Automated Classification of Norms in Sources of Law

Emile de Maat and Radboud Winkels

Leibniz Center for Law, University of Amsterdam
{demaat,winkels}@uva.nl

Abstract. The research described here attempts to achieve automated support for modelling sources of law for legal knowledge based systems and services. Many existing systems use models that do not reflect the entire law, and simplify parts of the text. These models are difficult to validate, maintain and re-use. We propose to create an intermediate model that has an isomorphic representation of the structure of the original text. A first step towards automated modelling is the detection and classification of provisions in sources of law. A list of different categories of norms and provisions that are used in Dutch legal texts is presented. These categories can be identified by the use of typical text patterns. Next, the results of experiments in automated classification of provisions using these patterns are presented. 91% of 592 sentences in fifteen different Dutch laws were classified correctly. Some conclusions about the generality of the approach are drawn and future research is outlined.

Keywords: Categorisation of Norms, Experimental Results, Natural Language Processing.

1 Introduction

If we want to design and build systems to support users in handling legal knowledge or data, we will always have to start with the sources of law. After all, in a modern constitutional state, all legal action is grounded in and justified by these sources. Legal texts, however, are meant to be read by humans, and are written in natural language. In order to make these sources available to machines, they need to be translated from natural languages to formal languages. This is a time and effort consuming task, usually performed by knowledge engineers with the aid of legal experts. People are researching ways to support and partially automate this task. A first and relatively easy step is to transform unstructured or badly structured text into a well structured one. We use the MetaLex XML interchange format for legal sources for that.[1] Secondly, we want to find and resolve all references in the text, both internal and to external sources, and tag these explicitly, also using MetaLex. That research has been described before; see e.g. [1]. This chapter will discuss the next step: Recognizing and classifying norms or provisions in the legal sources. The idea is that this classification facilitates the suggestion of model fragments to be used for representing the meaning

[1] www.metalex.eu

E. Francesconi et al. (Eds.): Semantic Processing of Legal Texts, LNAI 6036, pp. 170–191, 2010.

of these norms and provisions. In the (E)-Power project this classification was left implicit and the step from the surface structure of sentences to a formal representation was typically too large to yield useful automatic translations [2]. Making this intermediate step explicit should bridge this gap to some extent. We will first present a classification of norms at a very general level, based on their function in the legal system (Section 2). In Section 4 we will present a finer grained classification and typical examples from Dutch law. These examples also show the typical language structures legislative drafters use (at least in the Netherlands) that will help us in automatically recognising these norms in legal texts. Next, we will show the results of a classifier based upon these typical language structures (Sections 5 and 6). We will end with a discussion and conclusions (Section 7).

2 Models of Legislative Texts

The goal of a law is (or perhaps: should be?) to set rules for the people (and organisations) living in a country (or whatever the jurisdiction of the law is). It tells them what they can do and cannot do, and what their rights and duties are. So, we could expect a law to consist mainly of statements like "Everybody has the right to freedom of speech" and "If you take care of a child less than eighteen years of age, you have a right to child benefit". This turns out not to be the case. Hart [3] distinguishes two types of rules in a law: primary and secondary rules. The primary rules are the rules that refer to human behaviour. Secondary rules are actually rules about primary rules, and form a meta-level. Three types of secondary rules are given by Hart: rules of recognition, rules of change and rules of adjudication. Rules of recognition determine which rules are 'official', rules of change allow for the changing of rules and rules of adjudication empower individuals to judge whether a rule has been broken.

An example of a primary rule is the following one:

General Child Benefit Law, article 7, sub 1
Conform the stipulations of this law, the insured has a right to child benefit for an own child, a stepchild and a foster child which:
a. is younger than 16 years of age and belongs to his household; or
b. is younger than 18 years of age and is maintained by him for a significant amount.

This sentence gives the right to child benefit to the insured, provided certain conditions are met. On the other hand, the following norm is a typical example of a secondary rule. Although it comes from the General Child Benefit Law, there is nothing in this sentence that helps a citizen determine whether or not he has a right to Child Benefit, and, if so, to what amount of Child Benefit.

General Child Benefit Law, article 24b
By Ministerial Decree additional rules can be set regarding the articles 24, sub 1, 2, 3, 4, 5 and 6, and 24a.

In a sense, these secondary rules can be seen as overhead in the law. We want the law to regulate certain things (in this case, child benefit). In addition to the rules on child benefit, however, we need some rules to fit these "core rules" in the legal framework. Other examples are auxiliary provisions that handle enactment of the law, changes, etc.

It is not only the secondary norms, however, that form overhead. The following sentence, for example, is clearly a primary norm, directing some part of the behaviour of citizens:

General Child Benefit Law, article 14, sub 2
A request is made by means of an application form, which is provided by the Social Insurance Bank.

It is not a "core rule", however, as it does not post a direct rule regarding who receives child benefit and how much. Of course, it is rather obvious why it is present in the law. Unfortunately, by merely specifying who receives child benefit and how much, the benefits are not automatically distributed. A system needs to be set up for this to happen. This leads to two layers of additional procedural overhead. The first layer (to which the example above belongs) is still directed to the behaviour of citizens (or citizens' organisations). These rules are not primary rules, even though they are dealing with citizens' behaviour, as they only exist to support the primary rules. On the other hand, they are not rules of recognition, change or adjudication, and therefore, are not instances of Hart's secondary rules. The second layer is about the internal workings of the government and the duties of civil servants. This second layer will contain similar procedural rules, but this time aimed at civil servants. It will also contain norms of competence (as defined in [4]), as well as Hart's rules of adjudication.

An example of a sentence from this second layer is the following:

General Child Benefit Act, article 17d, sub 3
The Attorney General will inform the Social Insurance Bank of any circumstances as meant under sub 1 or 2.

All in all, we come to a four-layer model of the law, as illustrated in Figure 1:

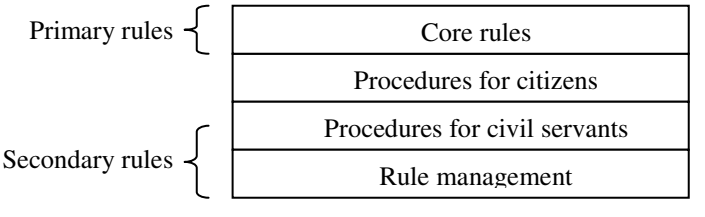

Fig. 1. Layers of norms

This model depicts the four layers of norms as we have described above. For a more complete model of the law, we must remind ourselves that not all the text in a law forms actual norms. In addition to the norms, the body of the law contains definitions. Although definitions could be considered to be normative (for example, a definition of a car could be considered a normative statement determining what may or may not be called a car for the purpose of this law), it is certainly not a normative rule in the sense of e.g. Hart, as it does not deal with human behaviour (primary rule), nor with other rules (secondary rule). The difference with normative rules becomes even clearer when the sentences are studied without context: the rules will usually still have some meaning, though without their accompanying definitions, they are probably vaguer. The definitions, on the other hand, have no meaning outside of the law.

Most primary rules will make use of definitions, but also the procedural rules introduced above. Thus, we cannot say that definitions belong to one specific level of the law. Together with rules, the definitions make up the body of the law. Add to that the introduction, conclusion and appendices, and we come to a more complete model of a legislative text:

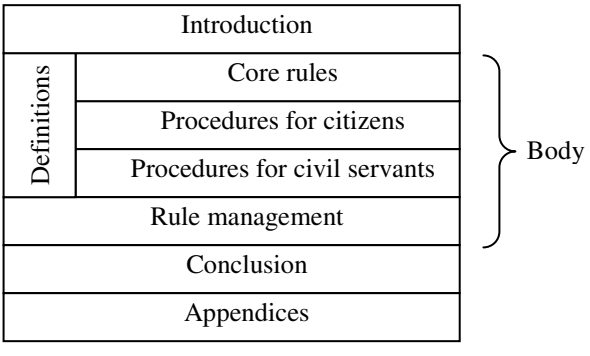

Fig. 2. More complete model of a law text

Most Knowledge Management Systems will focus on the body (and perhaps appendices) of a law text, as the introduction and conclusion will generally not include the kind of information needed for these systems. It is interesting to note that most systems do not focus on all the layers we have distinguished. Most systems that are aimed at given information to citizens will be aimed at the core rules and the procedures for citizens (for example our system for the Legal Services Counter [5]), whereas a process management system for civil servants will most likely incorporate only the two procedural layers. Systems for searching through laws will only model the secondary rules (if any), as those rules are the ones that link the laws together (see for example the Tax Administration Semantic Network [6]).

There have been few attempts to model a law in its entirety. Even within the POWER project [2], which was aimed at laws in their entirety, secondary rules were never modelled and procedural norms often left out. Furthermore, when a model of a

law is made, it is usually a model of the meaning of the law, not of the law as a text. This means that though the model will generate the correct outcome, it will not always do so based on the same structure followed by the legal text.

A third remark regarding the existing models is that they are often targeted to a specific population, and therefore simplified to fit that population. For example, when there is a rule that applies to people younger than 25 or older than 65, the second part will often be omitted in a model for an application aimed at young people. Terms will also often be simplified or (partially) interpreted. For example, the law may use the word "vehicle" which may be simplified to "car" in a specific model.

These models do bring potential problems when they need to be updated (because of a change in the legislation) or when one wants to re-use the model. Because of the simplifications, the model does no longer have a one-to-one correspondence to the sentences in the law, which makes it more difficult to determine which elements of the model are affected by a textual change in the law. Similarly, unless they have been clearly documented, any interpretations are difficult to undo or modify.

To avoid such problems, it would be preferable to first create a model of the entire law, and then derive specific application models from it, as depicted in Figure 3.

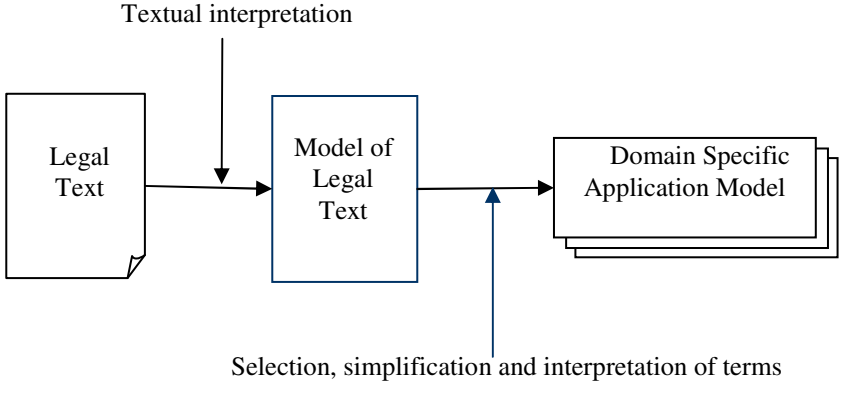

Fig. 3. Model of legal text and domain specific application model

This intermediate model of the law has a clear correspondence to the original text, adhering to the isomorphic representation principle [7]. It should be distinguished from the MetaLex version of the law that deals with structure and referencing. Based on this structure, individual norms or rules are identified (see below) and this leads to suggested model fragments (types of norms are associated with typical representation patterns). It also differs significantly from the integrated, simplified or specialized application specific model that should be built afterwards.

Such an intermediate model should be easy to maintain, update and re-use. Ideally, such a model would be published by the legislator, next to the original textual version of the law, using some generic format, such as perhaps LKIF [8].

3 Granularity for Finding Norms

To make a model of an entire law, each element of the law text should be represented in the model. What is the level of granularity that we should use? Input to the process is a structured, MetaLex version of the law. MetaLex distinguishes and identifies articles, full sentences and sentence fragments. The leading principle in MetaLex design was that the smallest unit that could be referred to should be identifiable. Legal texts can have references of the form "the last sub clause of the previous sentence".

Typically for law texts, articles are the units that can be read and understood without the context of the rest of the law (this does not imply that we do not need the rest for 'correct' application of the law of course). Within articles, sentences are the smallest units that are meaningful by themselves. For our purpose we will start at the sentence level. We assume that each sentence contains a single norm (from which an inverse norm may be derived[2]). Our current research shows some other possibilities in Dutch legal text.

First of all, it may be the case that a sentence contains two main sentences, as in the following example:

Bailiff Law, article 3a, sub 3, second sentence
Because of required speed, the notice may be given orally, in which case it is immediately confirmed in writing.

This sentence describes both the right to give the notice orally, as the duty to confirm the notice in writing (in those cases).

Another situation in which a sentence can contain multiple norms arises when a subordinate clause contains an (implicit) norm:

General Child Benefit Law, article 14, sub 2
A request is made by means of an application form, which is provided by the Social Insurance Bank.

This sentence contains the (explicit) duty to use specific forms to make a request, and an (implicit) obligation for the Social Insurance Bank to provide such a form.

A third category is formed by sentences like this one:

Customs Law, article 13, sub 1
During the investigation, as meant in article 12, only civil servants of the Tax Administration, that have been selected by the inspector, may enter a house without the inhabitant's permission.

In essence, this sentence contains a permission: *Civil servants of the Tax Administration that have been selected by the inspector may enter a house without the inhabitant's*

[2] There are always two sides to a norm. For example, if A is obliged to pay a certain amount of money to B, then B has the right to receive that amount from A. A legal text will only mention one of those two norms; the other can be derived.

permission. Adding the word "only" adds a prohibition to the sentence: it is forbidden for any other person to enter the house. Without the word "only" the sentence does not say anything about other persons.

4 Sentence Types and Patterns

As explained in Section 3, we treat the sentence as the smallest unit that contains a single norm. Obviously, different sentences in a legal source will lead to different models. Therefore, it is important to distinguish between the different types of sentences that appear in legal sources. In this section, we will present a number of sentence types that can be distinguished. Biagioli et al. [9] have used similar categories. However, their categories were based on whole provisions (articles) instead of single sentences, and classified normative provisions as a permission, duty or exception. Sentences on the other hand can be classified in a more detailed way than provisions, so we have some additional classes as will be shown below.

This categorisation is based on earlier research [10] into the types of sentences that occur in laws. However, this earlier research was largely based on a single (though extensive) law, the Income Tax Act 2001. In addition, as it was part of the (E-)POWER project, it focused on the core rules and procedures expressed in the law, and did not pay attention to the rule management part. Hence, the patterns needed to be extended. About 20 other Dutch laws were studied for this purpose, mostly from the last 50 years, but one going back to 1875.

4.1 Definitions

As mentioned in Section 2, definitions are used to describe the terms that occur in a legal source. The terms defined can later be used in both primary and secondary norms. A definition mentions both the term being defined as well as the actual definition. An example of a definition is:

General Administrative Law Act, article 1:4, sub 1
By administrative judge is understood: an impartial body that is appointed by law and charged with administrative judicial settlement

In most Dutch legal texts, definitions use clear patterns, such as "*onder … wordt verstaan*" (by … is understood). This gives us a good method of recognising definitions.

Type extensions are very similar to definitions. However, instead of completely defining a new term, they expand or limit an earlier definition. The most common use of type extensions is to expand a *common sense* definition. In these cases, the law source does not define the term, but instead uses the common meaning of the word, and expands upon that meaning. For example:

Protection of Antarctica Act, article 1. sub 2, introduction and sub a
In this law and the stipulations based on it is also understood by Antarctic environment those ecosystems that are dependent on or related to the Antarctic environment.

They use similar text patterns, but with the inclusion of the word "*ook"* (also) or "*niet*" (not).

Closely related to definitions are the deeming provisions. A deeming provision is a sentence that declares one situation equal to another situation, in a certain context. If one situation is deemed equal to another situation, then the rules that apply to the latter also apply to the first. Deeming provisions introduce some kind of legal fiction. For example:

Income Tax Act 2001, article 2.2, sub 2, introduction and sub a
A Dutchman who is employed by the State of the Netherlands is always deemed to live in the Netherlands if he is posted as a member of a diplomatic, permanent or consular representation of the Kingdom of the Netherlands in foreign countries.

The effect of this statement is that someone is considered to live in the Netherlands, even though he actually lives outside of the Netherlands. In computer models, deeming provisions often appear in the same manner as definitions.

Deeming provisions follow the pattern "*wordt geacht te"* (is deemed to).

4.2 Core Rules and Procedural Rules

The actual content of the legal sources is formed by norms. Normative sentences may confer rights and permissions or impose duties and obligations. Procedural rules are expressed as norms as well, usually they are obligations. Research has been performed on how to distinguish these different types of norms [11]. During this research, it became clear that it was difficult to separate rights and permissions, and that it was likewise difficult to separate duties and obligations. Because of this, these were grouped together. The work yielded a large amount of patterns that were incorporated in the classifier for this project, as they proved useful not only to distinguish between rights and obligations, but also to distinguish these norms from other types of sentences.

An important conclusion was that a majority (almost 71%) of the rights could be identified by the verb "*kunnen*" (may). Another 17% could be identified by the phrase "*is bevoegd*" (is qualified). A host of smaller patterns accounted for the remaining rights.

Sentences denoting duties usually did not follow a pattern; 80% of these sentences was a statement of fact. This means that the text does not state what must happen, but instead simply states that it happens. For example:

Funeral Act, article 46, sub 1
No bodies are interred on a closed cemetery.

In The Netherlands the guidelines for legal drafting recommend the use of the 'statement of fact' instead of a normative formulation [12]. Obviously, such statements of fact have few words in common with similar statements from different domains. Thus, there is no pattern to be found either.

Application Provisions
Application provisions specify situations in which other legislation (usually an article or subsection of an article) does or does not apply. In this way, the application domain of a norm can be extended or restricted (effectively creating an exception to a rule).

Often, an application provision that states that another piece of legislation does apply seems to be included to take away any doubts as to whether it ought to apply or not.

Constitution, article 7, fourth member
The previous members do not apply to making commercial advertisements.

The patterns used by these sentences are "*is (niet) van toepassing*" (does (not) apply).

Penalization
The violation of some norms will carry punishment in the form of a fine or jail. If this is the case, the law will specify the penalties. In Dutch law, this is usually done through sentences like: "*wordt gestraft met*" (is punished with). For example:

Mining Act, article 133, sub 1
Breaking article 43, sub 2, is punished with a monetary fine of the second category.

In general, these sentences are followed by another sentence that denotes whether the punishable fact is a crime or a misdemeanour, such as:

Mining Act, article 133, sub 2
The fact marked as punishable by this article is a misdemeanour.

These sentences always follow this same structure.

Value Assignments and Changes
Any mathematical formulas in a law are given by means of value assignments and changes. A value assignment is a sentence that gives an initial value for a concept value changes are later steps that modify such a value.

An example of such a sentence is:

Income Tax Act 2001, article 3.3, sub 1
Taxable wages are wages reduced with the employee's discount.

These sentences use a range of mathematical operations (to reduce, to increase) and comparisons (at most) which in combination with the verb to be or to amount to can be used to detect them.

4.3 Rule Management

Rule management encompasses a number of sentence types: setting the enactment date, setting the citation title, changing an existing law, and delegating the creating of the new rules.

Enactment Date
This is the first type of sentence that deals with the maintenance of legal texts. All laws will contain a provision for their own enactment date, often relating it to their publication date or deferring that decision to a Royal Decree.

Notaries Act, article 134
This law is enacted on a date to be set by Royal Decree, which may differ for separate parts and articles.

The central pattern used for these sentences is "*treedt in werking op*" (is enacted on), though it is possible to extended this pattern into a number of standardised sentences which are used most of time.

Citation Title
If it is thought necessary, a law will also define a short title which can be used to refer to it.

Notaries Act, article 135
This act may be referred to as: Notaries act.

These sentences follow the standard format "*Deze wet kan worden aangehaald als"* (This act may be referred to as). Together with the enactment date, the citation title is usually present at the end of a legal text.

It is also possible that a law will modify the short title of another law. This is usually done to avoid confusion, when a new law has the same name as he predecessor. In addition, it also happens that the short title may be abbreviated even further. Both types of actions are very rare, and, so far, we have not encountered them often enough to recognise them as a standard type of sentence.

Change Provisions
By means of a change provision, a law can modify some existing legislation. Most laws are change laws, which change some existing legislation rather than introducing a large amount of new rules. In these laws, change provisions make up the bulk of the text.

We distinguish four different types of changes: inserting new text, modifying text, renumbering and deleting or repealing text.

Inserting text follows the pattern "*wordt ingevoegd*" (is inserted) or "*wordt toegevoegd*" (is appended), depending on whether the new text is added somewhere within a section or at the end of the section. An example of such a sentence is:

Act of June 6th, 2002 (Stb. 303), article I, sub IIa
To article 7.36, a new sentence is appended, to read as follows: Article 7.34, sub 5, applies correspondingly.

Modifying text can happen in two ways. If a small amount of text is to be changed, then that text is quoted as well as the new text. The pattern followed is "*wordt vervangen door*" (is replaced by). For example:

Act of June 6th, 2002 (Stb. 303), article III, sub V
In article 7.12, sub 1, second sentence, «article 7.3b» is replaced by: article 7.3c.

Alternatively, if an entire sentence, section or article is replaced, the modifying provision will simply refer to that element and quote the new text:

Act of June 6th, 2002 (Stb. 303), article IV, sub B
Article 2.8 will read: …

The pattern for such replacements is "*komt te luiden*" (will read).

A repeal can repeal an entire law, a section of a law or just a bit of a text. It follows the pattern "*vervalt"* (is repealed), as in this example:

Act of June 6th, 2002 (Stb. 303), article I, sub QQ
Article 17.2 is repealed.

The last change is the renumbering of structural elements. Because renumbering an element requires the modification of all text referring to that element, it is somewhat uncommon for articles to be renumbered. On the other hand, anything below the level of article (subsections and lists) is almost always renumbered to keep a continuous numbering.

Renumbering can either be done explicitly or implicitly. Explicitly renumbering means that the action is identified as a renumbering, using a pattern like "*wordt vernummerd tot*" (is renumbered to) or "*wordt verletterd tot*" (is re-lettered to).

Act of June 6th, 2002 (Stb. 303), article IIIc, sub C
The articles 17a.1 to 17a.25 are renumbered to the articles 17.20 to 17.54.

Implicit renumbering is done by viewing the number not as an attribute of the structural element, but just as a piece of text, which is subsequently changed.

Act of June 6th, 2002 (Stb. 303), article IIIa, sub A
The heading of chapter 5a will read: Chapter 5. Accreditation in higher education.

Additionally, renumbering is often not done in a separate provision, but mentioned as a side effect of a provision that inserts new text.

None of the sentences above have a complete reference. They refer to articles, but not to articles in a specific law. Normally, this would mean that they refer to an article in the same text, but this is most often not the case. If many changes are made in the same text, then they are grouped and preceded by a sentence that sets the scope, such as:

Act of June 6th, 2002 (Stb. 303), article I, introduction
To the Higher Education and Academic Research Act, the following modifications are made:

Such scope declarations follow the pattern "*wordt als volgt gewijzigd"* (is modified as follows) or "*worden de volgende wijzigingen aangebracht"* (the following modifications are made).

Delegation
Delegations confer the power to create additional rules to some legal entity. Most often, this power is conferred onto a minister, for the creation of rules that do not require (immediate) involvement of the parliament. In some cases, the delegation is an order to create rules to arrange for something; in other cases, it merely allows for the creation of rules should the need arise. An example pattern is "*kan regels stellen*" (may create rules).

Agricultural Tenancies Act, article 3, sub 1
By Order in Council, rules are set with regard to the highest allowed rent.

Related to the delegations are the publishing provisions, which order a minister to publish certain rules created by a (lower) entity.

5 Experimental Classifier

We built a classifier (in Java) that takes well structured legal sources as input and tries to classify their sentences according to their type based on typical patterns associated with these types. An important limitation to this approach is formed by the statements of fact. As stated in section 4, these do not follow any recognisable patterns. Because of this, we hope to identify this important group of statements "by default": if we can identify patterns for everything else, we may assume that anything not classified by these patterns is one of these statements of fact that actually reflect norms.

The classifier assumes that the input is structured using MetaLex XML. In MetaLex, sentences and lists are marked, as well as the separate list items within each list. This enables the classifier to treat each sentence separately. This means that a necessary pre-condition for the classifier is that the structure of the documents has already been marked. For legacy texts an automatic structure recogniser would therefore be desirable. For more recent texts, this is usually not a problem. These documents often have already been tagged, or have even been produced in XML format using modern XML based legislative editors like MetaVex or XMLeges[3].

The classifier is a simple pattern matcher. We used 88 patterns from about twenty Dutch laws. Most patterns consist of only a verb phrase, like "*mogen"* (may) for a right/permission or "*wordt aangehaald als*" (is referred to as) for the defining of a citation title. Sometimes, additional keywords have been added, as in "*kan regels stellen*" (may create rules).

The patterns are stored in a format for the Java pattern matcher (`java.util.regex`). The patterns mentioned above become:

[3] In addition to automatically marking the structure, such editors could even tag sentences during construction as being of a specific type, for instance because the legislative drafter chooses a particular template in MetaVex [13].

```
\s+(mag|mogen)\b
\s+wordt\s+aangehaald\s+als(:)?\s+
\s+kan\s+regels\+stellen\s+
```

In this format, \s+ denotes one or more whitespace characters, \b denotes a word boundary, and (:)? is an optional colon. The first pattern allows for either the singular or the plural form of the verb.

The classifier will attempt to match a sentence to each available pattern. If the sentence matches several patterns, the classifier will prefer the longest of the matches. (This does not happen often; however, some of the patterns overlap, such as "*kan*" for a right and "*kan regels stellen*" for a delegation).

A specific strategy needed to be chosen to tackle embedded lists, such as.

Tobacco Act, article 1
In this law, and in the stipulations based on it, is understood by:
a. tobacco products: … ;
b. Our Minister: …;
c. appendix: …;
…

Our initial assumption was that we could base the classification on the first part of the sentence, and that the individual list items were not needed for the classifications. This assumption did not hold, as we encountered list in which the verbs, and thus the patterns, did not occur in that first sentence part [14]. An example of such a list is:

Bill 20 585 nr.2, article 5
Our Ministers:
a. appoint, suspend and discharge the chairman and other members, after hearing the council involved;
b. appoint, suspend and discharge the advising members.

As the first sentence part does not contain any patterns, these lists were always classified as a statement of fact (the default) which, in the case of this example, would have been correct.

To come to a more correct classification, we came to a different approach: the classifier first classifies the introduction. If it does not find an explicit pattern, it then continues to classify the different list items.

If the first sentence does contain an explicit pattern, the list would be classified as entirely of the matching type. If the first sentence does not contain an explicit pattern, but (some of) the list items did, the list items would be classified independently. If neither contained explicit patterns, the list as a whole would be classified as a statement of fact.

Another group of somewhat difficult sentences are those that replace or insert text. Those sentences quote old and/or new text fragments, which also may contain a pattern. Thus, the sentence as a whole can contain multiple patterns and can easily be misclassified.

Because we assume that the input is tagged in MetaLex, this does not pose an actual problem, though. In MetaLex, such texts are marked using so-called 'quote' elements. When a sentence is parsed that contains such elements, the contents of those quotes are not used in classifying the sentence. If the quoted text contains *complete* sentences or lists however, the classifier will attempt to classify those as well as the containing sentence.

6 Results

We tested the classifier on eighteen different Dutch regulations, different from the twenty we used to come up with the patterns. Four of these eighteen regulations were completely new laws; the others changed already existing laws (as is the more common situation). With the exception of a single Royal Decree, these where all bills pending at Parliament. In this they differed from the set used to derive the patterns, which where all acts that had already been passed. The length of the laws varied from very short (three sentences) to quite long (166 sentences on 23 pages A4); most were quite recent (patterns in the past have been different).

All laws are listed in Table 1 below. To check whether clauses were classified correctly, all sentences and lists in all laws were also classified manually.

Table 1. Results per source

Source	*Sentence*			*List*				*Type*
	Total	Correct	%	Total	Correct	Partial	%	
Royal Decree Stb. 1945, F 214 (as modified per 01/01/2002)	26	23	97%	4	4	0	75%	New
Bill 20 585 nr. 2	31	30	97%	4	3	1	75%	New
Bill 22 139 nr. 2	22	20	91%	2	2		100%	New
Bill 27 570 nr. 4	21	16	76%					Change
Bill 27 611 nr. 2	11	11	100%	1	1		100%	Change
Bill 30 411 nr. 2	141	128	91%	25	20	3	80%	New
Bill 30 435 nr. 2	40	39	98%	4	3	1	75%	Change
Bill 30 583 nr. A	27	27	100%					Change
Bill 31 531 nr. 2	3	3	100%					Change
Bill 31 537 nr. 2	29	29	100%	2	2	0	100%	Change
Bill 31 540 nr. 2	7	7	100%					Change
Bill 31 541 nr. 2	8	8	100%					Change
Bill 31 713 nr. 2	7	6	86%	2	2	0	100%	Change
Bill 31 722 nr. 2	31	22	71%	6	5	0	83%	Change
Bill 31 726 nr. 2	78	67	86%	2	1	1	50%	Change
Bill 31 832 nr. 2	7	7	100%	3	3		100%	Change
Bill 31 833 nr. 2	4	4	100%					Change
Bill 31 835 nr. 2	99	90	91%	7	4	3	57%	Change
Total	**592**	**537**	91%	**62**	**50**	**9**	81%	

Table 1 shows the total number of sentences and lists in the sources we used for testing, as well as the number of sentences and lists that were classified correctly. The column *type* indicates whether a law was completely new, or whether it was a change law, mainly aimed at modifying an existing piece of legislation. Change laws are far more common than completely new legislation. They tend to contain less definitions and norms than new legislation.

The classifier performs well, classifying 91% of all sentences and 81% of all lists correctly[4]. We expect these results to generalize over all Dutch laws. For most natural language domains, a pattern based systems is thought to have too little generalisation capacity. Although languages do have underlying rules, people will often stretch and bend these to their need. As a result, a system based upon patterns is often too rigid to deal with all the information that can occur [16], and a statistical method is usually recommended [16,17]. From our results, it becomes clear that most laws use only a limited set of patterns. If a pattern is missing, the accuracy of the classifier could drop fast. However, the amount of variation in legal texts is restricted, as legal drafters will seldom use a completely new style, instead using the style of older laws or the official guidelines. This is also confirmed in our tests: the majority of all sentences is classified by a small number of patterns.

On lists, the classifier does perform a lot worse than on sentences. Many lists that should be classified as a statement of fact, and that should be classified based on the first sentence part, were misclassified in this new approach. This was due to the fact that the list items contained one of the other patterns, usually as a subordinate clause. (Subordinate clauses lead to more problems, which we will discuss below).

In most cases, the first sentence part is supposed to form a correct sentence with each of the separate list items. In order to classify a list, we could derive each of those sentences and classify them. This would solve some of the problems, when the pattern was split over the introduction and the individual list items.

Table 2 presents the results for the different types of sentences (as the performance on lists depends very much on the approach chosen on how to handle them, and not only on the patterns, we will keep them out of the discussions on the performance of the patterns). The column *"In corpus"* shows the number of sentences present in the test set for each type, both as an absolute number and as a percentage. The column "*Missed"* shows how many of these sentences were not correctly identified. For example, the test set contained 40 application provisions, but one was incorrectly classified (meaning that 39 were correctly classified). The column "*False*" presents the amount of sentences that were incorrectly classified as a particular type, e.g. in the same row as before eight sentences were incorrectly classified as an application provision. Each false positive corresponds to a 'missed' somewhere else.

The greatest part of all the sentences is formed by the norms. 43% of all sentences belong to one of the norm categories. The next biggest category consists of the change provisions, with 41%. These changes can be further broken down as follows:

[4] Which is actually worse than the method we tested in [19], where we classified all lists based on their introduction sentence.

Table 2. Results per sentence type

Type	*In corpus*		*Missed*	*False*
Definition	2%	12	1	0
Norm - Right/Permission	11%	64	4	13
Norm - Obligation/Duty	5%	29	0	1
Delegation	3%	19	6	0
Publication Provision	1%	4	0	0
Application Provision	7%	40	1	8
Enactment Date	3%	17	1	0
Citation Title	1%	3	0	0
Value Assignment	0%	1	0	0
Penalisation	0%	0	0	2
Change	41%	241	16	8
Mixed Type	1%	3	3	0
Norm - Statement of Fact (default)	27%	159	23	23
Total		592	55	55

Table 3. Results for change sentence types

Type	*In corpus*		*Missed*	*False*
Scope	9%	54	0	0
Insertion	7%	44	1	0
Replacement	19%	111	4	0
Repeal	4%	23	7	8
Renumbering	2%	9	4	0
Total		241	16	8

Some sentences were a concatenation of two sentences. For example, one sentence contained two changes: a renumbering and a repeal. These sentences are listed in Table 2 as 'Mixed type'. About half of the misses were caused by patterns that were unknown to the classifier. These sentences were incorrectly classified as the default (statement of fact), and sometimes as a norm of the type right/permission.

Two notable patterns were missing: a renumbering pattern dealing with re-lettering rather than renumbering, and a new pattern for delegations. Both of these should probably be added to the classifier. The other missing patterns occurred only once, and unless we encounter them much more often in the future, these are errors that are unlikely to be repaired in future versions of the classifier.

Those misclassifications that were not caused by missing patterns were instead caused by patterns that were somehow too broad. For example, most false positives of

the "repealed" type sentences were provisions concerning the repeal of fines instead of articles. This will require more sophisticated patterns or dedicated 'anti-patterns' (i.e. not applicable when it contains the word 'fine').

Both false penalisations were in fact a right; the pattern that triggered this classification was part of a qualification of a legal body that was given certain rights. Such a qualification is given in a subordinate sentence. This means that the classifier will find two (or even more) patterns: one in the subordinate sentence, and one in the main sentence. As it does not have the option to distinguish between the two, it will pick the longest match (which will not always be the correct one).

If the main sentence does not contain any pattern (because it is a statement of fact), the classifier will only find the pattern in the subordinate sentence, and will automatically arrive at the wrong conclusion. This is the cause of almost all false rights and false application statements. It would be preferable if the classifier could ignore the subordinate sentences completely. This would require that the sentences be split into main and subordinate sentences before classification, which would mean creating a far heavier application than the current classifier. However, as the classification is the first step in the larger process to automatically generate models, splitting main and subordinate sentences will be of use (most likely even necessary) in later steps of the process.

We only encountered one value assignment in the texts classified during this experiment. These seem to be specific to certain domains (i.e. taxes), and perhaps they are usually deferred to lower order regulations. The test set did not contain any deeming provisions, nor any penalisation provisions.

Of the 88 patterns we identified in our training set, only 44 were actually present in our test set. A possible explanation for this is that our training set was more spread out in time, while our test set consists mainly of rather recent laws (not counting the Royal Decree from 1945, the laws in the test set are all from the last 20 years, whereas the training set contains several laws from before 1960, going back as far as 1875). However, these test results may also be an indication that there are too many patterns in our classifier, and that some ought to be removed.

Table 4 shows the distribution of the patterns that were actually encountered in the test set. The numbers suggest that there are a couple of main patterns that account for a majority of the sentences identified. For example, of the 60 correctly identified rights, 55 used the pattern "may" and four used the pattern "is qualified". This corresponds to Franssen's [11] conclusion that the majority of rights could be identified with those two patterns. This distribution again suggests that some of the other patterns may be superfluous.

Table 4 also shows that most of the false positives are caused by a small set of patterns: one pattern for repeal, one for rights and one for application provisions. A possible solution to the false positives may be to narrow these patterns down. However, these three patterns are also responsible for a large number of correct identifications, and narrowing them down may reduce the success rate.

If we study the number of patterns encountered in each separate law, shown in Table 5, we see that most laws do not use a lot of different patterns. One or two per type seems most common, with sometimes three or four patterns for the bigger categories. This may be explained by the fact that a limited number of legal drafters work

Table 4. Distribution of patterns used

Type	Patterns Known	Patterns Used	Results per pattern	
			Correct	False
Definition	14	5	6	0
			2	0
			1	0
			1	0
			1	0
Norm – Right/Permission	17	3	55	13
			4	0
			1	0
Norm - Obligation/Duty	15	8	6	1
			5	0
			3	0
			2	0
			2	0
			1	0
			1	0
			1	0
Delegation	7	5	5	0
			4	0
			2	0
			1	0
			1	0
Publication Provision	1	1	4	0
Application Provision	5	5	36	5
			2	0
			1	0
			0	1
			0	2
Enactment Date	1	1	16	0
Citation Title	2	2	2	0
			1	0
Value Assignment	8	1	1	0
Penalisation	3	1	0	2
Change – Scope	2	2	49	0
			5	0
Change – Insertion	4	4	22	0
			18	0
			2	0
			1	0
Change - Replacement	3	3	66	0
			40	0
			1	0
Change – Repeal	2	1	16	8
Change - Renumbering	3	2	4	0
			1	0
Total	**87**	**44**	**393**	**32**

on a specific law (on the really small ones perhaps only one) and have their specific sets of regular expressions they use.

This suggests that if a pattern that is used in a law is missing in the classifier, there will be a huge drop in the accuracy of the classifier for that type, as most patterns account for a fairly large portion of the sentences. During this test, this has only occurred in small categories, so this had only a limited impact (i.e. in Bill 31 835, all five repealing sentences have been missed due to a missing pattern).

Table 5. Experimental results for all bills

	Number of sentences	Definition	Norm - Right/Permission	Norm - Obligation/Duty	Delegation	Publication Provision	Application Provision	Enactment Date	Citation Title	Value Assignment	Change - Scope	Change - Insertion	Change - Replacement	Change - Repeal	Change - Renumbering
Number of patterns		14	17	15	7	1	5	1	2	8	2	4	3	2	3
Royal Decree Stb. 1945, F 214	26		1	3	3		2	1						1	
Bill 20 585 nr. 2	31		2	1			1	1	1						
Bill 22 139 nr. 2	22		2	1			1	1			1		1		
Bill 27 570 nr. 4	21		1					1			1		2	1	1
Bill 27 611 nr. 2	11		1	2				1			1	1	1		
Bill 30 411 nr. 2	14 1	4	2	3	2	1	2	1	1	1	1	2	2	1	
Bill 30 435 nr. 2	40		1				2	1	1		1	2	2	1	1
Bill 30 583 nr. A	27			1				1			1	2	1		
Bill 31 531 nr. 2	3						1	1					1		
Bill 31 537 nr. 2	29			2			1	1			2	2		1	
Bill 31 540 nr. 2	7	1						1			1	1		1	1
Bill 31 541 nr. 2	8						1	1			1	1	1		
Bill 31 713 nr. 2	7						1	1			1	1	1	1	
Bill 31 722 nr. 2	31		1		1		1	1			1	1	2	1	
Bill 31 726 nr. 2	78		1	1	2		1				1	2	2	1	1
Bill 31 832 nr. 2	7	1						1					1		
Bill 31 833 nr. 2	4						1	1				1			
Bill 31 835 nr. 2	99		1		1		1	1			1		2	1	

7 Conclusions and Discussion

In this chapter, we have discussed the detection and classification of norms in legal texts using normative sentences combined with sentences that form definitions, deeming provisions, exceptions and application provisions. We distinguished between different layers of norms in a more detailed extension of Hart's primary and secondary norms. In a lot of existing models and applications, deeming provisions, exceptions and application provisions are not retained, but simplified to if-then-else statements. That is why, at the start of this chapter, we proposed to make an intermediate model of the law that retains isomorphism with the original text and would be easier to maintain.

A first step to the creation of such models is classifying the provisions that occur in a law. In Section 4, we have presented a possible classification. The classification presented there seems to be adequate for Dutch laws. All provisions encountered while gathering information are covered, as well as all provisions encountered in our test data. The classification is based on Dutch law texts. In order to extend it to other jurisdictions, some modification may be needed. However, a comparison with Tiscornia and Turchi [15] suggests that the classification for Dutch and Italian law is rather similar.

Although there are multiple language constructs for each sentence type, these are limited, making them easy to detect. As an experiment, we have set up a classifier that attempts to classify sentences based on these patterns. That classifier works well. Within the laws used for our experiment, 91% of all phrases were classified correctly, with hardly any false positives. Almost 43% of all phrases were classified as some type of norm, a further 41% as clauses changing an existing law. These results are similar to those reached in [9], where a machine learning approach is used to classify provisions in Italian law. This suggests that both methods are capable of reaching comparable results, despite the fact that patterns are often seen as less suitable for dealing with natural language. We suspect that the issues that arise with a pattern-based approach, like the errors generated by auxiliary sentences, will also hamper a machine learning approach.

We expect these results to generalize over all Dutch laws, though we need more patterns for specific domains (see below). For other languages and jurisdictions, we expect that the same categories, with different language patterns, will form a good starting point.

Of course, there is room for improvement as well. A major cause of misclassifications is the occurrence of patterns in subordinate sentences. Detecting these sentences beforehand so they may be ignored during the classification would lead to a serious increase in performance. Another improvement can be achieved in handling the lists. Both straightforward methods we have tested did not perform as well as we would desire (though the recognition of subordinate sentences would help here as well).

Finally, additional patterns may be added to improve results. We know that specific domains (such as tax law) will be needing specific patterns. For example, in our earlier research on the Income Tax Act 2001, the patterns used for definitions and type extensions were: *x is y*, or: *x are y* instead of *by x is understood y* 10. So far, our results suggest that the Income Tax Act was unique in its use of these patterns, and we have not included them in the current version of the classifier (meaning that this classifier will not perform well on the Tax Income Act for definitions). Also, some

of the sentences that were not classified in the test used language constructs that may also be new patterns, which should be added if they are encountered more often.

The next step should be to create actual models from a sentence, using its classification as a basis.

Acknowledgements

These experiments and their results were presented in two earlier conference papers: [14] and [22]. An earlier version of the categorization of norms was presented in [19].

We would like to thank our student Gijs Kruitbosch for his work on the classifier.

References

1. de Maat, E., Winkels, R., van Engers, T.: Automated Detection of Reference Structures in Law. In: van Engers, T. (ed.) Legal Knowledge and Information Systems - JURIX 2006: The Nineteenth Annual Conference, pp. 41–50. IOS Press, Amsterdam (2006)
2. van Engers, T.M., Kordelaar, P.J.M., Den Hartog, J., Glassée, E.: POWER: Programme for an ontology based working environment for modelling and use of regulations and legislation. In: Tjoa, A.M., Wagner, R.R., Al-Zobaidie, A. (eds.) Proceedings of the 11th workshop on Databases and Expert Systems Applications (IEEE), Greenwich, London, pp. 327–334 (2000)
3. Hart, H.: The Concept of Law. Clarendon Press, Oxford (1961)
4. Ross, A.: Directives and norms. Routledge and Kegan Paul Ltd., England (1968)
5. van Engers, T.M., Winkels, R., Boer, A., de Maat, E.: Internet, portal to justice? In: Gordon, T. (ed.) Legal Knowledge and Information Systems. Jurix 2004: The Seventeenth Annual Conference. Frontiers in Artificial Intelligence and Applications, pp. 131–140. IOS Press, Amsterdam (2004)
6. Winkels, R., Boer, A., de Maat, E., van Engers, T., Breebaart, M., Melger, H.: Constructing a semantic network for legal content. In: Gardner, A. (ed.) Proceedings of the Tenth International Conference on Artificial Intelligence and Law (ICAIL 2005), pp. 125–140. ACM Press, New York (2005)
7. Bench-Capon, T.J.M., Coenen, F.P.: Isomorphism and Legal Knowledge Based Systems. Artificial Intelligence and Law 1(1), 65–86 (1992)
8. Hoekstra, R., Breuker, J., Di Bello, M., Boer, A.: The LKIF Core ontology of basic legal concepts. In: Casanovas, P., Biasotti, M.A., Francesconi, E., Sagri, M.T. (eds.) Proceedings of the 2nd Workshop on Legal Ontologies and Artificial Intelligence Techniques (LOAIT 2007). CEUR-WS.org, pp. 43–63 (2007)
9. Biagioli, C., Francesconi, E., Passerini, A., Montemagni, S., Soria, C.: Automatic semantics extraction in law documents. In: Gardner, A. (ed.) Proceedings of the Tenth International Conference on Artificial Intelligence and Law (ICAIL 2005), pp. 133–140. ACM Press, New York (2005)
10. de Maat, E.: Natural Legal Modelling. Master's thesis, University of Twente, Enschede (2003)
11. Franssen, M.: Automated Detection of Norm Sentences in Laws. In: Twente Student Conference on IT (2007)
12. Aanwijzigingen voor de regelgeving. Circulaire van de Minister-President. Original in Staatscourant 1992, 230 (1992); Last modification in Staatscourant 2005, 58 (2005)

13. van de Ven, S., Hoekstra, R., Winkels, R., de Maat, E., Kollár, A.: MetaVex: Regulation Drafting Meets the Semantic Web. In: Casanovas, P., Sartor, G., Casellas, N., Rubino, R. (eds.) Computable Models of the Law. LNCS (LNAI), vol. 4884, pp. 42–55. Springer, Heidelberg (2008)
14. de Maat, E., Winkels, R.: Automatic Classification of Sentences in Dutch Laws. In: Francesconi, E., Sartor, G., Tiscornia, D. (eds.) Legal Knowledge and Information System - JURIX 2008: The 21st Annual Conference Annual Conference, pp. 207–216. IOS Press, Amsterdam (2008)
15. Tiscornia, D., Turchi, F.: Formalization of legislative documents based on a functional model. In: Proceedings of the 6th international conference on Artificial intelligence and law, pp. 63–71. ACM Press, New York (1997)
16. Nanning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
17. Moens, M.-F.: Innovative techniques for legal text retrieval. Artificial Intelligence and Law 9(1), 29–57 (2001)
18. de Maat, E., Winkels, R., van Engers, T.: Making Sense of Legal Texts. In: Grewendorf, G., Rathert, M. (eds.) Formal Linguistics and Law, Mouton, De Gruyter (2010)
19. de Maat, E., Winkels, R.: Categorisation of Norms. In: Lodder, A., Mommers, L. (eds.) Legal Knowledge and Information Systems - JURIX 2007: The Twentieth Annual Conference, pp. 79–88. IOS Press, Amsterdam (2007)
20. Gonçalves, T., Quaresma, P.: Is linguistic information relevant for the classification of legal texts? In: Gardner, A. (ed.) Proceedings of the Tenth International Conference on Artificial Intelligence and Law (ICAIL 2005), pp. 168–176. ACM Press, New York (2005)
21. McCarty, T.: Deep Semantic interpretations of legal texts. In: Winkels, R. (ed.) Proceedings of ICAIL 2007, pp. 217–224. ACM Press, New York (2007)
22. de Maat, E., Winkels, R.: A Next Step towards Automated Modelling of Sources of Law. In: Hafner, C. (ed.) Proceedings of ICAIL 2009, pp. 31–39. ACM Press, New York (2009)