgements per court.

The distribution of judgments per month does not reflect the base data well, as shown in figure 5.1. This could result from the fact that there are many decisions in the base data that are no judgements.

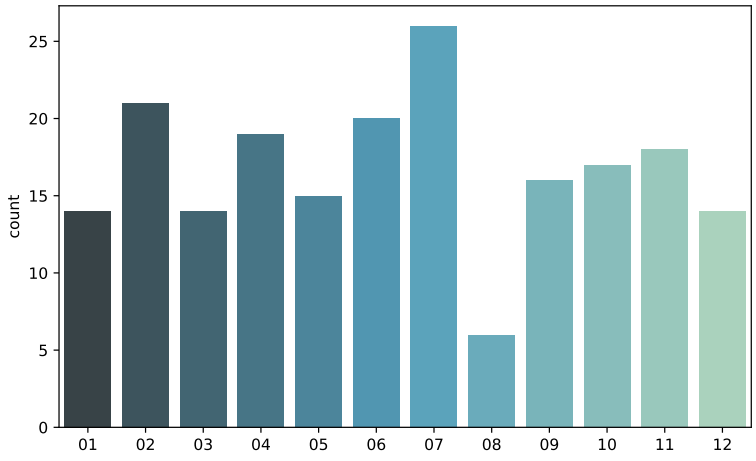


Figure 5.5: Distribution of judgements per month.

Figure 5.6 shows the distribution of the judgments per year. 29% (58) of all selected judgments were issued in the year 2016, followed by 22% (44) from the year 2017 and 21% (41) issued in the year 2015. This selection differentiates slightly from the base data where 24.5% (7,663) decisions are issued in 2015, 23.9% (7,487) in 2016 and 21.7% of the decisions are issued in 2017. The percentages of selected judgements and decisions issued in 2018 and 2019 are roughly the same. No judgments from 2020 are selected. However, decisions of 2020 form only 0.4% of the base data, which makes not selecting them a valid representation of the base data.

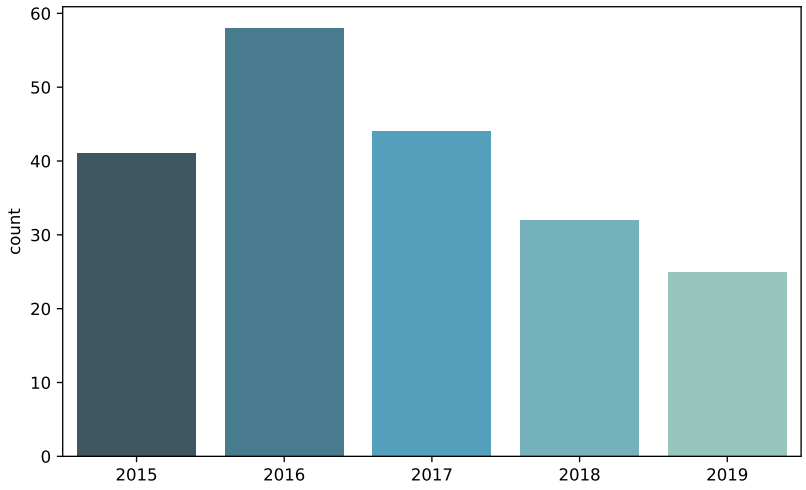


Figure 5.6: Distribution of judgements per year.

6 Learning on German Legal Judgments

The decision reasons of judgments are the argumentation of the judge why a case is dismissed or accepted. These decision reasons are written in *Urteilsstil*. Therefore, detecting parts of the writing style *Urteilsstil* should be comparable to detecting arguments and their components in text. In the paper "Argumentation Mining: The Detection, Classification and Structure of Arguments in Text" [PM09] Mochales Palau and Moens classify argumentative propositions in judgements of the European Court of Human Rights (ECHR). The approach used in [PM09] is tested on the corpus created for this thesis. Furthermore, the approach is tailored towards detecting *definitions* and *subumptions*.

For classification purposes the corpus is segmented into sentences. This task proved rather difficult. The standard python natural language library NLTK[1] has a German codec to work with German text. However, the German Legal Judgements contain more punctuation than "normal" German text. NLTK interprets this additional punctuation as sentence limiters, leading to incorrectly segmented sentences. A different well-known natural language processing tool is spaCy[2]. This tool offers two models for German. Both are trained on the TIGER and WikiNER corpus. spaCy uses a dependency parser to detect sentence boundaries and works with a statistical model of the language underneath. This statistical model is the week point of the approach. The model is trained on the above mentioned corpora that are well suited for general purpose language but not for specialised tasks like the one at hand. Therefore, even this library failed to segment the sentences in a correct way. In the end the tokeniser and sentence splitter SoMaJo[3] is the only tool that segmented the sentences mostly correct. Unfortunately, SoMaJo

[1] `https://www.nltk.org/`
[2] `https://spacy.io/`
[3] `https://github.com/tsproisl/SoMaJo`

has no function to only segment sentences, the text is always tokenised first. To get the segmented sentences the token are joined together with a white-space, leading to sentences with blanks before and after every token, even before periods. Since no other library or tool is able to perform as good as SoMaJo these additional blanks have to be accepted into the corpus.

The remainder of this chapter begins with the adaption of the base approach from [PM09]. Subsequently a novel approach for detecting *definition* and *subsumption* is presented. In the end both approaches are discussed.

The implementation is done in a Jupyter Notebook that is digitally attached to this thesis.

6.1 Base approach

Mochales Palau and Moens follow a two-tier approach for the classification of premisses and claims in judgements of the ECHR. At first a classifier decides whether a sentence is argumentative or not. Building on the decision of the first classifier a second one detects premises and claims in the argumentative text. Mochales Palau and Moens use a corpus of 47 judgments of the ECHR. The decision reasons are the main argumentative parts of a judgement. The used writing style should be comparable to classic argumentation. That is why the approach of Mochales Palau and Moens should lead to a meaningful classification of `definitions` and `subsumptions`. The approach is adapted to the task at hand and some features are changed. However, the results of [PM09] form the baseline for the classification task of this thesis.

The machine learning follows a standard pipeline for two times, as shown in figure 6.1. At first the corpus data is loaded and prepared for the first classification, then the features are engineered and the first model is trained and evaluated. The trained model is then used to make predictions, which are a feature for the second classifier. Following the pipeline the second time the remaining features are engineered, the second model is trained and evaluated and the final predictions about `definition` or `subsumption` can be done.

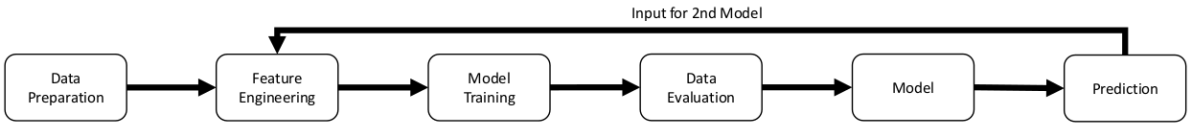


Figure 6.1: Machine Learning pipeline for the thesis. At first the data are prepared, then features are extracted and a first model is trained, after the data are evaluated the predictions of the first model are an input for the second classification. The pipeline is followed again and in the end the finished model can make predictions.

6.1.1 Detection of Argumentative Text

In the first classification Mochales Palau and Moens detect argumentative text in the ECHR corpus. For this classification they use an maximum entropy model and the following features:

- **Unigrams**
  Every word in the sentence.
- **Bigrams**
  Pair of successive words in the sentence.
- **Trigrams**
  Three successive words in the sentence.
- **Adverbs**
  Have to be detected by a part-of-speech (POS) tagger.
- **Verbs**
  Have to be detected by a POS tagger. Excluding common verbs like “to be”, “to do” or “to have”.
- **Modal Auxiliary**
  Modal auxiliary indicate wishes, possibilities or compulsions. A binary feature indicates the presence of such a verb, detected by a POS tagger.
- **Word Couples**
  All possible permutations of words in a sentence.
- **Text Statistics**
  Length of a sentence, average length of words and number of punctuation marks.

- **Punctuation**
  In which sequence the punctuation marks are present in a sentence.
- **Key Words**
  Keywords that indicate argumentation.
- **Parse Features**
  Depth of parse tree and number of subclauses of the parsing tree of each sentence.

This classification can be adapted to detect whether the sentence at hand should be considered for further classification into *definition* or *subsumption* or not. The maximum entropy model used by the authors is also known as logistic regression. For this thesis the *scikit-learn*[4] implementation of the *LogisticRegression*[5] is used.

To adapt the approach of the base paper the features mentioned above are changed/implemented in the following way:

- In contrast to [PM09] token are regarded as the smallest unit of a sentence and not words. By using token special characters like "§" are included into the classification which could be essential for the results.
- N-grams are extracted with the *scikit-learn* implementation *Count Vectorizer*[6].
- For detecting adverbs, verbs and modal auxiliary the *TreeTagger*[7] from Helmut Schmid is used. It is a state of the art part-of-speech tagger for German and other languages.
- The word couple feature has to be excluded due to the size of the feature space. Even on a subset of only 50 judgements the resulting vocabulary contains over two million word couples.
- Unfortunately, no keyword lists for *"Urteilsstil"* and *"Gutachtenstil"* exist. Good language and eloquent writing is one of the main goals law practitioners want to archive within their texts. Therefore, predefined keyword lists would be counter

[4] `https://scikit-learn.org/stable/index.html`
[5] `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html`
[6] `https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html`
[7] `https://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/`

productive. Due to this lack of a list the keyword feature is excluded from this thesis.

- Parse features are excluded a feature because these would require a NLP library like NLTK or spaCy that are not able to handle German legal judgements. spaCy offers the possibility to train their segmentation model. However, the time restrictions ot the thesis do not allow for detouring in training an additional model for obtaining meaningful parse features.

For the first classification the labels `definition`, `subsumption` and `other` are changed to `interest` (definition, subsumption) and `no_interest`. This change allows for a binary classification. At first the logistic regression is trained stand alone on every feature with a five-fold cross-validation. Five folds are used because this is the standard parameter in the *K-Folds cross-validator*[8] implementation of *scikit-learn*.

Table 6.1 shows the result of the stand alone training. The precision and recall for unigrams and bigrams is relatively high. Unfortunately, [PM09] does not report how their features performed in the first classification, thus, a comparison is not possible. Models trained on the features `modal auxiliary` and `punctuation sequence` always predict the majority class `interest`.

The base paper does not clearly state if all features are used at once of if the best feature of the reported ones is used. Therefore, the logistic regression is trained on all features at once by concatenating the feature vectors. This approach ran into memory problems due to the vocabulary of bigrams and trigrams. These two features are subsequently excluded.

A model trained with this concatenated vector always predicts the majority class, reaching a precision of 0.37, a recall of 0.5, a F-1 measure of 0.43 and an accuracy of 0.74. It seems unlikely that [PM09] used this method for their classification. Therefore, unigrams are chosen as best feature to train the logistic regression for the first classification task.

[8] `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html`

Feature	Precision	Recall	F1-Measure	Accuracy
Unigrams	**0.84**	**0.82**	**0.83**	**0.87**
Bigrams	0.79	0.79	0.79	0.84
Trigrams	0.78	0.65	0.67	0.80
Adverbs	0.74	0.68	0.70	0.80
Verbs	0.80	0.74	0.76	0.83
Modal Auxiliary	0.37	0.50	0.42	0.74
Sentence Statistics	0.78	0.64	0.66	0.80
Punctuation Sequence	0.37	0.50	0.43	0.74

Table 6.1: Results of the evaluation of the first classifier training. Unigrams reach the best results in all metrics.

6.1.2 Classification of Premises and Claims

In the second classification Mochales Palau and Moens only consider argumentative text and classify it into `premise` or `claim`. Following this logic the classification is adapted to `definition` and `subsumption`. All text labelled as `other` is excluded from the corpus. For the classficication task the authors use a support vector machine (SVM) and the following features:

- **Absolute Location**
  The absolute location of a sentence in the document.
- **Sentence Length**
  A binary feature that indicates if a sentence is longer than 12 words.
- **Tense of Main Verb**
  Can take the values: `Present`, `Past`, `NoVerb`
- **History**
  Argumentative category of last and next sentences.
- **Information 1st Classifier**
  Classification as argumentative/ not argumentative.
- **Rhetorical Patterns**
  Type of the rhetorical patterns in the current, previous and next sentences.

- **Article Reference**
  Binary feature whether an article of of the law is contained in the sentence. Detected with a POS tagger.

- **Article**
  Binary feature whether a definition of an article of of the law is contained in the sentence. Detected with a POS tagger.

- **Argumentative Patterns**
  Type of argumentative patterns that are contained in the sentence.

- **Type of Subject**
  Type of agent in the sentence. Detected with a POS tagger.

- **Type of Main Verb**
  One of four different argumentative types (premise, conclusion, final decision, none). Detected with a POS tagger and a predefined list.

In the thesis the features are adapted in the following way:

- Additional to the absolute location in the document the absolute sentence position in the paragraph is extracted.

- In the **Sentence Length** feature token are used instead of words.

- The features **Tense of Main Verb**, **History**, **Rhetorical Patterns**, **Argumentative Patterns**, **Type of Subject** and **Type of Main Verb** are excluded. Some of these features are extracted with predefined word lists that are not available for this thesis. Other features like **History**, **Rhetorical Patterns**, **Argumentative Patterns** do not seem to be applicable to the classification into `definition` and `subsumption`.

- **Article Reference** and **Article** are excluded because these two features are part of the required classification outcome. Furthermore, it is not clear from the paper how these features are extracted.

For the second classification the *scikit-learn* implementation of an SVM, the *Linear Support Vector Classification*[9], is used. Due to the exclusion of eight features and only one new feature the resulting feature vector is short and not as informative as the

[9] `https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html`

original one. Additionally, the exclusion of all sentences labelled as `other` excludes 6,481 sentences from the corpus.

The SVM is trained with a five fold cross-validation on 80% of the remaining corpus. In this setup the resulting models chose to only predict the majority class `subsumption`. Training the model on the whole training data the model is able to generate a recall of 0.96 for the `definition` label with a precision 0.28. The model seems to favour the minority class now. The macro averaged precision is 0.47 and the macro averaged recall 0.49 with a macro averaged F1-score of 0.25. These results show that the two-tier approach, as executed, is not suitable for the task. Therefore, a new approach is presented below.

6.2 Novel Approach

A two-tier classification approach introduces the error of the first classification into the second one. To avoid this a multi-class classification can be used. The implementation of the logistic regression and the SVM in *scikit learn* offers a multi-class classification option. The algorithm decomposes the multi class problem into a binary problem. The correct prediction is determined in a one vs. the rest approach.

Since `unigrams` performed best in the first classification task this feature is used to train a logistic regression and a SVM in a multi class setting. However, one-hot-encodings on huge vocabularies like the one at hand are generally a bad idea. The term frequency - inverse document frequency (tf-idf) computes a value for every term in all documents and ranks it accordingly. The *TfidfVecorizer*[10] implementation in *scikit learn* is used for this classification.

Tabel 6.2 shows the results of a five fold cross-validation over logistic regressions and SVM trained with either unigrams or tf-idf. The tf-idf outperforms both unigram models. The SVM is slightly better than the logistic regression in recall and thus F1-measure. Both models are saved for later usage on exercise case solutions, due to only minor performance differences.

[10] `https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html`

Feature and Classifier	Precision	Recall	F1-Measure	Accuracy
Unigrams/Logistic Regression	0.77	0.75	0.76	0.77
Unigrams/SVM	0.74	0.74	0.74	0.75
tf-idf/Logistic Regression	**0.80**	0.74	0.76	**0.79**
tf-idf/SVM	0.78	**0.76**	**0.77**	**0.79**

Table 6.2: Results of the evaluation of the novel classifier training. Tf-idf vectors reach the best results in all metrics.

6.3 Discussion

The baseline of [PM09] could not be reached within this thesis. This can be explained due to the many changes that are necessary to fit the thesis task. A different corpus from a slightly different domain and a different language is used. The features are adapted towards *Urteilsstil* and away from classical argumentation. In addition a different implementation of the classification algorithms is used. Nevertheless, the first binary classification lead to promising results. The exclusion of eight of eleven features in the second classification is very problematic. The remaining feature vector is not informative enough to train a model beyond choosing the majority class. The argumentation specific features can not be used for writing style components, the required features are just too different.

However, a most basic tf-idf feature with a multi-class classification already leads to promising results on the corpus. Therefore, the corpus seems to be a good basis for the detection of `definition` and `subsumption` in German legal judgements.

As pointed out above the two-tier approach introduces the error of the first classification task. This can be prevented by using multi clsss classification approaches.

In future work more sophisticated features like sentence embeddings should be explored.

7 Transferring to Exercise Cases

In theory the definitions and subsumptions used in *Urteilsstil* should have the same structure as the ones used in *Gutachtenstil*. Therefore, the model trained on the judgments is used on a small corpus of exercise case solutions. These solutions are written in *Gutachtenstil*.

This chapter introduces the exercise case solution corpus, describes the classification on the corpus and discusses the outcomes of the classification.

7.1 Exercise Case Solution Corpus

The solution corpus consists of 14 case solutions provided by Prof. Dr. Kramer, professorship of public law at the university of Passau. These are solutions for exercise cases, for students in the first semesters, of their legal studies. Therefore the argumentation is very detailed. Furthermore, the solutions contain editorial notes, that help students to better understand the problems. These 14 solutions are annotated by the same annotator as the judgment corpus.

The corpus consists of 1.451 sentences. 343 (24%) of them are `definitions`, 503 (34%) `subsumptions` and the remaining 605 (42%) sentences are labelled as `other`. Figure 7.1 shows over all solutions how many sentences are labelled as `definitions`, `subsumptions` or `other` and how many sentences solutions contain. Over all documents the `definition` has the least amount of sentences, with a median of 25.5 sentences. `Subsumptions` have a slightly higher median of 27 sentences. In contrast of the sentence distribution of `definition` the distribution of `subsumption` sentences resembles the distribution of the overall sentences per solution. This is the same behaviour as seen in judgements. Sentences labelled with `other` have a median of 42.5 sentences, which means that on average the solutions consist of more not relevant text than judgments. This deviation

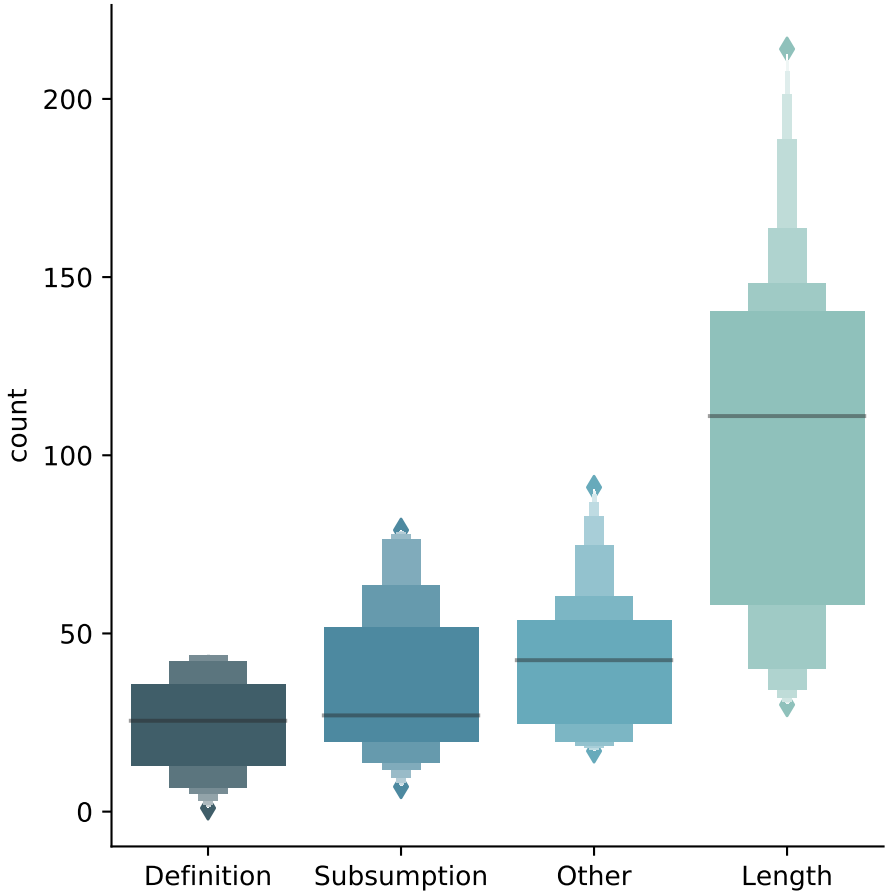


Figure 7.1: Number of sentences for all labels and all exercise case solutions. The last box represents the number of sentences in one document over all labels.

can be explained with the editorial notes that do not belong to the writing style and are mostly marked as `other`. It is a main deviation from the judgement corpus, where most sentences are labelled as `subsumption` and most sentences are relevant for the classification. Solutions have a median of 111 sentences, at least 30 and at most 214. The lower bound of sentences is comparable to the 38 sentences lower bound in the judgement corpus. The case with the most sentences has significantly less sentences than the judgement with the most sentences (862). However, this judgement is an outlier. More important is the median of sentences in the document. The case median is 14 sentences higher than the median of 97 that can be found within the judgement corpus. In other words, cases are on average longer than judgments and contain more sentences not marked as `definition` or `subsumption`.

7.2 Classification

In chapter 6.2 two models are trained with the tf-idf feature. The first one utilises a logistic regression and the second one a SVM.

These models are loaded and the sentences are classified with the pre trained models. Table 7.1 shows that Logistic Regression outperforms the SVM slightly. Both models favour the majority class from the corpus they were trained on, `subsumtion`. Unfortunately, the majority class of this corpus is `other`.

Feature and Classifier	Precision	Recall	F1-Measure	Accuracy
tf-idf/Logistic Regression	**0.60**	**0.57**	**0.56**	**0.57**
tf-idf/SVM	0.59	0.54	0.53	0.54

Table 7.1: Results of the evaluation of the classification of exercise case solutions. Logistic Regression outperforms the SVM but both lead to bad results.

7.3 Discussion

In theory `definition` and `subsumption` should be the same in both *Urteilsstil* and *Gutachtenstil*. However, the model trained on the judgement corpus brings its own biases into the new classification. Furthermore, the editorial notes are not always marked as `other`. These notes are not written in the *Gutachtenstil* which complicated the classification task. In addition 14 case solution is a really small sample, a bigger corpus might lead to better results.

In future work the editorial notes should be removed and a larger corpus should be used.

8 Conclusion

This chapter concludes this thesis. At first the thesis is summarised and implications and limitations are discussed. Furthermore, possible future work is introduced.

8.1 Thesis Summary

This thesis addressed the problem of recognising two parts of German legal writing styles in free text. Natural language processing research on legal text did not consider German legal writing styles until now. To fill this gap this thesis utilised argumentation mining techniques for detecting parts of German legal writing styles.

Argumentation Mining is a supervised machine learning task, that needs annotated corpora. Since no research in this area was done before no corpora existed. Therefore, this thesis introduced two novel German law corpora. The first one contains 31,334 German court decisions and the corresponding meta data. The second corpus consists of 200 randomly chosen judgments. In these judgements sentences are labelled with the *Urteilsstil* components `definition` and `subsumtion`. Every sentence that is not marked as one of these components is labelled with `other`. Leading to a fully labelled corpus, that misses the third component of the *Urteilsstil*, the conclusion. The corpus was annotated by a legal expert, who passed the first legal state exam.

The judgement corpus was used to answer the following research questions:

1. Is it possible to automatically recognise the definition part of the *Urteilsstil* in a legal decision, using argumentation mining techniques?
2. Is it possible to automatically recognise the subsumption part of the *Urteilsstil* in a legal decision, using argumentation mining techniques?

3. Is it possible to train a model on legal decisions in *Urteilsstil* and use it on exercise case solutions in *Gutachtenstil*?

The paper "Argumentation Mining: The Detection, Classification and Structure of Arguments in Text" from Mochales Palau and Moens formed the baseline for the first two questions. It was theorised that German legal writing style components resemble classical argumentation components, namely premise and claim. Therefore, the two tier approach used in the paper was adapted to the writing style detection. The used features were either adjusted to the writing style or excluded. Especially in the second tier of the classification a lot of features had to be excluded since they were tailored to general argumentation. However, legal writing styles can not directly be matched into the premise and claim scheme of general argumentation. Subsumptions and definitions are an abstraction level away from premises and claims, thus rendering the specific features of argumentation mining not applicable for the task. Hence the first two questions must be answered in the negative.

Nevertheless, by excluding the use of the argumentation mining techniques promising results can be produced on the judgement corpus. A model trained on simple tf-idf vectors and a multi-class classification reached a precision of 0.8(0.78) with a recall of 0.74(0.76). These results clearly indicate that judgement corpus can be used for the writing style detection task.

Unfortunately, the third question must also be answered negatively. The change of the domain from *Urteilsstil* to *Gutachtenstil* and the changed majority class proved too much for the pre-trained models. Additionally, the case solution corpus of 14 solutions was very small. It also contained editorial notes that are not written in *Gutachtenstil*. They should help the students to better understand the solutions, but complicated the classification task.

8.2 Future Work

In future work the classification should focus on multi class classification. Features should be generated in a writing style specific way, comparable to the argumentation specific features of Mochales Palau and Moens. In addition sentence embeddings like word2vec and glove should be explored. Even transformer models like BERT might be an interesting research direction. When changing the domain form judgements to case solutions the solution corpus should be increased and editorial notes excluded.

A different research direction would be the detection of the interaction of subsumption and definition. These interactions could be especially interesting in nested cases. Another open question is, how similar judgements of different courts are. Some law professionals see the writing style as rough guidelines. Exploring the differences between the courts and the law professionals seems interesting.

The judgement corpus will be extended by a conclusion label. Furthermore, the given labels should be checked by further legal experts to reach a better gold standard.

The decision corpus could be used to generally explore the German law system. It might be possible to recognise law professionals by their writing style. If this recognition is possible one could trace the law professionals through the courts and thus their career. The seasonality of topics in courts and the outcomes of decisions on different days of the year might also be an interesting topic.

In other future work, the detected definitions can be ontologically modelled for argumentation mining, using the procedure of [Mit+17]. Furthermore, legal corpora can help model legal terminology into upper level ontologies like SUMO [MPG19].

A Code

```
import scrapy  # scraping library, does all the heavy lifting
from scrapy.linkextractors import LinkExtractor # library to extract links from a page
import os
import json
import re

class LegalDecisions(scrapy.Spider):
    # name has to be unique in my project
    name = 'legal_decisions_small'
    # directory where I want to save the scraped legal decisions
    save_dir = 'decisions'

    # this function starts the spider
    def start_requests(self):
        # if this is the first document of this kind, make a new directory
        if not os.path.exists(self.save_dir):
        os.mkdir(self.save_dir)
        # scraping starting point
        urls = ['https://www.gesetze-bayern.de/Search/Filter/DOKTYP/rspr']
        for url in urls:
            # in contrast to the return statement I do not only return a value to my caller but also retain enough state
            # information to pick up my function after the last yield run
            # the request should not be filtered because it gets redirected to the same URL
            # and this would be filtered out
            yield scrapy.Request(url, self.parse_legal, dont_filter=True)

    # callback function to handle the response, by selecting all links to decisions and then going to the next page
    def parse_legal(self, response):
        # response is an HTML document of the whole website
        # we traverse the DOM and get all list elements wit the class hitlistItem
        # .getall gives us all items that match the description
        for list_item in response.css('li.hitlistItem').getall():
            # if the list item is a judgment we consider it further
            if 'Urteil␣' in list_item:
                # get the link of the judgment
                url = re.findall('href=".+"', list_item)[0]
                # cut the HTML
                url = url[6:len(url)-1]
```

```
39                  # add the first part of the url to every link
40                  url_full = 'https://www.gesetze-bayern.de' + url
41                  # create a new request for every link
42                  yield scrapy.Request(url_full, self.parse_decision)
43
44          # after parsing all links the spider should go to the next page
45          # the link to the next page is always the last item of the pagination
46          next_page = LinkExtractor(restrict_css='ul.pagination␣:last-child').
                extract_links(response)
47          # as we got an list of all links of the pagination we have to select the last
                one to go to the next page
48          next_page = next_page[len(next_page)-1].url
49
50          # as long as there is a next page crawl it
51          if next_page is not None:
52          yield scrapy.Request(next_page, self.parse_legal, dont_filter=True)
53
54      # function to parse the legal decisions
55      # just get the name, the title and the reasoning part of the decision and
            disregard everything else
56      def parse_decision(self, response):
57          # get the metadata from the doc metadata
58          name = response.css('h1.titelzeile::text').get()
59          meta_title = response.xpath('//div[@id="doc-metadata"]/div/div/b/text()').get
                ()
60          meta_title = re.sub('/', '$', meta_title)
61          # search for a h2 that contains the text "ründe"
62          # to drive scrapers nuts they decided to not be consistent with their headings
63          # get everything after this h2
64          reasoning = response.xpath(
65          '//h2[contains(.,"ründe")]/following-sibling::div/div[@class="absatz␣gruende
                "]/text()').getall()
66          # join the list to one string but keep the structure with new lines
67          reasoning = '␣\n␣'.join(reasoning)
68          # put everything together in one JSON for later use
69          final_format = {
70              'title': meta_title,
71              'name': name,
72              'text': reasoning
73          }
74          # save each decision as extra file and ensure the right encoding
75          with open('{}/{}.json'.format(self.save_dir, meta_title), 'w', encoding='utf-8
                ') as f:
76              json.dump(final_format, f, ensure_ascii=False)
77          yield
```

Listing A.1: Scraping of the simple corpus with srapy. Only the document title, legal judgment title and grounds for decision are extracted. Implicit meta data like paragraph structure is discarded.

```
import json
import os
from random import randint
import re

# The script goes throug the judgments in the folder and saves it with a 5% chance.
# Only decisions that have grounds for decisions with over 8000 characters are concidered.
# 25 judgments are bached into one file, 8 files are saved. This leads to 200 judgments. The batch of 25 is neccessary due to the upload constrains of one MB in doccano.
# In the end the names of all scraped decisions is saved into an extra file.

path = 'master_thesis_scrape/decisions/'
corpus = ''
i = 0
count = 1
scraped_decisions = []

for file in os.listdir(path):
    if randint(1, 101) < 6:
        with open (path + file, encoding='utf8') as f:
            data = json.load(f)
            if len(data['text']) > 8000:
                corpus += '{"text": "' + data['text'] + '"}'
                    i += 1
                    scraped_decisions.append(data['title'])
    if i == 25 and count < 9:
        with open('corpus' + str(count) + '.json', 'w', encoding='utf-8') as file:
            json.dump(corpus, file, ensure_ascii=False)
            corpus = ''
            count += 1
            i = 0
with open ('scraped_descisions.txt', 'w', encoding='utf-8') as file:
    file.write('\\n'.join(str(item) for item in scraped_decisions))
```

Listing A.2: Extraction of the randomly chosen 200 legal judgments. The legal judgments are saved into batches of 25 to adhere to the upload limit of 1MB.

```
import scrapy  # scraping library, does all the heavy lifting
from scrapy.linkextractors import LinkExtractor  # library to extract links from a page
import os
import json
import re

class LegalDecisions(scrapy.Spider):
    # name has to be unique in my project
    name = 'legal_decisions_big'
    # directory where I want to save the scraped legal decisions
    save_dir = 'decisions_big'

    # this function starts the spider
    def start_requests(self):
        # if this is the first document of this kind, make a new directory
        if not os.path.exists(self.save_dir):
            os.mkdir(self.save_dir)
        # scraping starting point
        urls = ['https://www.gesetze-bayern.de/Search/Filter/DOKTYP/rspr']
        for url in urls:
            # in contrast to the return statement I do not only return a value to my caller but also retain enough state
            # information to pick up my function after the last yield run
            # the request should not be filtered because it gets redirected to the same URL
            # and this would be filtered out
            yield scrapy.Request(url, self.parse_legal, dont_filter=True)

    # callback function to handle the response, by selecting all links to decisions and then going to the next page
    def parse_legal(self, response):
        # response is an HTML document of the whole website
        # traverse the DOM by using the CSS class hltitle and child elements that are links
        # the div above is needed to uniquely identify the attribute, would scrape all links on the website otherwise
        # .getall gives us all links that are a child of this div on the page
        for url in response.css('div.hltitel a::attr(href)').getall():
            # add the first part of the url to every link, because of horrible webdesign
            url_full = 'https://www.gesetze-bayern.de' + url
            # create a new request for every link
            yield scrapy.Request(url_full, self.parse_decision)
        # after parsing all links the spider should go to the next page
        # the link to the next page is always the last item of the pagination
        next_page = LinkExtractor(restrict_css='ul.pagination :last-child').extract_links(response)
        # as we got an list of all links of the pagination we have to select the last one to go to the next page
        next_page = next_page[len(next_page) - 1].url
        # as long as there is a next page crawl it
        if next_page is not None:
```

```
45              yield scrapy.Request(next_page, self.parse_legal, dont_filter=True)
46
47      # function to parse the legal decisions
48      # just get the name, the title and the reasoning part of the decision and
            disregard everything else
49      def parse_decision(self, response):
50          # get the metadata from the doc metadata
51          title = response.css('h1.titelzeile::text').get()
52          meta_title = response.xpath('//div[@id="doc-metadata"]/div/div/b/text()').get
                ()
53          court = re.findall(r'[A-Za-zäöüÄÖÜ ]+,', meta_title)[0]
54          decision_style = re.findall(r', [A-Za-z ]+ v\.', meta_title)[0]
55          decision_date = re.findall(r'\d\d.\d\d.\d\d\d\d', meta_title)[0]
56          decision_id = re.findall(r'- .+', meta_title)[0]
57          meta_title = re.sub('/', '$', meta_title)
58          # clean the metadata
59          court = court[:len(court) - 1]
60          decision_style = decision_style[2:len(decision_style) - 3]
61          decision_id = decision_id[2:]
62          # get meta data from the rsprbox
63          # get all rsprboxzeile divs that are between a rsprboxueber that contains "
                nemkette" and a rsprboxueber that
64          # either contains "chlagw" of "eits", because there are not always guiding
                guidelines but if they exists they
65          # come before the keywords
66          norm_chains = response.xpath('//div[@class="rsprboxueber"][contains(.,"
                menkette")]/following-sibling::div[@class="rsprboxzeile"][following-
                sibling::div[@class="rsprboxueber"][contains(.,"chlagw") or contains(., "
                eits")]]/text()').getall()
67          decision_guidelines = response.css('div.leitsatz::text').getall()
68          decision_keywords = ""
69          # all rsprboxzeile that follow on a rsprboxueber that contains "instan" and
                are followed by a rsprboxueber that
70          # contains "undstell"
71          lower_court = response.xpath('//div[@class="rsprboxueber"][contains(.,"instan
                ")]/following-sibling::div[@class="rsprboxzeile"][following-sibling::div[
                @class="rsprboxueber"][contains(., "undstell")]]/text()').getall()
72          # quick and dirty fix for additional information that is mixed in the lower
                court, the code was ignoring the
73          # contains for the following sibling from above, could not fix it otherwise
74          if 'Revision zugelassen' in lower_court:
75              lower_court.remove('Revision zugelassen')
76          additional_information = ""
77          decision_reference = ""
78          # to not have null values in the JSON the response is checked for values
                before the values are assigned
79          if response.xpath(
80          '//div[@class="rsprboxueber"][contains(.,"lagwort")]/following-sibling::div[
                @class="rsprboxzeile"]/text()').get() is not None:
81              decision_keywords = response.xpath('//div[@class="rsprboxueber"][contains
                    (.,"lagwort")]/following-sibling::div[@class="rsprboxzeile"]/text()').
                    get()
82          if response.xpath('//div[@class="rsprboxueber"][contains(.,"eiterf")]/
```

```
            following-sibling::div[@class="rsprboxzeile"]/text()').get() is not None:
83              additional_information = response.xpath('//div[@class="rsprboxueber"][
                    contains(.,"eiterf")]/following-sibling::div[@class="rsprboxzeile"]/
                    text()').get()
84          if response.xpath('//div[@class="rsprboxueber"][contains(.,"undstell")]/
                following-sibling::div[@class="rsprboxzeile"]/text()').get() is not None:
85              decision_reference = response.xpath('//div[@class="rsprboxueber"][contains
                    (.,"undstell")]/following-sibling::div[@class="rsprboxzeile"]/text()').
                    get()
86          # get the text from every div with the class 'absatz tenor'
87          tenor = response.xpath(
88          '//h2[contains(.,"enor")]/following-sibling::div/div[@class="absatz tenor"]/
                text()').getall()
89          # get every div with the class absatz tatbestand that follows on a h2 that
                contains 'bestand'
90          legal_facts = response.xpath('//h2[contains(.,"bestand")]/following-sibling::
                div/div[@class="absatz tatbestand"]/text()').getall()
91          # get every node after the h2 that contains 'ründe', it is saved to a list
92          decision_reasons = response.xpath('//h2[contains(.,"ründe")]/following-sibling
                ::div/div[@class="absatz gruende"]/text()').getall()
93          # save all components in a JSON file
94          final_format = {
95              'meta': {
96                  'meta_title': meta_title,
97                  'court': court,
98                  'decision_style': decision_style,
99                  'date': decision_date,
100                 'file_number': decision_id,
101                 'title': title,
102                 'norm_chains': norm_chains,
103                 'decision_guidelines': decision_guidelines,
104                 'keywords': decision_keywords,
105                 'lower_court': lower_court,
106                 'additional_information': additional_information,
107                 'decision_reference': decision_reference
108             },
109             'decision_text': {
110                 "tenor": tenor,
111                 "legal_facts": legal_facts,
112                 "decision_reasons": decision_reasons
113             }
114         }
115         # save each decision as extra file and ensure the right encoding
116         with open('{}/{}.json'.format(self.save_dir, meta_title), 'w', encoding='utf-8
                ') as f:
117             json.dump(final_format, f, ensure_ascii=False)
118         yield
```

Listing A.3: Scraping of the full corpus. As much metadata as possible is preserved.

B Screenshots

Figure B.1: Title of a legal judgment document

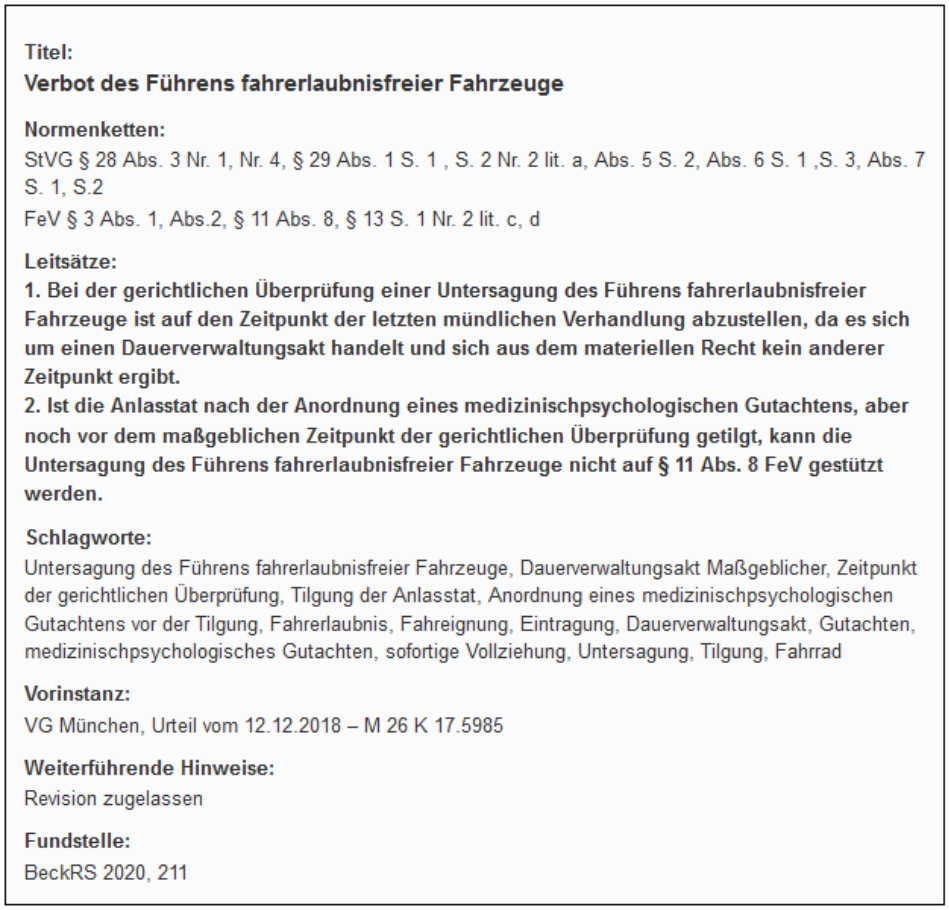


Figure B.2: Metadata of a legal judgment with all possible components.

Tenor

I. Die Beschwerde wird zurückgewiesen.

II. Die Antragsteller tragen die Kosten des Beschwerdeverfahrens einschließlich der außergerichtlichen Kosten der Beigeladenen als Gesamtschuldner.

III. Der Streitwert wird auf 3.750 Euro festgesetzt.

(a) Purport with Roman numerals for structuring

Tenor

1. Die Klage wird abgewiesen.

2. Der Kläger trägt die Kosten des Verfahrens.

(b) Purport with Arabic numbers for structuring.

Tenor

Der Beschluss des Amtsgerichts Kitzingen vom 16.11.2019 wird dahingehend ergänzt, dass der Tenor wie folgt lautet:

„Das gegen den Richter am Amtsgericht ... gerichtete Ablehnungsgesuch des Verteidigers vom 07.11.2019 wird als unbegründet zurückgewiesen."

(c) Purport without structuring.

Figure B.3: Different varieties of purport structuring available on the website.

Tatbestand

I.

1 Die Parteien streiten um Rundfunkbeiträge für den Zeitraum vom 1. Januar 2016 bis 31. Dezember 2016.

2 Der Kläger ist Inhaber einer Arztpraxis unter der Anschrift ... Der Beklagte führt für den Kläger unter der Anschrift ... ein Beitragskonto mit der Nummer ... für eine nicht private Nutzung (Betriebsstätte).

(a) Offenses with Roman numerals for structuring

Tatbestand

1 Der Kläger wendet sich gegen den Widerruf seiner Erlaubnisse zur Führung der Berufsbezeichnungen „Masseur und medizinischer Bademeister“ sowie „Physiotherapeut“.

2 1. Dem am ... ... 1963 geborenen Kläger wurde mit Urkunde der Regierung von Schwaben vom 15. April 1988 mit Wirkung ab 1. April 1987 die Erlaubnis zur Ausübung einer Tätigkeit unter der Berufsbezeichnung „Masseur und medizinischer Bademeister“ erteilt. Mit Urkunde des Regierungspräsidiums Stuttgart vom 26. August 1996 wurde ihm die Erlaubnis zum Führen der Berufsbezeichnung „Physiotherapeut“ erteilt.

3 2. Mit unanfechtbar gewordenem Bescheid der Regierung von ... vom 30. November 2015 wurden die Erlaubnisse wegen beruflicher Unzuverlässigkeit und gesundheitlicher Nichteignung zur Berufsausübung widerrufen und die entsprechenden Urkunden eingezogen. Begründet wurde der Widerruf mit der Unzuverlässigkeit des Klägers zur Berufsausübung sowie mit seiner gesundheitlichen Ungeeignetheit aufgrund seiner Alkoholsucht. Diese habe sich in mehreren alkoholbedingten Ausfallerscheinungen des Klägers in seiner Praxis sowie in drei Trunkenheitsfahrten im Zeitraum vom 23. April 2015 bis 25. September 2015 gezeigt.

4 Die Trunkenheitsfahrten am 23. April und 30. April 2015 hatten zu einer rechtskräftigen Verurteilung durch das Amtsgericht Obernburg am Main vom 19. November 2015 wegen fahrlässiger Trunkenheit im Verkehr in zwei tatmehrheitlichen Fällen, jeweils in Tateinheit mit vorsätzlichem Fahren ohne Fahrerlaubnis in Tatmehrheit mit Vorenthalten und Veruntreuen von Arbeitsentgelt zu einer Freiheitsstrafe von sechs Monaten auf Bewährung sowie einer Sperre der Fahrerlaubniserteilung bis zum 18. Mai 2017 geführt; die Strafe war dem Kläger mit Wirkung vom 22. November 2018 erlassen worden. Der weitere Sachverhalt, dass der Kläger am 25. September 2015 polizeilich angetroffen wurde, als er unter Alkoholeinfluss Fahrrad fuhr (Blutalkoholkonzentration 3,15 Promille), führte nicht zu einer strafrechtlichen Verurteilung.

5 3. Mit Bescheid der Regierung von Schwaben vom 19. Oktober 2017 wurde dem Kläger die Erlaubnis zum Führen der Berufsbezeichnung „Masseur und medizinischer Bademeister“ wieder

(b) Offenses with Arabic numbers for structuring.

Tatbestand

1 Der Kläger begehrt Rechtsschutz im Zusammenhang mit der Nutzung und Räumung eines von der Beklagten zur Verfügung gestellten Holzlagerplatzes.

2 Die Beklagte überlässt Holzlagerplätze in ihrem Gemeindegebiet an Gemeindeeinwohner. Das Benutzungsverhältnis wird durch einen privatrechtlichen Vertrag zwischen der Beklagten und dem jeweiligen Nutzer ausgestaltet. Die Vertragsparteien bezeichnen diesen ausweislich der dem

(c) Offenses without structuring.

Figure B.4: Different varieties of Offenses structuring available on the website.

Entscheidungsgründe

23 Die zulässige Klage ist nicht begründet.

24 Die im streitgegenständlichen Bescheid verfügte Ausweisung des Klägers und das fünf- bzw. siebenjährige Einreise- und Aufenthaltsverbot sind rechtmäßig und verletzten den Kläger nicht in seinen Rechten, § 113 Abs. 1 Satz 1 VwGO. Der Kläger hat keinen Anspruch auf Erteilung einer Aufenthaltserlaubnis, § 113 Abs. 5 VwGO.

I.

25 Die Ausweisung des Klägers erweist sich im maßgeblichen Zeitpunkt der Entscheidung des Gerichts (vgl. BVerwG, U.v. 15.1.2013 - 1 C 10.12 - juris Rn. 12) als rechtmäßig.

26 1. Nach § 53 Abs. 1 AufenthG wird ein Ausländer, dessen Aufenthalt die öffentliche Sicherheit und Ordnung gefährdet, ausgewiesen, wenn die unter Berücksichtigung aller Umstände des Einzelfalles vorzunehmende Abwägung der Interessen an der Ausreise mit den Interessen an einem weiteren Verbleib des Ausländers im Bundesgebiet ergibt, dass das öffentliche Interesse an der Ausreise überwiegt.

27 a.) Der weitere Aufenthalt des Klägers im Bundesgebiet stellt eine Gefährdung der öffentlichen Sicherheit dar, da mit beachtlicher Wahrscheinlichkeit davon auszugehen ist, dass der Kläger erneut erheblich straffällig wird (vgl. zum Prognosemaßstab BVerwG, U.v. 15.1.2013 - 1 C 10.12 - juris).

Figure B.5: Decision Reasons with Roman numerals for structuring

Entscheidungsgründe

20 1. Die Klage ist zulässig, insbesondere sind beide Kläger klagebefugt (§ 42 Abs. 2 VwGO).

21 1.1. Zwar hat der Kläger zu 1 kein Miteigentum an einem der nahe der Bahnstromleitung gelegenen Grundstücke bzw. dem durch den Leitungsmast (weiterhin) beanspruchten Grundstück FlNr. 1911. Auch in eine eigentumsbezogene Rechtsposition der Klägerin zu 2 wird durch den angegriffenen PFB nicht eingegriffen. Denn die für diesen Mast schon bestehende Grunddienstbarkeit bleibt unverändert; diejenigen Ersatzmasten, die - wie im Fall des Grundstücks FlNr. 1911 - an unveränderter Position gebaut werden, befinden sich trotz geringfügig größerer

Figure B.6: Decision Reasons with Arabic numbers for structuring.

Entscheidungsgründe

17 Über die Klage entscheidet das Gericht gemäß § 101 Abs. 2 VwGO ohne mündliche Verhandlung, weil die Beteiligten hierzu schriftlich ihr Einverständnis erklärt haben.

18 1. Die Klage ist unzulässig.

19 a) Die Klägerin hat ihre ursprüngliche Anfechtungsklage nach Erledigung der angegriffenen Nebenbestimmungen durch Aufhebung des streitgegenständlichen Bescheides (vgl. Art. 43 Abs. 2 BayVwVfG) in eine Fortsetzungsfeststellungsklage gemäß § 113 Abs. 1 Satz 4 VwGO umgestellt

Figure B.7: Decision Reasons with Arabic numbers and additional letters for structuring.

Entscheidungsgründe

74 Die Klage ist zulässig, insbesondere fristgerecht erhoben worden.

75 Der streitgegenständliche Ablehnungsbescheid des Bayerischen Landesamts für Steuern vom 14. September 2018 ist mit einer fehlerhaften Rechtsbehelfsbelehrung:versehen, da nicht auf die Möglichkeit hingewiesen wurde, dass gemäß Art. 15 Abs. 1 Satz 1 Nr. 5 VwGO anstelle der Erhebung der Klage fakultativ auch ein Widerspruchsverfahren durchgeführt werden kann.

Figure B.8: Decision Reasons without structuring.

Entscheidungsgründe

15 Die zulässige Klage hat auch in der Hauptsache Erfolg, nicht jedoch bezüglich der geltend gemachten Rechtsanwaltskosten.

A.

16 Das angerufene Gericht ist sachlich und örtlich zuständig (§§ 23, 71 Abs. 1 GVG; §§ 12, 13, ZPO).

17 Die Klageänderung hinsichtlich der Rechtsanwaltskosten ist gemäß § 264 Nr. 2 ZPO zulässig.

B.

18 Die Klage ist hinsichtlich des Unterlassungsbegehrens vollumfänglich begründet. Hinsichtlich der geltend gemachten Rechtsanwaltskosten ist die Klage jedoch unbegründet.

19 1. Die Klageseite hat gegen die Beklagte einen Anspruch auf Unterlassung der im Tenor genannten Äußerungen gemäß § 1004 Abs. 1 Satz 2 BGB (analog) i. V. m. § 823 Abs. 1 BGB, Artt. 1 Abs. 1, 2 Abs. 1 GG, § 186 StGB.

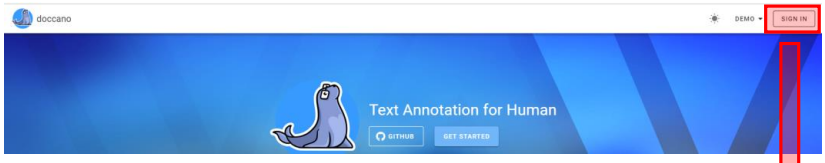
20 a) Der Schutzbereich des § 1004 BGB umfasst neben sämtlichen absoluten Rechten auch die in § 823 Abs. 1 BGB geschützten Rechtsgüter, so auch das allgemeine Persönlichkeitsrecht. Daher ist aus §§ 823, 1004 Abs. 1 Satz 2 BGB i. V. m. Arft. 1 Abs. 1, 2 Abs. 1 GG für den geltend gemachten Unterlassungsanspruch eine Gesamtanalogie zu bilden (Herrler in Palandt, § 1004 Rn. 4).

21 b) Vorliegend hat die Beklagte durch ihre Äußerungen gegenüber Dritten sowohl das allgemeine Persönlichkeitsrecht der Klägerin zu 1) (Artt. 1 Abs. 1, 2 Abs. 1 GG) als auch das Unternehmenspersönlichkeitsrecht der Klägerin zu 2) (Art. 2 Abs. 1 GG) verletzt.

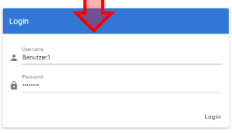
Figure B.9: Decision Reasons with several kinds of structuring.

C Annotation Guide

Öffnen Sie die Seite http://doccano.padim.fim.uni-passau.de/.
Klicken Sie rechts oben auf „SIGN IN“.

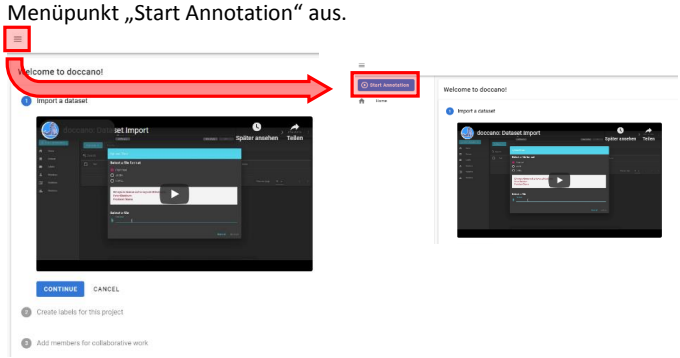


Geben Sie nun Ihre Benutzerdaten ein und klicken Sie auf „Login“
Nach dem Login sehen Sie die Übersicht über alle verfügbaren Projekte.

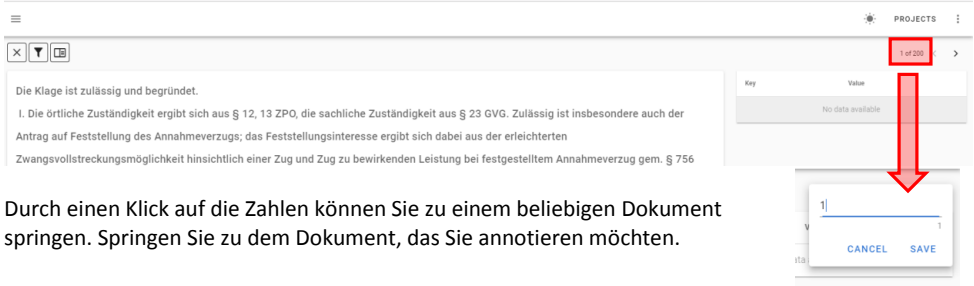


Wählen Sie das einzig verfügbare Projekt „Gerichtsentscheidungen“ durch einen Klick auf den blauen Link aus.

Klicken Sie auf der folgenden Seite links oben auf die drei Striche. Wählen Sie anschließend den Menüpunkt „Start Annotation“ aus.



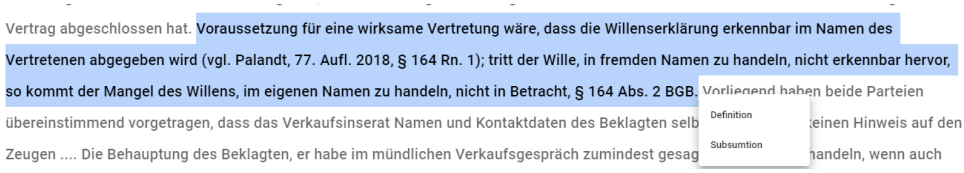
Kontrollieren Sie zunächst rechts oben bei welchem Dokument Sie sich befinden.



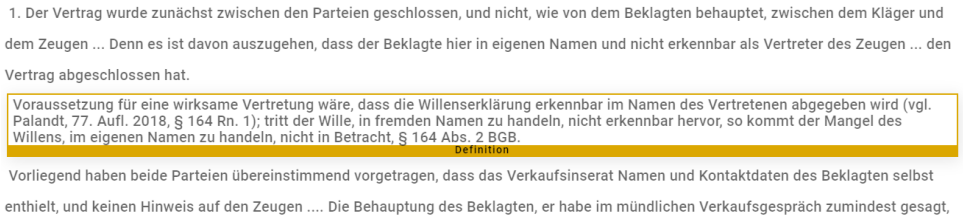
Durch einen Klick auf die Zahlen können Sie zu einem beliebigen Dokument springen. Springen Sie zu dem Dokument, das Sie annotieren möchten.

Die gegebenen Dokumente sind Entscheidungsgründe von Gerichtsentscheidungen.
Annotieren Sie in diesen alle Definitionen und Subsumtionen gemäß des Urteilsstils.
Um eine Textstelle zu annotieren müssen Sie diese mit einem Rechtsklick der Maus markieren.

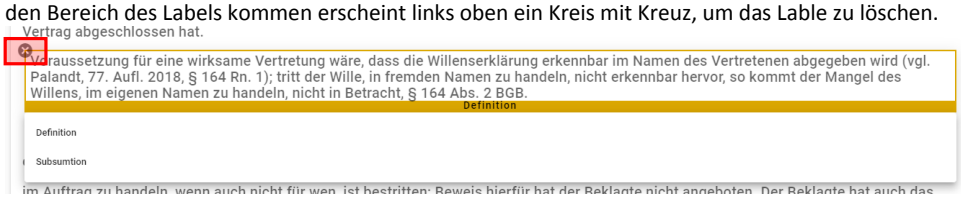
Ziehen Sie die Markierung soweit wie nötig. Sobald Sie die rechte Maustaste loslassen öffnet sich ein Pop-Up in dem Sie Definition oder Subsumtion als Label der Textstelle auswählen können.



Sobald Sie das entsprechende Lable ausgewählt haben wird der Text von diesem umschlossen.



Durch einen Klick auf den annotierten Text können Sie das Label ändern. Sobald Sie mit der Maus in den Bereich des Labels kommen erscheint links oben ein Kreis mit Kreuz, um das Lable zu löschen.



Für das vorliegende Projekt wird der Urteilsstil folgendermaßen definiert[1]:

	Endergebnis	Die Klage ist begründet. Der Bekl. schuldet dem Kl. 1500€ Schadensersatz.
Definition	Abstrakte Rechtsfolge/ Abstrakter Tatbestand (Gesetzeswortlaut)	Wird durch ein Tier eine Sache beschädigt, so ist nach §833 S1 BGB derjenige, der das Tier hält, verpflichtet, dem Verletzten den daraus entstehenden Sachschaden zu ersetzen.
Subsumtion	Feststellungssatz	Diese Voraussetzungen sind hier erfüllt.
	Ungefilterter Lebenssachverhalt	Der Hund des Bekl. zerkratzte den Lack des Wagens des Kl. Die Neulackierung verursachte Kosten iHv 1500€.

Wie auch im Gutachtenstil können im Urteilsstil Anspruchsgrundlagen oder Normen auf weitere Vorschriften und/oder dogmatische Unterpunkte verweisen. Dies Ändert den oben gezeigten Aufbau. Statt der Subsumtion folgt eine erneute Definition.

Bitte markieren Sie auch diese „geschachtelten“ Definitionen und Subsumtionen.

Beschränken Sie sich bei der Markierung von Definitionen und Subsumtionen bitte auf vollständige Sätze. Sollte ein Satz sowohl eine Definition als auch eine Subsumtion enthalten sollte dieser nicht markiert werden.

Bitte markieren Sie nur Textstellen, bei denen Sie sich absolut sicher sind.
Es kann durchaus vorkommen, dass Dokumente keine klar erkennbaren Definitionen und/oder Subsumtionen enthalten.

Sollten Sie Fragen oder Probleme bezüglich des Systems haben kontaktieren Sie bitte Stefanie Urchs unter Stefanie.Urchs@uni-passau.de

[1] Vgl. J. Danger, *Urteil und Urteilsstil in der zivilrechtlichen Assessorklausur: Eine praktische Hilfestellung*, JA 2005, 523

Bibliography

[Ajd58] Kazimierz Ajdukiewicz. *ABRISS DER LOGIK*. Aufbau-Verlag Berlin, 1958 (cit. on pp. 11, 12).

[Ale+16] Nikolaos Aletras et al. "Predicting judicial decisions of the European Court of Human Rights: A natural language processing perspective". In: *PeerJ Computer Science* 2 (2016), e93 (cit. on p. 7).

[Ale83] Robert Alexy. *Theorie der juristischen Argumentation*. Suhrkamp Verlag AG, July 25, 1983. 397 pp. ISBN: 3518280368. URL: `https://www.ebook.de/de/product/1243350/robert_alexy_theorie_der_juristischen_argumentation.html` (cit. on p. 21).

[AA91] Kevin D Ashley and Vincent Aleven. "Toward an intelligent tutoring system for teaching law students to argue with cases". In: *Proceedings of the 3rd international conference on Artificial intelligence and law*. 1991, pp. 42–52 (cit. on p. 5).

[AB09] Kevin D Ashley and Stefanie Brüninghaus. "Automatically classifying case texts and predicting outcomes". In: *Artificial Intelligence and Law* 17.2 (2009), pp. 125–165 (cit. on p. 6).

[Aso+15] Kartik Asooja et al. "Semantic annotation of finance regulatory text using multilabel classification". In: *LeDA-SWAn (to appear, 2015)* (2015) (cit. on p. 8).

[Ben87] Trevor JM Bench-Capon. "Support for policy makers: formulating legislation with the aid of logical models". In: *Proceedings of the 1st international conference on Artificial intelligence and law*. 1987, pp. 181–189 (cit. on p. 5).

[Bra91] L Karl Branting. "Reasoning with portions of precedents". In: *Proceedings of the 3rd international conference on Artificial intelligence and law*. 1991, pp. 145–154 (cit. on p. 5).

[BA99] Stefanie Brüninghaus and Kevin D Ashley. "Toward adding knowledge to learning algorithms for indexing legal cases". In: *Proceedings of the 7th international conference on Artificial intelligence and law*. ACM. 1999, pp. 9–17 (cit. on p. 6).

[Ch 73] L. Olbrechts-Tyteca Ch. Perelman. *New Rhetoric, The*. University of Notre Dame Press, Feb. 28, 1973. 576 pp. ISBN: 0268004463. URL: `https://www.ebook.de/de/product/3748337/ch_perelman_l_olbrechts_tyteca_new_rhetoric_the.html` (cit. on p. 14).

[CM10] Michael Curtotti and Eric Mccreath. "Corpus based classification of text in Australian contracts". In: *Proceedings of the Australasian Language Technology Association Workshop*. 2010 (cit. on p. 7).

[Dan05] Dr. Jörg Danger. "Urteil und Urteilsstil in der zivilrechtlichen Assessorklausur: eine praktische Hilfestellung". In: *Juristische Arbeitsblätter* 07 (2005) (cit. on pp. 1, 24, 25, 38).

[DR97] Jody J Daniels and Edwina L Rissland. "Finding legally relevant passages in case opinions". In: *Proceedings of the 6th international conference on Artificial intelligence and law*. 1997, pp. 39–46 (cit. on p. 6).

[EG+92] FH van Eemeren, R Grootendorst, et al. *Argumentation, communication, and fallacies: a pragma-dialectical perspective*. Hillsdale, NJ [etc.] Lawrence Erlbaum, 1992 (cit. on pp. 15, 16).

[Eem+14] FH van Eemeren et al. *Handbook of argumentation theory. A comprehensive overview of the state of the art*. Dordrecht: Springer, 2014 (cit. on pp. 10, 11, 15).

[Fog55] Béla Fogarasi. *LOGIK*. Aufbau-Verlag Berlin, 1955 (cit. on pp. 11–13).

[FP07] Enrico Francesconi and Andrea Passerini. "Automatic classification of provisions in legislative texts". In: *Artificial Intelligence and Law* 15.1 (2007), pp. 1–17 (cit. on p. 8).

[Gam00] Johann Gamper. "A Parallel Corpus of Italian/German Legal Texts." In: *LREC*. 2000 (cit. on p. 9).

[Gei08] Anton Geist. "The Open Revolution: Using Citation Analysis to Improve Legal Text Retrieval". In: *Eur. J. Legal Stud.* 2 (2008), p. 137 (cit. on p. 3).

[GSM18] Ingo Glaser, Elena Scepankova, and Florian Matthes. "Classifying Semantic Types of Legal Sentences: Portability of Machine Learning Models." In: *JURIX*. 2018, pp. 61–70 (cit. on p. 8).

[GQ03] Teresa Gonçalves and Paulo Quaresma. "A preliminary approach to the multilabel classification problem of Portuguese juridical documents". In: *Portuguese Conference on Artificial Intelligence*. Springer. 2003, pp. 435–444 (cit. on p. 8).

[GQ05] Teresa Gonçalves and Paulo Quaresma. "Is linguistic information relevant for the classification of legal texts?" In: *Proceedings of the 10th international conference on Artificial intelligence and law*. ACM. 2005, pp. 168–176 (cit. on p. 8).

[GW09] Thomas F Gordon and Douglas Walton. "Legal reasoning with argumentation schemes". In: *Proceedings of the 12th international conference on artificial intelligence and law*. 2009, pp. 137–146 (cit. on p. 19).

[HEG14] Ivan Habernal, Judith Eckle-Kohler, and Iryna Gurevych. "Argumentation Mining on the Web from Information Seeking Perspective." In: *ArgNLP*. 2014 (cit. on p. 18).

[HG05] Ben Hachey and Claire Grover. "Automatic legal text summarisation: experiments with summary structuring". In: *Proceedings of the 10th international conference on Artificial intelligence and law*. ACM. 2005, pp. 75–84 (cit. on p. 17).

[Hou+13] Constantin Houy et al. "Towards automated identification and analysis of argumentation structures in the decision corpus of the German Federal Constitutional Court". In: *2013 7th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*. IEEE. 2013, pp. 72–77 (cit. on p. 19).

[IK10] Kishore Varma Indukuri and P Radha Krishna. "Mining e-contract documents to classify clauses". In: *Proceedings of the Third Annual ACM Bangalore Conference*. ACM. 2010, p. 7 (cit. on p. 7).

[KBB17] Daniel Martin Katz, Michael J Bommarito II, and Josh Blackman. "A general approach for predicting the behavior of the Supreme Court of the United States". In: *PloS one* 12.4 (2017), e0174698 (cit. on p. 7).

[Kel13] David Kelley. *The art of reasoning: An introduction to logic and critical thinking*. WW Norton & Company, 2013 (cit. on p. 12).

[KFP19] Sonja Kind, Jan-Peter Ferdinand, and Kai Priesack. "Legal Tech–Potenziale und Wirkungen". In: (2019) (cit. on p. 3).

[Kio+14] George Kiomourtzis et al. "NOMAD: Linguistic Resources and Tools Aimed at Policy Formulation and Validation." In: *LREC*. 2014, pp. 3464–3470 (cit. on p. 7).

[Lan+16] Jörg Landthaler et al. "Extending Full Text Search for Legal Document Collections Using Word Embeddings." In: 2016 (cit. on p. 4).

[Law+12] John Lawrence et al. "AIFdb: Infrastructure for the Argument Web." In: *COMMA*. 2012, pp. 515–516 (cit. on p. 18).

[Lou+97] Ronald P Loui et al. "Progress on Room 5: A testbed for public interactive semi-formal legal argumentation". In: *Proceedings of the 6th international conference on Artificial intelligence and law*. 1997, pp. 207–214 (cit. on p. 6).

[MOC13] K Tamsin Maxwell, Jon Oberlander, and W Bruce Croft. "Feature-based selection of dependency paths in ad hoc information retrieval". In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2013, pp. 507–516 (cit. on p. 4).

[MS08] K Tamsin Maxwell and Burkhard Schafer. "Concept and Context in Legal Information Retrieval". In: *Proceedings of the 2008 conference on Legal Knowledge and Information Systems: JURIX 2008: The Twenty-First Annual Conference*. IOS Press. 2008, pp. 63–72 (cit. on p. 4).

[MF10] Eneldo Loza Mencía and Johannes Fürnkranz. "Efficient multilabel classification algorithms for large-scale problems in the legal domain". In: *Semantic Processing of Legal Texts*. Springer, 2010, pp. 192–215 (cit. on p. 7).

[MPG19] Jelena Mitrović, Adam Pease, and Michael Granitzer. "Modeling Legal Terminology in SUMO". In: *Proceedings of TOTh* (2019) (cit. on p. 55).

[Mit+17] Jelena Mitrović et al. "Ontological representations of rhetorical figures for argument mining". In: *Argument & Computation* 8.3 (2017), pp. 267–287 (cit. on p. 55).

[MI09] Raquel Mochales and Aagje Ieven. "Creating an argumentation corpus: do theories apply to real arguments? A case study on the legal argumentation of the ECHR". In: *Proceedings of the 12th international conference on artificial intelligence and law*. 2009, pp. 21–30 (cit. on pp. 19, 26, 28).

[MM11] Raquel Mochales and Marie-Francine Moens. "Argumentation mining". In: *Artificial Intelligence and Law* 19.1 (2011), pp. 1–22 (cit. on pp. 9, 17).

[Moe01] Marie-Francine Moens. "Innovative techniques for legal text retrieval". In: *Artificial Intelligence and Law* 9.1 (2001), pp. 29–57 (cit. on pp. 3, 4).

[MU97] Marie-Francine Moens and Caroline Uyttendaele. "Automatic text structuring and categorization as a first step in summarizing legal cases". In: *Information Processing & Management* 33.6 (1997), pp. 727–737 (cit. on p. 16).

[Moe+07] Marie-Francine Moens et al. "Automatic detection of arguments in legal texts". In: *Proceedings of the 11th international conference on Artificial intelligence and law*. ACM. 2007, pp. 225–230 (cit. on p. 19).

[Nei+17] James O' Neill et al. "Classifying sentential modality in legal language: a use case in financial regulations, acts and directives". In: *Proceedings of the 16th edition of the International Conference on Articial Intelligence and Law*. ACM. 2017, pp. 159–168 (cit. on p. 8).

[Nor02] Don Norman. "Emotion & design: attractive things work better". In: *interactions* 9.4 (2002), pp. 36–42 (cit. on p. 36).

[Oht+93] Y Ohtake et al. "HELIC-II: As a legal argumentation support system". In: *Proceedings of 9th IEEE Conference on Artificial Intelligence for Applications*. IEEE. 1993, p. 464 (cit. on p. 6).

[Ops+09] Rob Opsomer et al. "Exploiting Properties of Legislative Texts to Improve Classification Accuracy". In: *Legal Knowledge and Information Systems: JURIX 2009, the Twenty-second Annual Conference*. Vol. 205. IOS Press. 2009, p. 136 (cit. on p. 8).

[PM09] Raquel Mochales Palau and Marie-Francine Moens. "Argumentation mining: the detection, classification and structure of arguments in text". In: *Proceedings of the 12th international conference on artificial intelligence and law*. ACM. 2009, pp. 98–107 (cit. on pp. iv, 1, 2, 9, 10, 17–19, 27, 41, 42, 44, 45, 49).

[Ree06] Chris Reed. "Preliminary results from an argument corpus". In: *Linguistics in the twenty-first century* (2006), pp. 185–196 (cit. on p. 9).

[Ree+08] Chris Reed et al. "Language resources for studying argument". In: *Proceedings of the 6th conference on language resources and evaluation-LREC 2008*. ELRA. 2008, pp. 2613–2618 (cit. on p. 18).

[RA87] E. L. Rissland and K. D. Ashley. "A Case-Based System for Trade Secrets Law". In: *Proceedings of the 1st International Conference on Artificial Intelligence and Law*. ICAIL '87. Boston, Massachusetts, USA: Association for Computing Machinery, 1987, pp. 60–66. ISBN: 0897912306. DOI: 10.1145/41735.41743. URL: https://doi.org/10.1145/41735.41743 (cit. on pp. 5, 17).

[RS91] Edwina L Rissland and David B Skalak. "CABARET: rule interpretation in a hybrid architecture". In: *International journal of man-machine studies* 34.6 (1991), pp. 839–887 (cit. on p. 5).

[RSF96] Edwina L Rissland, David B Skalak, and M Timur Friedman. "BankXX: Supporting legal arguments through heuristic retrieval". In: *Artificial Intelligence and Law* 4.1 (1996), pp. 1–71 (cit. on p. 6).

[RP12] Camino Rea Rizzo and Mª José Marín Pérez. "Structure and design of the British Law Report Corpus (BLRC): a legal corpus of judicial decisions from the UK". In: *Journal of English Studies* 10 (2012), pp. 131–145 (cit. on p. 9).

[RTM01] Rema Rossini Favretti, Fabio Tamburini, and Enrico Martelli. "Words from Bononia Legal Corpus". In: *International journal of corpus linguistics* 6.1-12 (2001), pp. 13–34 (cit. on p. 9).

[SG07] Erich Schweighofer and Anton Geist. "Legal Query Expansion using Ontologies and Relevance Feedback". In: *Proceedings of LOAIT 07* (2007), p. 149 (cit. on p. 3).

[Sta17] Christian Matthias Edwin Stab. "Argumentative writing support by means of natural language processing". PhD thesis. Technische Universität Darmstadt, 2017 (cit. on p. 18).

[Sug+17] Keet Sugathadasa et al. "Synergistic Union of Word2Vec and Lexicon for Domain Specific Semantic Similarity". In: *IEEE International Conference on Industrial and Information Systems (ICIIS)* (2017) (cit. on p. 9).

[Teu99] Simone Teufel. "Argumentative zoning: Information extraction from scientific text". PhD thesis. Citeseer, 1999 (cit. on p. 16).

[Tho01] Paul Thompson. "Automatic categorization of case law". In: *Proceedings of the 8th international conference on Artificial intelligence and law*. ACM. 2001, pp. 70–77 (cit. on p. 7).

[Tou84] Stephen Toulmin. *An introduction to reasoning*. New York London: Macmillan Collier Macmillan Publishers, 1984. ISBN: 0024211605 (cit. on p. 13).

[UMG] Stefanie Urchs, Jelena Mitrović, and Michael Granitzer. "Towards Classifying Parts of German Legal Writing Styles in German Legal Judgments". In: () (cit. on p. 2).

[Wal+91] RF Walker et al. "PROLEXS: Creating law and order in a heterogeneous domain". In: *International journal of man-machine studies* 35.1 (1991), pp. 35–67 (cit. on p. 5).

[WBM] Bernhard Waltl, Georg Bonczek, and Florian Matthes. "Rule-based information extraction: advantages, limitations, and perspectives". In: () (cit. on p. 4).

[Wal+17a] BERNHARD Waltl et al. "Automated extraction of semantic information from german legal documents". In: *IRIS: Internationales Rechtsinformatik Symposium*. 2017 (cit. on p. 4).

[Wal+16] Bernhard Waltl et al. "LEXIA: A data science environment for Semantic analysis of german legal texts". In: *Jusletter IT* 4.1 (2016), pp. 4–1 (cit. on p. 7).

[Wal+17b] Bernhard Waltl et al. "Classifying Legal Norms with Active Machine Learning." In: *JURIX*. 2017, pp. 11–20 (cit. on p. 8).

[Wal+17c] Bernhard Waltl et al. "Predicting the outcome of appeal decisions in germany's tax law". In: *International Conference on Electronic Participation*. Springer. 2017, pp. 89–99 (cit. on p. 8).

[Wal+19] Bernhard Waltl et al. "Semantic types of legal norms in German laws: classification and analysis using local linear explanations". In: *Artificial Intelligence and Law* 27.1 (2019), pp. 43–71 (cit. on p. 8).

[WRM08] Douglas N. Walton, Chris Reed, and Fabrizio Macagno. *Argumentation Schemes*. Cambridge University Press, July 1, 2008. 456 pp. ISBN: 0521723744. URL: https://www.ebook.de/de/product/7597705/douglas_n_walton_chris_reed_fabrizio_macagno_argumentation_schemes.html (cit. on p. 16).

[Web18] Dirk Weber. *Methodik der Fallbearbeitung*. Nomos Verlagsges.MBH + Co, Mar. 1, 2018. 178 pp. ISBN: 3848744554. URL: https://www.ebook.de/de/product/29950244/dirk_weber_methodik_der_fallbearbeitung.html (cit. on pp. 1, 21–23).

[Wid02] Robin Widdison. "New Perspectives in Legal Information Retrieval". In: *Int'l JL & Info. Tech.* 10 (2002), p. 41 (cit. on pp. 3, 4).

[Wis+14] Lars Wissler et al. "The Gold Standard in Corpus Annotation." In: *IEEE GSC*. 2014 (cit. on p. 33).

[ZS95] John Zeleznikow and Andrew Stranieri. "The split-up system: integrating neural networks and rule-based reasoning in the legal domain". In: *Proceedings of the 5th international conference on Artificial intelligence and law*. 1995, pp. 185–194 (cit. on p. 6).

[ZVH93] John Zeleznikow, George Vossos, and Daniel Hunter. "The ikbals project: Multi-modal reasoning in legal knowledge based systems". In: *Artificial Intelligence and Law* 2.3 (1993), pp. 169–203 (cit. on p. 6).

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich diese Masterarbeit selbstständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe und alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, als solche gekennzeichnet sind, sowie, dass ich die Masterarbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt habe.

Passau, July 7, 2020

Stefanie Urchs