CEUR-WS.org/Vol-2738/LWDA2020_paper_29.pdf

Using Probabilistic Soft Logic to Improve Information Extraction in the Legal Domain

Birgit Kirsch[1], Sven Giesselbach[1], Timothée Schmude[1], Malte Völkening[3], Frauke Rostalski[3], and Stefan Rüping[1,2]

[1] Fraunhofer IAIS, Schloss Birlinghoven, Sankt Augustin
`<name>.<surname>@iais.fraunhofer.de`
[2] Fraunhofer Center for Machine Learning, Schloss Birlinghoven, Sankt Augustin
[3] Institute for Criminal Law and Criminal Procedure, University of Cologne, Cologne

Abstract. Extracting information from court process documents to populate a knowledge base produces data valuable to legal faculties, publishers and law firms. A challenge lies in the fact that the relevant information is interdependent and structured by numerous semantic constraints of the legal domain. Ignoring these dependencies leads to inferior solutions. Hence, the objective of this paper is to demonstrate how the extraction pipeline can be improved by the use of probabilistic soft logic rules that reflect both legal and linguistic knowledge. We propose a probabilistic rule model for the overall extraction pipeline, which enables to both map dependencies between local extraction models and to integrate additional domain knowledge in the form of logical constraints. We evaluate the performance of the model on a German court sentences corpus.

Keywords: Information Extraction · Probabilistic Soft Logic · Legal-Tech

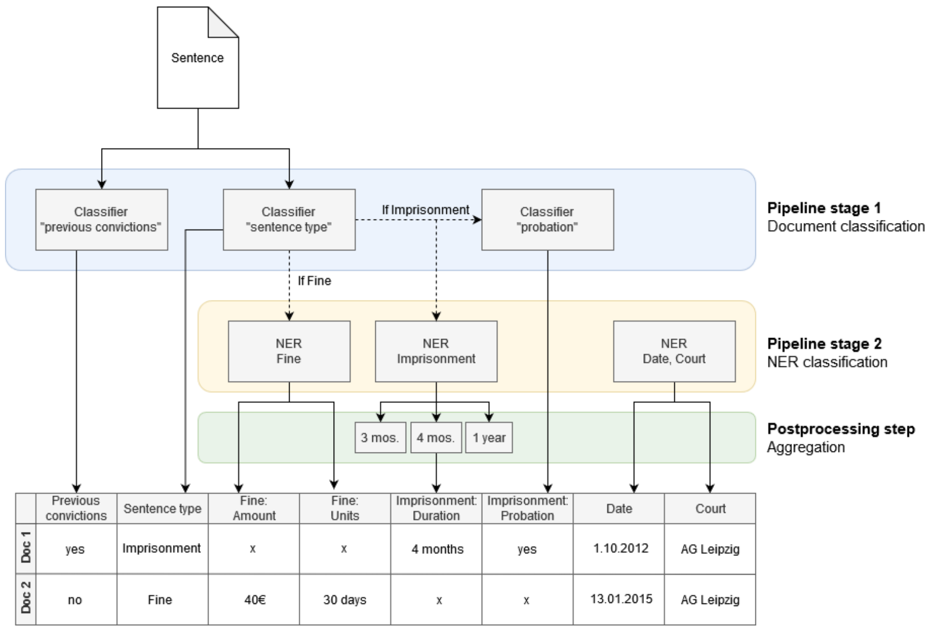
1 Introduction

In the year 2018 alone, there were approximately 870,000 court procedures in Germany[4]. All of them are documented in text, however to this date there are still only rudimentary solutions as to how to search for information within these documents. Transforming these documents into structured form produces data that can give insight into court processes and may be a valuable source to examine research questions such as the perceived imbalance in degrees of penalty between different regions in Germany, as described in Grundies (2018). According to the examinations of Grundies, there are substantial deviations (up to 15%) in how the same crime is punished in southern and northern Germany. The objective of this paper is to extract relevant information from these documents to populate a database that can be used for further analysis. While one

Copyright © 2020 by the paper's authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

[4] https://www.destatis.de/DE/Themen/Staat/Justiz-Rechtspflege/_inhalt.html

might think of using standard natural language understanding approaches to tackle this task, the particular challenge lies in the fact that the relevant information in these texts is interdependent and structured by complex legal domain knowledge. For example, the probability of occurrence of a monetary fine in the court sentence obviously depends on the type of court sentence (imprisonment or fine). Ignoring these dependencies leads to inferior solutions for information extraction. Hence, we demonstrate how the extraction of information from legal texts can be improved by the use of probabilistic soft logic rules that reflect legal knowledge.



	Previous convictions	Sentence type	Fine: Amount	Fine: Units	Imprisonment: Duration	Imprisonment: Probation	Date	Court
Doc 1	yes	Imprisonment	x	x	4 months	yes	1.10.2012	AG Leipzig
Doc 2	no	Fine	40€	30 days	x	x	13.01.2015	AG Leipzig

Fig. 1. Extraction process

Each document results in one entry in a knowledge base with seven facts about the document specific case and defendant, delineated in Section 3. Figure 1 displays the extraction pipeline to populate such a knowledge base and visualizes the dependencies between local components. It consists of the following two stages:

1. **Document Classification:** classification on document level predicts database entries with a fixed set of possible values, such as the type of court sentence (fine, imprisonment or acquittal).
2. **Named Entity Recognition (NER):** classification on token basis extracts all information with an unlimited set of possible values that have to be extracted from the document text itself, such as the duration of imprisonment. This will be treated as a named entity recognition task with the objective to assign a fixed set of classes to each token and extract the information

from the token text that is assigned the respective entity class. If a model assigns the same class to multiple text spans, they have to be consolidated to extract one value per document.

This baseline approach introduced for the use case raises multiple challenges: error propagation in the pipeline, value consolidation of NER predictions and the sparsity of training data.

Error propagation: One major disadvantage of this approach is that errors are propagated from the first classification component on document basis to the second classification component on token basis. This may lead to a lower performance with respect to the final aggregated document results. This problem is known from other NLP tasks such as knowledge base population. Recent work therefore focused on solving multiple tasks and modeling it as an end-to-end statistical inference problem (Sachan et al., 2018).

Value consolidation: One additional source of error is the consolidation of multiple token spans in the same document to extract one value per document. A NER classifier may in one document classify multiple token spans with the same class and as a consequence create multiple candidates that need to be consolidated to generate one valid value per NER class and document. Assigning the same NER class to multiple spans in one document is only valid when they refer to the same entity, e.g. the text span is similar. Integrating this as a hard constraint in a post-processing step may introduce additional errors.

Label sparsity: Since obtaining, anonymizing and labeling court sentence documents is a tedious process, the corpus available to train the pipeline classifiers is rather small. Beside the raw data there is valuable knowledge, both about dependencies between the information (e.g., a defendant with previous convictions is more likely to get imprisoned) and about the document structure (e.g., date and location of the court are likely to be in the beginning of the document). This sort of knowledge is not exploited in traditional approaches.

In order to address these challenges, we propose a probabilistic model constructed with a logical templating language that models a joint objective over the whole pipeline. By reasoning over all pipeline tasks jointly, we try to weaken the effect of propagated errors. By then modeling dependencies between NER candidates per document we address value consolidation challenges and with using a logical templating language, we enable to integrate additional background knowledge. Contributions of this paper can be summed up as the following:

- We propose a probabilistic model based on PSL for the overall extraction pipeline, which enables to map dependencies between local classification models.
- We model relevant concepts from the legal domain in the form of logical constraints and integrate them into the probabilistic model.
- We provide an empirical evaluation of the approach on a data set of court sentences and compare results with the traditional pipeline approach.

The remainder of this paper is organized as follows: Section 2 gives an overview of related work in the field of joint information extraction and probabilistic pipelines, Section 3 describes our approach, Section 4 provides an experimental evaluation on a German sentencing corpus and reviews benchmarks in comparison with a traditional pipeline approach, Section 5 concludes with a summary and future work.

2 Related Work

Modeling pipelines in NLP and handling error propagation is a well known problem in multiple fields of research. Former work proposed diverse approaches to tackle this, e.g. by using graphical models or inductive logic programming. Marciniak and Strube (2005) introduced a model built upon linear programming for NLP pipelines of cascading classifiers and Roth and Yih (2002) use a bayesian belief network for joint prediction for entity and relation classification models. Singh et al. (2013) apply a joint graphical model for the tasks entity tagging and relation extraction including co-reference resolution to allow flow of uncertainty across task boundaries. Besides the mere modeling of multiple tasks, former work also focused on incorporating on additional domain knowledge. Pawar et al. (2017) introduce a neural model to address boundary identification, entity type classification and relation type classification jointly (AWP-NN) and show that refining the output with a Markov Logic Network to incorporate additional knowledge improves the results. Min et al. (2017) propose a probabilistic graphical model to extract facts from documents end-to-end to fill a knowledge base. Besides taking into account full corpus information for joint inference, they empirically show that integrating knowledge about entity and relation occurrences improves the results. This work is based on Sachan et al. (2018), who propose to use the statistical relational learning framework PSL to model the pipeline in a probabilistic way. To our knowledge we are the first to introduce a rule based probabilistic model for the aforementioned NLP extraction pipeline that is able to incorporate additional domain knowledge and apply and evaluate it on a legal corpus. The concept of automatically extracting information from court procedure documents was proposed and developed as a prototype in the Legal Tech Lab Cologne[5] in early 2019 – an initiative, formed to find solutions for digitisation of legal procedures.

3 Approach

We propose a probabilistic model for an information extraction pipeline that jointly reasons over two pipeline stages: the **Document Classification Stage** and the **NER Stage**. This model is constructed using the logical templating language Probabilistic Soft Logic introduced by Bach et al. (2017). The following subsections explain both pipeline stages and the probabilistic model in detail.

[5] https://legaltechcologne.de/

3.1 Document Classification Stage

In this stage, all defendant and case-related information such as the type of court sentence is extracted. The objective of this stage is to perform three separate classification tasks and assign the following categories to each document:

- Previous Convictions (PC) : Yes/No
- Type of court sentence[6] (TS): Imprisonment/Fine/Acquittal
- Probation (PR): Yes/No

For each classification task, two separate model architectures are trained and applied, the inbuilt convolutional neural network (CNN) based model provided by *spaCy*[7], as well as a transformer based classification model, BERT (Devlin et al., 2018), which we will shortly discuss in the following subsections. The labels for the classification are assigned per court sentence in the documents.

BERT Classifier: Devlin et al. (2018) introduce a language model based on transformer networks (Vaswani et al., 2017), which has been shown to yield state-of-the-art performance in many natural language understanding tasks. We use the small German BERT model integrated in the hugging-face library[8]. Since BERT is restricted by memory constraints in the amount of tokens it can process and the document classes are available on a sentence level in the training set, indicating wether a sentence mentions a previous conviction (yes, no), a probation (yes, no) and the type of sentence, we train BERT to classify sentences instead of document classes. We train a separate classification model for each of the aforementioned categories and introduce the class *Other* for sentences which do not contain any class annotation. The values are aggregated to document classes. Therefore, we first filter out sentences in which the model predicts *Other* as the most likely class. Out of all remaining sentences we return the class probabilities for the sentence with the highest confidence of the classifier, i.e. where the maximum probability is the highest. If no sentences remain, we pick the highest probability for each of the classes, normalize and return the tuple of the new class probabilities.

spaCy Classifier: This architecture is based on a CNN with mean pooling and a final feed-forward layer. The network is fed with pretrained word embeddings trained on the German Wikipedia and the German common crawl (Ortiz Suárez et al., 2019).[9]

[6] Note that in the following it will be important to carefully distinguish between court sentences and sentences as a linguistic construct!

[7] An open-source library for Natural Language Processing, https://spacy.io/

[8] Model pretrained on the German Wikipedia, an online collection of legal court sentences and news texts, https://huggingface.co/bert-base-german-cased

[9] https://oscar-corpus.com/

3.2 Named Entity Recognition (NER) Stage

In the NER stage all document-class specific information such as amount of the fine have to be extracted from the document. Objective of the NER stage is to assign one of the following classes to each token in the document: date of the court sentence (date), court location (loc), amount of the fine (f_amnt), number of day-fines (f_units)[10] and duration of imprisonment (d_impr). Whether f_amnt and f_units or d_impr are present in a document depends on the type of court sentence (imprisonment or fine). For this classification task we rely on two architectures described below.

BERT NER: The named entity recognition model based on BERT has the same general transformer architecture as the classifier. Instead of predicting the class of sentences, a per-token classification takes place. We use a single feedforward layer with softmax activation to calculate the token labels.

spaCy NER: The spaCy NER model[11] utilizes a different architecture. Its embedding layer consists of two parts incorporating syntactic knowledge about the word, and a stack of 4 residual CNN layers to capture contextual information. Both layers are followed by a feed-forward network. Finally, a form of attention mechanism is used to incorporate additional information about the previous tokens and previous entity predictions. The final layer is a feed-forward and correction layer, which prevents illegal state shifts

3.3 Probabilistic Pipeline

We introduce a graphical model of a joint probability distribution to reason over all stage outputs at the same time. This model integrates both local stage model predictions, dependencies between the stages and additional background knowledge. For model construction we use the Statistical Relational Learning Framework PSL, introduced by Bach et al. (2017). It provides a First Order Logic templating language to define a joint probability distribution over a set of random variables. Therefore, rule templates are translated to a special type of Markov Random field, a *Hinge-Loss - Markov Random Field (HL-MRF)*. In this graphical model, each node represents a random variable and each edge a dependency between variables. Given a set of observed variables $X = (X_1, ..., X_n)$, a set of random variables $Y = (Y_1, ..., Y_{n'})$, a set of potential functions $\phi = (\phi_1, ..., \phi_m)$ and a set of weights $\omega = (\omega_1, ..., \omega_m)$, a *HL-MRF* represents the following probability density function over Y conditioned on X:

$$P(Y|X) = \frac{1}{Z(\omega, X)} \exp[-\sum_{j=1}^{m} \omega_j \phi_j(X, Y)] \quad (1)$$

[10] In German criminal law, fines are calculated in day-fines. The number of day-fines depends on the severeness of the offense while the amount of each day-fine is based on the offender's personal income.

[11] https://spacy.io/universe/project/video-spacys-ner-model

with Z as a normalization factor.

$$\phi_j(X,Y) = (\max\{l_j(X,Y),0\})^{p_j} \tag{2}$$

with l_j representing a linear function and $p_j \in \{1,2\}$.

Potential functions ϕ_j are defined per clique, a subset of fully connected nodes in the graph, assigning a probability mass to each clique state. A clique state is one assignment of values to all random variables participating in the clique. Assignment of a higher value to one clique state means that this state will be interpreted as being more likely. In *PSL*, these potential functions are generated using logical first order rules, such as:

$$w : Friends(A,B) \wedge Friends(B,C) \Rightarrow Friends(A,C) \tag{3}$$

A weight w is assigned to each rule and indicates its importance. $Friends$ is called a *predicate*. *Predicates* can take one to multiple arguments, such as A and B. Both A and B are *variables* and serve as placeholders. They can be substituted by concrete instances, referred to as *constants*. A *PSL*-model itself consists of a set of these template rules and a weight for each. Substituting all variables in the rule template set with their respective constants is called grounding. Every grounded rule represents one clique in the underlying graph structure and every grounded predicate is an observed or unobserved random variable mapped to a node in the clique. The weight of a grounded rule determines the weight ω_j of a potential function. A potential function assignment denotes the degree to which a rule is satisfied. Clique states that lead to satisfying a rule will be assigned a higher value than clique states that lead to the violation of a rule. Intuitively spoken, assignments of Y are more likely the fewer rules they violate.

Our proposed rule set can be categorized into four rule types: basic_rules, domain_rules, pipeline_rules and similarity_rules, examined in the following subsections. Table 1 provides a detailed explanation for all elements participating in each rule.

Basic Rule Set: The basic set models the relationship between the local classification models (reflected by the observed predicate $f\_cls\_pc^{(s1)}$ for stage one and $f\_cls^{(s2)}$ by stage two respectively) and the true unobserved stage prediction ($cls\_pc^{(s1)}$ for stage one and $cls^{(s2)}$ for stage two).

$$f\_cls\_pc^{(s1)}(d,m,c\_pc) \wedge T(m,c\_pc) \Rightarrow cls\_pc^{(s1)}(d,c\_pc)^2 \tag{4}$$

$$!f\_cls\_pc^{(s1)}(d,m,c\_pc) \wedge T(m,c\_pc) \Rightarrow !cls\_pc^{(s1)}(d,c\_pc)^2 \tag{5}$$

$$f\_cls^{(s2)}(d,z,m,c) \wedge T(m,c) \Rightarrow cls^{(s2)}(d,z,c)^2 \tag{6}$$

$$!f\_cls^{(s2)}(d,z,m,c) \wedge T(m,c) \Rightarrow !cls^{(s2)}(d,z,c)^2 \tag{7}$$

$$cls\_pc^{(s1)}(d,+c\_pc) = 1. \tag{8}$$

$$!cls^{(s2)}(d,z,c) \tag{9}$$

Domain	
$d = \{1, ..., D\}$	with D as the number of documents in the corpus
$m = \{SPACY, BERT\}$	local stage model
$z = \{1, ..., Z\}$	with Z as the number of token candidates
$c\_pc = \{yes, no\}$	class types predicted by $PreviousConviction$ classifier
$c\_st = \{Fine, Imprisonm., Acq.\}$	class types predicted by *courtsentencetype* classifier
$c\_pr = \{yes, no, other\}$	class types predicted by *probation* classifier
$c = \{date, ..., loc\}$	class types predicted by NER classifier
Observed Predicates	
$f\_cls\_pc^{(s1)}(d, m, c\_pc)$	prediction output of the local $PreviousConviction$ classifier model m for document d and class c_pc
$f\_cls\_st^{(s1)}(d, m, c\_st)$	prediction output of the local *courtsentencetype* classifier model m for document d and class c_st
$f\_cls\_pr^{(s1)}(d, m, c\_pr)$	prediction output of the local *probation* classifier model m for document d and class c_pr
$f\_cls^{(s2)}(d, z, m, c)$	prediction output of the local model m for document d, token candidate z and class c
$CLOSE(d, z1, z2)$	assigned 1, when there are at most 20 token between z1 and z2, 0 otherwise
$SIM(d, z1, z2)$	assigned 1, when the text z1 and z2 span over is equal, 0 otherwise
Unobserved Predicates	
$cls\_pc^{(s1)}(d, c\_pc)$	global prediction of stage one for document d and document class type c_pc
$cls\_ts^{(s1)}(d, c\_ts)$	global prediction of stage one for document d and document class type c_ts
$cls\_pr^{(s1)}(d, c\_pr)$	global prediction of stage one for document d and document class type c_pr
$cls^{(s2)}(d, z, c)$	global prediction of stage two for document d, candidate z and NER type c

Table 1. Variables and predicates participating in the rule model.

Rules 4 and 5 display the rules for the local classification models from stage one for $PreviousConvictions$. $T(m, c)$ is a trust score introduced by Sachan et al. (2018) that denotes the trustworthiness of a local model m when predicting a class c. Intuitively spoken, Rule 4 encodes that when a local model m predicts a class c_pc for a document d and the model is trustworthy, then the predicted unobserved class is more likely to be c_pc. A similar rule set exists for the class types $Typeofsentence$ and $Probation$. Rules 6 and 7 show the rules for the local NER classification models from stage two respectively. Rules 8 and 9 model the prior beliefs that the prediction probability for all class types of the PC-classifier sum up to one and that a token candidate z is not assigned the NER-class type.

Figure 2 displays a simplified model of the grounded observed (grey nodes) and unobserved predicates (white nodes) according to the above rule sets. The model is grounded only for one class (c1_pv, c1_pr) of the document classifiers pc and pr, for one document $d1$, two local stage one models $m1$ and $m2$, two local stage two models $m3$ and $m4$ to predict one class $c3$ and two NER candidates

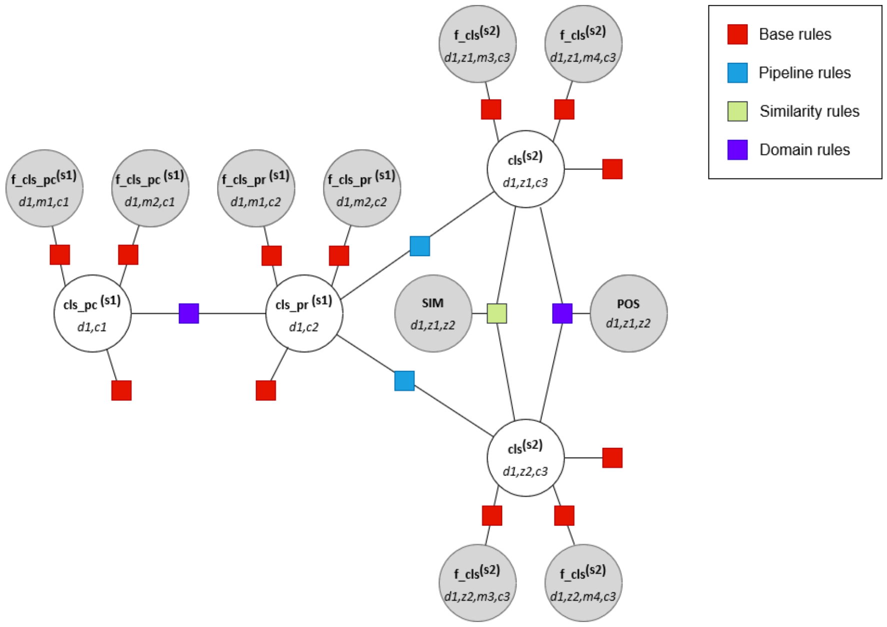


Fig. 2. Simplified grounding for rule model

$z1$ and $z2$ for which class $c3$ should be predicted. Potential functions resulting from the baseline rules are presented as red squares.

Pipeline Rule Set: This rule set incorporates relationships between prediction outputs of the document and NER classification stages and relationships between local models from one stage. Rule 10 for example expresses the dependency between the *st* classifier and the NER classifier. Whenever the predicted value for court sentence type is not "fine", it is unlikely that the information about the amount of the fine is present in the document

Rule 11 models the dependency between the PR and the ST classifiers for of the first stage.

$$!cls\_st^{(s1)}(d,'fine') \Rightarrow !cls^{(s2)}(d,z,'f\_amount')^2 \tag{10}$$

$$cls\_pr^{(s1)}(d,'no') \Rightarrow cls\_st^{(s1)}(d,'imprisonment')^2 \tag{11}$$

Domain Rule Set: The domain rules permit to incorporate knowledge about the document structure (e.g. in Rule 12) and class dependencies such as that defendants with previous convictions are more likely to be sentenced to imprisonment or that the height of the fine and the number of day-fines are most likely mentioned in the same paragraph (see Rule 13).

$$cls\_pc^{(s1)}(d,'yes') \Rightarrow cls\_st^{(s1)}(d,'imprisonment')^2 \tag{12}$$

$$cls^{(s2)}(d,z1,'f\_amnt') \wedge !CLOSE(d,z1,z2) \Rightarrow cls^{(s2)}(d,z1,'f\_units')^2 \tag{13}$$

Similarity Rule Set: The similarity rules model the relationship between multiple mentions of the same entity in one document. Rule 14 denotes that when two candidates in one document span over similar strings, they are more likely to be assigned the same NER type. This ensures that when one information like the date of the court sentence is mentioned multiple times in the document it is always assigned the same NER class type and when two candidates don't have equal content, they are unlikely to refer to the same entity (Rule 15).

$$cls^{(s2)}(d, z1, c) \wedge SIM(d, z1, z2) \Rightarrow cls^{(s2)}(d, z2, c)^2 \tag{14}$$

$$cls^{(s2)}(d, z1, c) \wedge !SIM(d, z1, z2) \Rightarrow !cls^{(s2)}(d, z2, c)^2 \tag{15}$$

4 Evaluation

Data: We evaluate the performance of the model introduced in Section 3 using a corpus with 146 German court sentences[12]. Each document describes an offence (theft), where the defendant is either sentenced to imprisonment or to a fine, and which is only related to one case and one defendant. The documents are split into training, validation and test set.

PSL Pipeline Model: For the probabilistic model we train local stage classifiers for document and token classification using the architectures described in Section 3. All classifiers are trained on the training set and applied to the validation and test set. We use the validation set to estimate the trustworthiness scores (Rules 4, 5, 6, 7) and to refine weights. To evaluate the performance of the rule model and its subsets, we do not provide an evaluation for the local stage tasks (document and token classification), but focus on the amount of correctly extracted values per document. We perform inference for the unobserved variables listed in Table 1 to predict all layer outputs at the same time. As proposed in Bach et al. (2017), we use the consensus optimization approach Alternating Direction Method of Multipliers (ADMM).

Basic Pipeline Model: We compare the proposed solution to the pipeline approach visualized in Figure 1, where all local models are trained using the BERT architecture (see Section 3). Since the models that were trained using the architecture of spaCy had a poor performance on the test set, the performance of the pipeline model based on these classifiers is not evaluated here. The local document classification models are trained on both training and validation set and are applied to the test set to predict document class labels. The local token classification models are trained both on training and validation set and are then applied to the test set. The NER classification results per document are consolidated by selecting the candidate with the highest probability score predicted by the model for each NER class. We model dependencies between the

[12] manually annotated by the University of Cologne

predicted type of sentence and the NER prediction as hard constraints. When the predicted class is "Fine", prediction for "f_amnt" and "f_units" are set to "NaN" (for document class "Imprisonment", prediction for "d_impr" is set to "NaN" respectively). We additionally provide results for the extraction workflow where this stage dependency is not modeled and contradicting results are not removed. Table 2 provides a comparison of model performances. The first seven entries denote the performance of the rule models consisting of different rule subset: basic (basic rules), sim (similarity rules), pipe (pipeline rules) and domain (domain rules). The scores per extracted information type reflect the amount of correctly extracted values on the test corpus.

Model	**Date**	**Loc**	**f_amnt**	**f_units**	**d_impr**	**pc**	**st**	**pr**	**avg.**
PSL: basic sim domain	0.783	**0.913**	0.826	0.826	0.870	0.870	0.913	0.783	0.848
PSL: basic sim	0.783	**0.913**	0.826	0.826	0.870	0.826	**0.957**	0.783	0.848
PSL: basic domain	0.783	**0.913**	0.826	0.870	0.826	0.870	0.913	0.783	0.848
PSL: basic pipe	0.783	**0.913**	0.826	0.783	**0.957**	0.826	0.913	**0.913**	0.864
PSL: basic sim pipe	0.783	**0.913**	0.826	0.783	**0.957**	0.826	0.913	**0.913**	0.864
PSL: basic sim pipe domain	0.783	**0.913**	**0.870**	**0.870**	0.913	**0.913**	0.913	**0.913**	**0.886**
PSL: basic pipe domain	0.783	**0.913**	**0.870**	**0.870**	0.913	**0.913**	0.913	**0.913**	**0.886**
BERT: constraints	**0.826**	0.870	**0.870**	0.826	**0.957**	0.826	**0.957**	0.783	0.864
BERT: no constraints	**0.826**	0.870	0.826	0.783	**0.957**	0.826	**0.957**	0.783	0.853

Table 2. Extraction performance of rule models

The two PSL models achieving the highest overall matching rate both contain basic, pipeline and domain rules. Modeling relations between NER candidates by integrating the similarity rules does not seem to have an influence on the performance in this experiment. On average over all information types, the highest scoring PSL model outperforms both BERT benchmark models. For some particular types ($Date$, d_impr and st) the BERT benchmark model achieves a higher matching rate. These results indicate that modeling the pipeline in a probabilistic way and integrating domain knowledge improves the overall performance. Since the data set is very small, further analysis on a bigger corpus and a more detailed analysis of the rule model effects are required.

5 Summary

This paper proposes a rule based probabilistic model to improve the extraction pipeline for information extraction from court sentence documents. The model enables to both map dependencies between local extraction components and to integrate additional domain knowledge in the form of logical constraints. We evaluate the performance of the model on a German court sentences corpus and show that the model improves results compared to a BERT benchmark model.

6 Acknowledgements

This research has been funded by the Federal Ministry of Education and Research of Germany as part of the competence center for machine learning ML2R (01IS18038B)

Bibliography

Bach, S.H., Broecheler, M., Huang, B., Getoor, L.: Hinge-loss markov random fields and probabilistic soft logic. J. Mach. Learn. Res. 18(1), 3846–3912 (2017)

Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. CoRR abs/1810.04805 (2018)

Grundies, V.: Regionale unterschiede in der gerichtlichen sanktionspraxisin der bundesrepublik deutschland. eine empirische analyse. In: Kriminalsoziologie. Handbuch für Wissenschaft und Praxis. pp. 295–316. Baden-Baden, Germany (2018)

Marciniak, T., Strube, M.: Beyond the pipeline: Discrete optimization in NLP. In: Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005). pp. 136–143. Association for Computational Linguistics, Ann Arbor, Michigan (Jun 2005)

Min, B., Freedman, M., Meltzer, T.: Probabilistic Inference for Cold Start Knowledge Base Population with Prior World Knowledge 1, 601–612 (2017)

Ortiz Suárez, P.J., Sagot, B., Romary, L.: Asynchronous Pipeline for Processing Huge Corpora on Medium to Low Resource Infrastructures. In: Bański, P., Barbaresi, A., Biber, H., Breiteneder, E., Clematide, S., Kupietz, M., Lüngen, H., Iliadi, C. (eds.) 7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7). Leibniz-Institut für Deutsche Sprache, Cardiff, United Kingdom (Jul 2019), https://hal.inria.fr/hal-02148693

Pawar, S., Bhattacharyya, P., Palshikar, G.: End-to-end relation extraction using neural networks and Markov logic networks. In: Proceedings of the 15th Conference of EACL: Volume 1, Long Papers. pp. 818–827. Association for Computational Linguistics, Valencia, Spain (Apr 2017)

Roth, D., Yih, W.t.: Probabilistic reasoning for entity & relation recognition. In: COLING 2002: 19th Intern. Conf. on Computational Linguistics (2002)

Sachan, M., Dubey, K.A., Mitchell, T.M., Roth, D., Xing, E.P.: Learning pipelines with limited data and domain knowledge: A study in parsing physics problems. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in NIPS 31, pp. 140–151. Curran Associates, Inc. (2018)

Singh, S., Riedel, S., Martin, B., Zheng, J., McCallum, A.: Joint inference of entities, relations, and coreference. In: Proceedings of the 2013 Workshop on AKBC. p. 1–6. AKBC '13, Association for Computing Machinery, New York, NY, USA (2013), https://doi.org/10.1145/2509558.2509559

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in NIPS 30, pp. 5998–6008. Curran Associates, Inc. (2017)