Making sense of legal texts

de Maat, E.

Publication date
2012

Summary

The rules of a modern society are codified in statutes, regulations and case law. These rules provide a specification of how a society should be. They tell us how civilians in a society should behave, and how the society's government and civil servants should behave. Those specifications differ from technical specifications as they are written in natural language rather than in a formal language or schematics. However, regulations are based on established best practices and design patterns. As such, legislation is far more regular than most other natural language text, and not that different from technical specifications.

Within the field of Computer Science & Law (the branch of computer science that deals with the law), this regularity of the legislation is used as the basis for applications that increase the accessibility of the legislation. One common type of application is a portal that allows users to search for legislation using keywords, titles, etc. In such portals, regulations are not stored as plain text, but in a more structured format. Such a format makes it possible to point users to a specific section of a document, and it also makes it possible to store different kinds of metadata, which can also help a user to find the information he needs.

For users without a legal background, however, such applications are not helpful enough. Such users are often looking for an answer to a specific legal problem, rather than mere legal information. For those users, getting a link to the relevant regulations is an insufficient answer. Thus, a second type of application is developed: systems that provide these answers by questioning the user about his situation and then applying the rules. Such systems are not only useful for citizens that need answers to their legal problem, but can also help civil servants process routine cases (for example, when processing permit applications).

This second type of application requires complete, formal and executable representations of the law. Creating such representations takes a lot of effort. This is a common problem with any knowledge-based system, and has become known as the *knowledge acquisition bottleneck*. The research presented here aims to (partially) overcome this bottleneck by attempting the automated translation of Dutch laws to computer models.

This automated translation is not done in a single step. Instead, it is broken up into several phases:

1. structure recognition: the structure of the law is recognised and marked-up;
2. detecting references: any references in the law to other legislative sources are detected and marked-up;
3. sentence classification: the sentences in the law are assigned a specific type;
4. modelling sentences: a model of each sentence is made;
5. integrating models: the models of individual sentences are integrated into a single model of the law.

The (Dutch) official guidelines for legislative drafting form the basis of the recognition of structure elements in Dutch legislative documents (discussed in chapter 2). These guidelines divide a law into six parts: the title, preamble, body, conclusion, signatures and appendices. The body is subsequently split into articles, which may be split into subparagraphs and may

contain a list with list items. The articles may be grouped into sections, divisions, titles, chapters and books.

These different parts are what we want to recognise and identify, i.e. we want to know that certain sections of the document belong together, but we also want to know that they form article 12, or chapter II.

The preamble, conclusion and signatures use specific formulas, which are prescribed in the guidelines. As a result, we can easily find and identify them by searching for the fixed elements of those formulas. Most other elements are marked with a header or an index. Such a header consists of a category label (such as *article* or *chapter*), an index (such as *4*, *II* or *5:128*) and optionally a title. By finding these headers, we can find the start of a new structure element (and the end of the previous element). Finally, articles and subparagraphs need to be split into sentences. This can be achieved using existing tools.

A prototype parser built on these principles using the General Architecture for Text Engineering (GATE) was tested on ten different laws. It recognised the structure on article level and above without errors. In 2% of the articles themselves, there was an error at the level of subparagraphs, lists or sentences. The structure of the remaining 98% of the articles was correctly detected by the prototype parser. The main issue that remains in the detecting of structure of laws is the inclusion of quoted text inside articles that modify existing laws. Such quoted text may contain structure elements that are not part of the law itself. Dutch laws do not include quotation marks to recognise the beginning and end of such parts, and as a result, this needs to be determined by comparing the indices of different elements (as the indices of the quoted text will usually not be continuous with the indices of the quoting texts).

Like headers, references conform to specific patterns, which we can use to detect them (see chapter 3). References to a legislative document are made using its (short) title or its signing date. Specific structure elements of a document are described using a category label (*chapter, article, list item*) and the index of the element. A complete reference includes the document that is being referred to, but incomplete references, which only refer to a structure element, are very common. Their exact target depends on their context. Within a law, an incomplete reference will often refer to a location within that law itself, unless it is part of a list of changes to be made in some other law, in which case it refers to the law being modified. References may also have multiple targets, either by listing multiple indexes (such as *articles 12, 13 and 14*) or by combining them in a range (such as *articles 12 – 14*).

The references that consist of a category label and an index (or multiple indices or a range) can easily be detected using patterns. The same holds true for documents being referred to by their signing date. Short titles, on the other hand, are very varied, and cannot be described using a pattern. In order to detect these, a list of all available titles is needed.

A prototype parser has been constructed that uses such a list of available titles as well as patterns for documents by signing date and category label plus index. This parser has been tested on six different Dutch laws containing over 1,000 references. It detected 97% of those references successfully, with only a few false positives.

In chapter 4, the classification of sentences that appear in Dutch laws is discussed, splitting the sentences into fifteen broad categories. First of all, there are the actual normative sentences: rights/permissions and obligations/duties. These sentences are supported by definitions, which define the terms being used by the normative sentences. Similar to definitions are the deeming provisions, which deem certain items to be equivalent for the application of certain norms. Application provisions specify additional situations in which certain norms do (not) apply, effectively creating additional conditions for those norms. Penalisation provisions set the punishment for violating a norm. A publication provision is a specific obligation ordering the publication of certain information. A value assignment calculates some value, which is used in some norm.

Next to these rules that form the actual law, there are some other sentence types that are about the law itself. There are sentences that set the enactment date or the short title of a law, and delegations delegate the right (or duty) to set additional rules. Other sentences modify existing regulations, by inserting new text, replacing text, repealing text or renumbering existing elements. Scope definitions set the scope for these changes, thereby simplifying the references that are needed in the modifying sentences.

Each class of sentence uses specific keywords or language patterns which can be used to classify the sentences, with one exception. Though many obligations do use keywords like *must* or *obligated*, it also common to describe an obligation as a fact, i.e. by stating that the obligation action happens rather than that it must happen. Such a description lacks a keyword like *must*.

Since the only sentences that do not (always) conform to a pattern are obligations, we can still use the patterns to classify the sentences. If no pattern applies to a sentence, then we assume it is an obligation. A classifier was built using such patterns. It was tested on eighteen different laws with 591 sentences (not counting lists). Of these 591 sentences, 157 were "statements of fact", i.e. obligations that were written down as a fact. The classifier was capable of correctly classifying 91% of the sentences.

When it comes to text classification, a machine learning approach is often more successful than knowledge-based approached. Therefore, a machine-learning approach using Support Vector Machines (SVMs) was tested, in addition to the pattern-based approach. This was done on the same data set that was used to test the pattern-based approach, using a leave-one-out (LOO) approach. This means that a classifier is trained on all sentences but one. This classifier is then used to classify the one sentence that was left out. This is then repeated for all the sentences. From the number of correct classifications, a LOO accuracy can be derived, which is an indicator for the actual accuracy of such a classifier. For the classification of legislative sentences, the machine-learning approach achieved a LOO accuracy of 95%. In a test on two new laws, the machine-learning classifier scored an accuracy of 95.77% and 88.78%, compared to 94.37% and 95.61% for the pattern-based approach, suggesting that neither approach is strictly better than the other. The pattern-based approach does have the advantage of transparency: the knowledge of what pattern was used to classify the sentence helps with the modelling of the sentences.

For the modelling, discussed in chapter 5, there is a difference between two groups of sentences: sentences that deal with the law itself and sentences that deal with the subject

matter of the law. Sentences that deal with the law itself are those sentences that modify existing laws, set the enactment date or citation title of a law, etc. These sentences describe a specific situation, and follow a strict, more or less standardised format. On the other hand, sentences that deal with the subject matter of the law are norms and definitions. These can describe a great variety of situations, and do not follow standard formats.

For sentences that deal with the law itself, we can rely on the few standardised phrases that are used in order to extract the information that is needed to model the sentence. For example, when dealing with a sentence that replaces some text, we need to know where we should replace the text, what text to replace and by what to replace it. In these sentences, the location is always mentioned in the form of a reference, the text to replace is marked using quotation marks, and the text to replace it with follows the (first) colon in the sentence. For other sentence types that deal with the law, such as setting the enactment date and repealing a law, similar patterns can be applied to extract the needed information. A manual count of a corpus consisting of 343 sentences that deal with the law, suggests that about 96% of these sentences can be modelled using this approach.

Sentences that deal with the subject matter of the law are more diverse, and do not conform to a few standardised phrases. For those sentences, a more generic method is proposed. For normative sentences, a frame is created describing the action or situation that is being allowed or disallowed. Such a frame includes information such as action, agent and patient. This information can be extracted from the sentence using existing natural language processing tools.

These frames form a basic tool for applying the rules. By describing a case in similar terms (i.e. action, agent, patient, etc.) we can compare the situation in the case with the situations described in the rules. If the situation matches, the rule applies, and it becomes clear if the situation in the case is allowed or not (under that rule).

A similar approach can be followed for conditions and definitions. When dealing with a condition, if the case matches the description, this indicates that the rule to which the condition applies should be followed. When dealing with a definition, a match means that any rules that apply to the defined concept also apply to this case.

After the sentences have been modelled, the individual models must be integrated. Provisions can be linked because they explicitly refer to each other, using a reference or because they use the same concepts. Provisions that are linked through references are easily detected, as the references have already been detected. The link between provisions that use the same concepts, and that use the same word(s) to identify these concepts is also easily detected. However, many sentences are linked through common sense relations between the concepts they use. For example, the concepts *gift* and *giver* are related, and thus sentences that use these two different concepts are related. However, since this relationship is not defined within the law, the models of these two sentences cannot be integrated automatically unless we add this knowledge to the process. Since the body of common sense knowledge is huge, the automated translation process is unlikely to have all knowledge that is needed available. This means that only specific parts of the integration step can be completed automatically.

As this integration step cannot be performed automatically, it is not possible to automatically create models of the law. However, automated tools can help reduce the knowledge acquisition bottleneck significantly, as they are successful in detecting the structure of a law, finding the references contained in that law, classifying the sentences and suggesting model fragments for those sentences. Furthermore, the intermediate results of the process also have their uses. For example, the recognition of references has been used in the construction of a semantic network of legislation for the Dutch Tax and Customs Administration, and the patterns identified for the classification of sentences are included in MetaVex, an editor for legislation. These applications are discussed in chapter 6.

In order to actually start using this method, it will need to be refined. The accuracy can be improved by gathering more patterns from a wider training set and by implementing various changes suggested in this thesis. Furthermore, legislative drafters can also improve the automated processing of laws. Several pieces of information in the law can be expressed in a more precise manner, which may also make them easier to read for humans.

Next to improving the accuracy, the method also needs to be expanded. With regards to the classification, it may be desirable to introduce several subclasses. The method also needs to be expanded for other documents than laws, such as royal decrees and case law. For this, patterns that are specific for those documents will have to be added.