Extracting structured knowledge from Dutch legal texts: a rule-based approach

Bakker, R.M.; Boer, M.H.T. de; Drie, R.A.N. van; Vos, D.

Extracting Structured Knowledge from Dutch Legal Texts: A Rule-based Approach

Roos M. Bakker[1,2,*], Maaike H.T. de Boer[1], Romy A.N. van Drie[1] and Daan Vos[1]

[1]*Nederlandse Organisatie voor Toegepast Natuurwetenschappelijk Onderzoek (TNO), Anna van Buerenplein 1, Den Haag, 2595DA, the Netherlands*

[2]*Universiteit Leiden, Reuvensplaats 3, Leiden, 2311BE, the Netherlands*

Abstract

Legal texts are difficult to interpret, and its interpretation depends on the knowledge and experience of the legal expert. Formalising interpretations can improve transparency. However, creating formalisations of legal texts is labour-intensive, and automatically creating them is still a challenge. Previous work showed that rule-based systems have mixed success on Dutch legal texts. They use complex rule systems for specific cases, making them hard to compare. Because of the lack of analysis, the success of these methods is also unclear. In this paper, we propose a new rule-based architecture for detecting the different roles of Flint frames, a knowledge representation language which aims to be a generic and less task-dependent language. The rules in this architecture are based on Part-of-Speech tags and universal dependency tags. Our analysis shows that this combination yields more precise extraction of the roles of Flint frames than previous methods, and the use of universal dependency tags allows this method to also be applied to other languages. For further improvement we suggest extending the rules for extracting the recipient role, add rules for recognising complex relative clauses, and testing this framework on English legal texts.

Keywords

Information Extraction, Knowledge Modelling, Legal Interpretation Support

1. Introduction

Automatically translating sources of norms, a name that covers a wide variety of legal text, including legislation, policy guidelines, contracts and doctrine, into formal and computer executable interpretations is considered to be challenging. Experts in Natural Language Processing (NLP) have been working on this topic since the early 2000s, using a wide variety of NLP techniques to 'translate' these sources of norms into (semi) formal knowledge representation languages (KR) [1, 2, 3]. A recent language is Flint [4]. Flint is aimed to be a more generic and less task-dependent modeling language that can be used to express the interpretation of any source of norms. One of the important considerations in Flint is that the perspectives

EKAW'22: Companion Proceedings of the 23rd International Conference on Knowledge Engineering and Knowledge Management, September 26–29, 2022, Bozen-Bolzano, IT

*Corresponding author.

roos.bakker@tno.nl (R. M. Bakker); maaike.deboer@tno.nl (M. H.T. d. Boer); romy.vandrie@tno.nl (R. A.N. v. Drie); daan.vos@tno.nl (D. Vos)

https://gitlab.com/calculemus-flint/flintfillers (R. M. Bakker)

0000-0002-1760-2740 (R. M. Bakker); 0000-0002-2775-8351 (M. H.T. d. Boer)

of all agent roles are included, as well as that the focus is on the action part of norms. As a result, the KR can be used in different task contexts using multiple forms of reasoning. We may, for instance, use the KR as a knowledge base for multidisciplinary teams, as specification for decision-support systems, to detect anomalies such as conflicts in duties, or to build eServices assisting citizens.

In this paper, we propose a new knowledge driven approach, inspired by previous work from De Maat et al. [3] and Bakker et al. [5]. We use part-of-speech (POS) and dependency tagging, and then allocate parts of the sentence to the slots in a Flint frame using a new set of rules. We use universal dependency tags such that our method will be transferable to other languages. Our approach differs from earlier work in that 1) it can deal effectively with passive sentences; 2) it uses the latest dependency parsing techniques; and 3) that it is applicable on flint frames. We hypothesise that our approach will improve extraction of the frames and will allow for a combination of the methods used in Bakker et al. [5].

2. Related Work

Relevant work on extracting information from legal texts using NLP has been done for different languages, such as Italian [1, 6] and English [2, 7]. In this section, we focus on previous work on Dutch legal texts. In 2.1 we discuss previous work that improves modeling of Dutch legal texts using NLP techniques. We use Flint frames as a basis for the extracted information, they will be discussed in 2.2.

2.1. Rule-based systems for Dutch

In the previous decade, NLP research within the Dutch legal text domain has mainly focused on syntactic and semantic analysis. One of the first papers is that of Van Gog et al. [8] from 2001. In that paper a tool named OPAL (Object-oriented Parsing and Analysis of Legislation) is proposed. The goal of this tool is to support the modeling of legislation using noun phrases and specific patterns in legal sentences. Unified Modeling Language (UML) and Object Constraint Language (OCL) are used as languages.

Another paper in that decade uses Web Ontology Language (OWL) as the knowledge representation language [3]. The goal of this paper by de Maat et al. is to extract norms, which have specific subtypes: obligations, rights, application provisions, penalisations, calculations, delegations and publication provisions. Each sentence of the Dutch law texts is first classified to one of the subtypes using a pattern matcher and an Machine Learning (ML) classifier. After classification, frames are extracted using rules and transformed to OWL. These rules are based on extracted dependencies using Alpino [9]. An example of a rule is that the subject is the agent of the action.

Recently, Bakker et al. [5] propose to view the formalisation of a part of the law as a Semantic Role Labelling task. They compare performance between a transformer-based model based on the Dutch BERTje [10] and a rule-based method. The results show that the transformer-based method outperforms the rule-based method on both a small annotated data set with Dutch law text and on the Dutch Aliens Act. In the discussion it is mentioned that the rule-based method

Table 1
Action role definitions with simplified example of a manually created act frame from the GDPR

Role	Definition	example
Act	Name of the act frame	collect personal data
Action	Action that causes the transition of an object	collect
Actor	Agent role that is allowed to perform action	processor
Object	The object acted upon	personal data
Recipient	Agent role having a normative relation with the actor concerning his action	data subject
Precondition	Set of conditions that must be met to allow the action of the actor	personal data are processed lawfully, fairly and in a transparent manner in relation to the data subject
Creating postcondition	Facts or normative relations created by action of the actor	controller shall be able to demonstrate compliance with Art. 5(1) GDPR
Terminating postcondition	Facts or normative relations terminated by action of the actor	-
Source	Reference to the source of the act type, including information on version	Art. 5 (1) GDPR

is not optimised yet, as only a limited number and analysis of the rules is available. The authors also mention Flint as the formalism in which the semantic roles could be used.

2.2. Flint Frames

Flint [4] is a framework to represent sources of norms. Flint is aimed to be a generic and less task-dependent modeling language that can be used to express the interpretation of any source of norms. The novelty of this framework is that the focus is on the action part of norms and that the perspectives of all agent roles affected by the norms are included. As a result, the KR can be used in multiple task contexts using multiple forms of reasoning.

Flint is based on three types of frames to express the interpretation of sources of norms: acts, facts and duties, as described in [4]. In this paper we focus on act frames, as they are the core of Flint. An overview of the roles with a definition and example are shown in table 1. The action is the core of the act frame, and is accompanied by an actor that performs the action, the object that is acted upon and the recipient that has a relation with the actor concerning the action. Before the action can be performed there are preconditions, and after the action is performed there are post-conditions that can be created or terminated. Metadata such as the name of the act and the original text source are also included in the act frame.

3. Problem Statement

Dutch legal texts are not representative for normal written Dutch. The sentences are longer, they contain complex adverbial clauses, and are mostly written in passive voice. The following

sentence is a typical example: *A return visa can be refused by Our Minister if the foreign national has not demonstrated by submitting documents that there is an urgent reason that does not allow postponement of departure.*[1] If we want to formalise such a sentence to a norm, in our case a flint frame, we notice that the sentence is in passive voice, it has a large relative clause, and it contains information of which it is unclear how it fits into the frame.

Previous work on extracting information from Dutch legal texts shows different approaches on tackling these challenges using language patterns and rules. A downside of many of these approaches such as van Gog and van Engers [8], De Maat et al. [3], Bakker et al. [5] is that they use different KR languages. [8] use UML and OCL, [3] use types of norm frames that are translated into OWL, and [5] use Flint. This makes the methods harder to compare and apply in a broader context.

What previous approaches do have in common is that they make use of tagging, [8] and [5] use POS-tags, [3] use dependency tags. The sentences are tagged manually in [8], and automatically in [3, 5] by using Alpino [9] and Pattern [11] respectively. The rules in these methods are based on the tags. For instance, [5] create the actor of an act frame from the first Noun Phrase in the sentence, whereas [3] create the actor from the dependency tag subj (subject). [8] use a different KR language, but take a similar approach where noun phrases are stored as individuals in an ontology-like model. At the time of their work, in 2001, treebanks for automatic tagging such as [3] and [5] use were not available yet. As [5] mention, POS-tags do not contain enough information to capture the complex structure of sentences in legal texts. Their results show that the performance of their rules was very low, especially on the recipient role in the flint frame. The cause was too simplistic rules and wrong tags. [3] showed in 2009 that they can correctly tag complex active sentences using dependency tags. For passive sentences however, more complex operations are needed and the performance remains unclear. It has to be noted here that since then, the tagging packages and databases for Dutch have improved. For [8], analysis of their rules and the performance is not present. Analysis of the rules and the performance on legal text is sparse in all cases. [5] show an accuracy performance, but an analysis of the rules and why they don't work properly is missing. [3] do a short analysis, but only on one example sentence which is not typical for legal texts.

4. Proposed Solution

In this work, we choose the Flint language for formalisation because of it's aim to be a generic and less task-dependent modeling language. This way, our work can also be translated to other types of texts. To improve and solve the issues stated above, we propose a new architecture and set of rules.[2] The architecture is shown in figure 1 and consists of three steps. First, the law texts are preprocessed. To make them suitable for tagging, they are split into the smallest element of the law, often a sentence. For Dutch law source texts, they also need to be extracted from the xml form in which they are documented. Second, the sentences are POS- and dependency

[1] translated, orginal sentence from de Vreemdelingenwet, artikel 2x-1a: Een terugkeervisum kan worden geweigerd door Onze Minister indien de vreemdeling niet door overlegging van documenten aannemelijk heeft gemaakt dat sprake is van een dringende reden die geen uitstel van vertrek mogelijk maakt.

[2] The implemented pipeline and rules of this architecture can be found here: https://gitlab.com/calculemus-flint/flintfillers/flintfiller-rlb

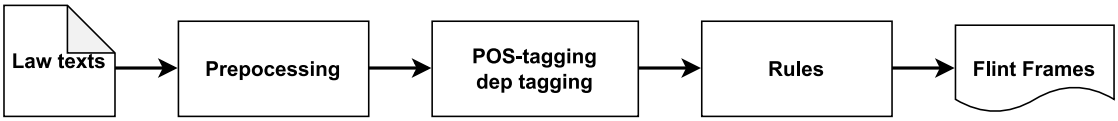

Figure 1: The architecture to extract flint frames based on three different steps: preprocessing, tagging, and rules.

tagged. Third, rules are applied to extract the relevant roles to build flint frames. This pipeline is similar to the work in [5], and we also focus on the main type of Flint frame, the act frame. The difference is that we combine universal dependency tags with POS-tags, and propose more detailed rules to recognise the roles of an act frame in the Flint language. We hypothesise that the combination of POS-tags and dependency tags will allow more detailed rules, which in turn will produce better quality flint frames.

4.1. Part-of-speech Tags and Dependency Tags

Part-of-speech (POS) and dependency tagging was one of the first modern NLP techniques. POS-tagging tags grammatical parts of the sentence, not unlike the parsing of a sentence that children learn in elementary school. An example can be seen in table 2[3]. We tag the sentences automatically using spaCy [12], which uses both Alpino and Lassysmall [9, 13], the main treebanks for Dutch. These libraries are up-to-date and offer improved tagging compared to taggers used in previous work such as [3, 5]. Below we describe the dependency tags that we use in our rules, an overview of the complete set and of the POS-tags can be found on the universal dependencies website[4].

Table 2
Example tagged sentence

Example sentence	Our Minister may grant an exemption or dispensation from the first and second paragraphs to the alien.
POS tags	['NP', ('Our', 'PRP$'), ('Minister', 'NNP')]['VP', ('may', 'MD'), ('grant', 'VB')]['NP', ('an', 'DT'), ('exemption', 'NN'), ('or', 'CC'), ('dispensation', 'NN')]['PP', ('from', 'IN')]['NP', ('the', 'DT'), ('first', 'JJ'), ('and', 'CC'), ('second', 'JJ'), ('paragraphs', 'NNS')]['NP', ('the', 'DT'), ('alien', 'NN')]
Dep tags	[('Our', 'nmod:poss'), ('Minister', 'nsubj'), ('may', 'aux'), ('grant', 'root'), ('an', 'det'), ('exemption', 'obj'), ('or', 'cc'), ('dispensation', 'conj'), ('from', 'case'), ('the', 'det'), ('first', 'amod'), ('and', 'cc'), ('second', 'conj'), ('paragraphs', 'obl'), ('to', 'case'), ('the', 'det'), ('alien', 'nmod'), ('.', 'punct')]

We compared the definitions of the roles of flint frames to the definitions of the different dependency tags and part of speech tags. As [3] described, the dependency tags offer more

[3]All examples in this paper are translated from the Dutch Aliens Act, because of this small differences may occur in the tagging and the frames compared to the original Dutch ones.
[4]https://universaldependencies.org/nl/index.html

semantic insight in the words. Another advantage is that by using universal dependency tags, the method will be transferable to other languages. The POS tags are useful for chunking the information and for elimination of adverbial clauses. The dependency tags that comply with the actor are *nsubj* for active sentences, which is the nominal subject of finite sentences. For actors in the passive sentences the dependency tag is *obl:agent*, which is used for prepositional arguments and adjuncts of a verbal head and for the door-phrase that can be present in passives. The tags for the object are *obj* for active sentences: the direct object of verbal heads, and *nsubj:pass* for passive sentences, the subject of passive sentences. The tag for a recipient is *iobj*: indirect objects that are not introduced by a preposition. Finally, actions can be recognised by *root*, the root or main verb, *ccomp* which are complement clauses that are dependents of a verb, and *xcomp*: head of non-finite verbal complements of verb and redicative complements of non-copula verbs.

4.2. Rules

After the sentences are tagged with POS-tags and dependency tags, we extract the roles of the flint frames by formulating rules. An overview of the rules is given in pseudocode in algorithm 1. The rules are applied per sentence in the legal text. We first form phrases of the sentence by using the POS-tags for creating chunks. For instance, in the example shown in table 2 the first phrase, a noun phrase, is 'Our Minister'. This way, a role can be multiple words. The first rule states that if a token in a phrase (a word in for instance a noun phrase) is labeled with *nsubj* or *obl:agent*, the complete phrase will be the actor role of a flint frame. In the example sentence 'Our Minister may grant an exemption or dispensation from the first and second paragraphs to the alien' in 2, 'Minister' is labeled *nsubj*, and part of the noun phrase 'Our Minister'. The second rule determines that a phrase is a object when a token in that phrase is labeled *obj* of *nsubj:pass*. In our example, 'exemption or dispensation' is labeled *obj* and together forms a noun phrase, so this part of the sentence will be the object. The third rule determines the recipient, if a token is labeled *iobj*, the phrase will be the recipient. In our example this tag does not occur. Finally, to get the action phrase, which often consists of multiple verbs, we take all the phrases with tokens labeled as root, *ccomp*, or *xcomp*, and they will form the action of the act frame. In the final step of the architecture, the roles resulting from these rules are assembled and put together in a flint frame format, of which we will see more examples in the next section.

5. Analysis and Insights

To get insights in the performance of the architecture and the rule set, we created act frames using the tags and the rules, and compared them to manually created act frames. We chose the Alien Act (*Vreemdelingenwet*) as a use case. We saw that actors, objects, and actions are recognised correctly in most cases. However, recipients were only recognised correctly in a few cases. This is best illustrated with some examples. For instance, the following active voice sentence contains all roles: 'Our minister may grant an exemption or dispensation from the first and second paragraphs to the alien[5].' A domain expert created a manual act from this sentence,

[5]Translated from the Dutch Aliens Act, article 2y-3

Algorithm 1 Rules expressed in pseudocode

```
for phrase in sentence do                                   ▷ e.g. 'Our Minister'
    for token in phrase do                                  ▷ e.g. 'Minister'
        if token = nsubj or obl : agent then
            phrase ← actor                                  ▷ e.g. 'Our Minister' -> actor
        else if token = obj or nsubj : pass then
            phrase ← object                                 ▷ e.g. 'exemption or dispensation' -> object
        else if token = iobj then
            phrase ← recipient      else if token =rootorccomporxcomp then
             phrase ← action                                ▷ e.g. 'may grant' -> action
         end if
      end for
  end for
```

and we compared this to the automatic act. The resulting frames can be seen in table 5, as well as two other examples. All roles are correctly detected except the recipient. The minor changes in the actor and the object are caused by the domain expert adding their contextual knowledge, this information is not in the sentence. The recipient is not recognised correctly because the dependency tag for alien was *nmod*. We also tagged the English version of this sentence to check if the tagger algorithm was at fault. In English the token was tagged as *obl*, which is also not a recipient according to our rules. In the second example in 5 the frames are almost identical, and a recipient is found. The only mistake here is that 'him' refers back to the foreign national in the first part of the sentence, this cannot be derived with our rules.

Finally, the third example shows an error in the action. This is due to multiple verb phrases in the sentence, one in the first part, and one in the conditional clause. This conditional clause causes also problems with the actor and the object. For this sentence, the domain expert created the frame only from the first part of the sentence, whereas the rules extract roles from the complete sentence.

In our analysis we found few recipients, so we tested with very simple sentences when the *iobj* tag occurs at all in Dutch. For simple sentences as 'I give her a gift', her was correctly tagged with the *iobj* tag, but when we changed it to 'I grant her a wish', her is tagged as *expl:pv*. This indicates that the recipient role requires more analysis and possibly more complex rules to be extracted correctly.

From our first analysis, we can conclude that actors, actions, and objects can be recognised in active and passive sentences with our rule set. Improvements need to be made for recipients, and for dealing with long sentences with relative clauses. Finally, implicit roles might be added by a domain expert, but to add these automatically would be a very hard task for which you would have to look at the complete law text. Further insights may be gathered from more extensive experimentation and quantitative evaluation which we did not do due to time constraints. An interesting experimental setup would be to manually label a set of legal texts with the roles of the flint frames, and to compare these to the roles assigned automatically with our method.

Table 3
Example act frames created manually and automatically

frame	Manual act	Automatic act
action	may grant	may grant
actor	Our Minister of Justice and Security	Our Minister
object	exemption for exceeding the period of validity of the return visa	exemption or dispensation
recipient	alien	
source text		Our Minister may grant an exemption or dispensation from the first and second paragraphs to the alien.
frame	Manual act	Automatic act
action	assigned	assigned
actor	Our Minister	Our Minister
object	a counselor	a counselor
recipient	the foreign national	him
source text		At the request of the foreign national, a counselor is assigned to him by Our Minister.
frame	Manual act	Automatic act
action	refused	refused allow
actor	Our Minister	Our Minister urgent reason
object	A return visa	a return visa postponement
recipient		
source text		A return visa can be refused by Our Minister if the foreign national has not demonstrated that there is an urgent reason that does not allow postponement of departure.

6. Conclusion and Discussion

Legal texts are difficult to interpret, formalising them can improve transparency. However, creating formalisations of legal texts is labour-intensive, and automatically creating them is still a challenge. Previous work showed that rule-based systems have mixed success on Dutch legal texts [8, 3, 5]. This has multiple reasons. First of all, they focus on different types of formalisation languages, making them hard to apply in a broader context and to compare the performance. Second, they use outdated POS- or dependency tags [8, 3], or use only POS tags which do not express enough information such as [5]. Third, proper analysis of the rules is missing. Therefore, we propose a new rule-based architecture for detecting the different roles of Flint frames. We chose the Flint language for formalisation because of it's aim to be a generic and less task-dependent modeling language. This way, our work can also be translated to other types of texts. Our architecture has three steps: preprocessing the text, tagging the sentences with POS- and universal dependency tags, and finally applying rules. The rules are all of the form *if* a word (token) has a certain dependency tag *then* the complete phrase will be marked as one of the roles in the flint frame. The phrase is recognisable by the POS-tags. Finally, for

each sentence that contains an action, a flint frame is build with the roles that were found in this sentence. In the analysis we saw that the roles actor, object, and action are correctly recognised in most cases. However, the recipient often was missing, even though it occurred in a frame created manually by a domain expert of the same sentence. This had several causes: The legal sentences were too complex, the recipient is often tagged with the broad dependency tag obl, and sometimes the domain expert would add roles that were not explicitly present in the sentence based on their contextual knowledge. We also found that complex relative clauses cause multiple roles to appear in one frame with the current rule set. We can conclude that our proposed architecture improves the extraction of all but one roles in the flint frame. Because of the combination of POS- and dependency tags, it is more precise than previous methods. It is also applicable to other languages because of the use of universal dependency tags. For a more extensive insight in our method it would be interesting to quantitatively evaluate the results of this solution against manually created frames in future work. Furthermore, we focused our analysis on laws, our solution might also be useful for other types of legal texts. For further improvement we suggest extending the rules for extracting the recipient role, add rules for recognising relative clauses, and testing this framework on English legal texts.

Acknowledgments

The authors would like to thank the Governmental Advisory Board on Digital Government (OBDO) and Dutch Ministry of the Interior and Kingdom Relations for the financial support of our research. Furthermore we would like to thank Tom van Engers and the Norm Engineering project team, particularly Robert van Doesburg, for their insight and feedback.

References

[1] E. Francesconi, The "norme in rete" project: Standards and tools for italian legislation, International Journal of Legal Information 34 (2006) 358–376. doi:10.1017/S0731126500001517.

[2] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, I. Androutsopoulos, Legal-bert: The muppets straight out of law school, arXiv preprint arXiv:2010.02559 (2020).

[3] E. De Maat, R. Winkels, T. van Engers, Making sense of legal texts, volume 212, Walter de Gruyter, 2009.

[4] R. van Doesburg, T. M. van Engers, Explicit interpretation of the dutch aliens act, in: Proceedings of the Workshop on Artificial Intelligence and the Administrative State co-located with 17th International Conference on AI and Law, 2019, pp. 27–37.

[5] R. M. Bakker, R. A. van Drie, M. H. de Boer, R. van Doesburg, T. van Engers, Semantic role labelling for dutch law texts, in: Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), Marseille, 2022, pp. 448–457.

[6] R. Brighi, L. Lesmo, A. Mazzei, M. Palmirani, D. P. Radicioni, Towards semantic interpretation of legal modifications through deep syntactic analysis, in: Proceedings of The Twentieth First Annual Conference on Legal Knowledge and Information Systems, volume 21, 2008, p. 202.

[7] S. Shaghaghian, L. Y. Feng, B. Jafarpour, N. Pogrebnyakov, Customizing contextualized language models for legal document reviews, in: 2020 IEEE International Conference on Big Data (Big Data), IEEE, 2020, pp. 2139–2148.
[8] R. van Gog, T. M. van Engers, Modeling legislation using natural language processing, in: 2001 IEEE International Conference on Systems, Man and Cybernetics. e-Systems and e-Man for Cybernetics in Cyberspace, volume 1, IEEE, 2001, pp. 561–566.
[9] L. Van der Beek, G. Bouma, R. Malouf, G. Van Noord, The alpino dependency treebank, in: Computational linguistics in the Netherlands 2001, Brill, 2002, pp. 8–22.
[10] W. de Vries, A. van Cranenburgh, A. Bisazza, T. Caselli, G. van Noord, M. Nissim, Bertje: A dutch bert model, arXiv preprint arXiv:1912.09582 (2019).
[11] T. De Smedt, W. Daelemans, Pattern for python, The Journal of Machine Learning Research 13 (2012) 2063–2067.
[12] M. Honnibal, M. Johnson, An improved non-monotonic transition system for dependency parsing, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 1373–1378. URL: https://aclweb.org/anthology/D/D15/D15-1162.
[13] G. Bouma, G. Van Noord, Increasing return on annotation investment: the automatic construction of a universal dependency treebank for dutch, in: Proceedings of the nodalida 2017 workshop on universal dependencies (udw 2017), 2017, pp. 19–26.