# Exploiting Properties of Legislative Texts to Improve Classification Accuracy

Rob OPSOMER [a], Geert DE MEYER [b], Chris CORNELIS [c], and Greet VAN EETVELDE [a]

[a] *Ghent University, Dep. of Civil Techniques, Environmental and Spatial Management*
[b] *Flemish Institute for Technological Research*
[c] *Ghent University, Dep. of Appl. Math. & CS, Computational Web Intelligence*

**Abstract.** Organizing legislative texts into a hierarchy of legal topics enhances the access to legislation. Manually placing every part of new legislative texts in the correct place of the hierarchy, however, is expensive and slow, and therefore naturally calls for automation. In this paper, we assess the ability of machine learning methods to develop a model that automatically classifies legislative texts in a legal topic hierarchy. It is investigated whether such methods can generalize across different codes. In the classification process, the specific properties of legislative documents are exploited. Both the hierarchical structure of legal codes and references within the legal document collection are taken into account. We argue for a closer cooperation between legal and machine learning experts as the main direction of future work.

**Keywords.** Text classification, machine learning, legal classification, legislative documents, law

## Introduction

In recent years, there has been a growing interest in improving the accessibility of legislation. The EMIS Navigator web site[1], a project of VITO (Flemish Institute for Technological Research), is one example of this effort. It offers online access to environmental legislation applicable in Flanders, Belgium.

Legislation in the EMIS Navigator is organized into a hierarchy of legal subjects such as 'Procedures'. Here, these subjects are called 'indexes' or 'classes'. Users can browse through the hierarchy and find all legislation related to the subject of their choice. While this approach provides great usability, it has the disadvantage that legal experts have to categorize the legislation into the index hierarchy, an expensive and time-consuming process. To alleviate this problem, automated text classification by means of machine learning can be used.

While text classification has been applied to tasks in the legal domain before, it has never been tried, to the best of our knowledge, to exploit the specific properties of legislative texts in order to improve the classification accuracy. Furthermore, it remains unclear whether it is possible to build a model that generalizes across legal codes. This

[1] http://www.emis.vito.be/navigator

ability is very important, since the goal of automated classification in the EMIS Navigator is to train a classifier on existing, manually classified codes, and then to use the trained system to automatically classify parts of new codes as soon as they are published.

In this paper, we explore these questions. We assess the capability of standard machine learning methods to automatically classify environmental legislation into a complex legal index hierarchy and to generalize across legal codes, and we exploit the peculiarities of legal documents in order to improve the classification accuracy.

The remainder of this paper is structured as follows. In section 1, standard text classification techniques are explained. In section 2, the specific properties of legislative documents in the context of text classification are discussed. The data on which experiments are performed are described in section 3, and the experimental setup and results in section 4. These results are discussed and explained in depth in section 5. Measures to further improve them are proposed as future work in section 6, followed by a short conclusion in section 7.

## 1. Text Classification

The goal of text classification is to find a classifier $f$ that maps documents $d$, represented by a vector of features, to a class label $y$. The search for the classification function $f$ is usually performed by machine learning: the classifier *learns* the classification function from a set of class-labelled training examples, called the *training set*.

### *1.1. Features in Text Classification*

In text classification, the features (or dimensions) of the feature vector usually correspond to words occurring in the training set [3], in which case the feature vector is called a *bag of words*. In its most simple form, the features are binary and indicate the presence of the word corresponding with the feature in the document of interest. In our research, the more sophisticated tf-idf formula [5] is used. This formula generates, for a given word $w$ and document $d$, high weights if $w$ does not occur frequently in general, but does occur frequently in $d$.

### *1.2. Classification Algorithms*

In a hierarchical classification problem, i.e. a classification problem in which the classes are organized into a tree structure, one has the choice to either ignore this structure, and to consider the leafs of the tree as a flat set of classes, or to use the tree structure in the classification process. For flat classification, a wide range of machine learning algorithms exists. Here, the linear variant of the Support Vector Machine (SVM) algorithm [1] is used. The SVM algorithm has been shown to outperform other classifiers on many text classification tasks [2,3]. It is fast, very robust, even in high-dimensional feature spaces, and requires little or no manual parameter tuning.

For hierarchical classification, here, the tree structure is exploited by a divide-and-conquer strategy. A tree of flat classifiers is built, one for each internal node of the classification tree, plus one root node. To classify a document, first the root classifier decides to which of its children (i.e., top level classes) it belongs, and the document is passed to the classifier corresponding with the chosen child. This classifier then decides again to

which of its children the document belongs. This way, the document propagates through the class tree until it reaches a leaf node, which is the ultimate solution. It has been shown before that this approach yields slightly better results than the flat approach [4].

## 2. Text Classification and Law

Legislative texts exhibit interesting structural properties that can be exploited in the context of text classification. In this paper, two of them are discussed: the hierarchical structure of legislative texts, and legal references.

### *2.1. Hierarchical Structure of Legislative Texts*

Legal codes are, on a macro-level, highly structured documents. Although they are not always composed in the same way, usually they are organized into chapters, which branch off either into sections, further subdivided in articles, or directly into articles. All codes, chapters and sections carry a title, which often reveals a lot about the content of the corresponding documents.

We exploit this hierarchical structure by adding the titles of these books, chapters and sections to every part they contain. So, for example, for an article in code 'Decree General Regulations of Environmental Policy', chapter 'Targets and Principles', the words 'Decree General Regulations of Environmental Policy Targets and Principles' are added to the content of that article, before it is handed to the classifier.

### *2.2. Legal References*

In legislation, references such as 'see article xxx of code yyy' occur frequently. Obviously, these references convey a lot of information. We illustrate this with an example. Consider the paragraph below, §3, that belongs to the index 'Principles and Objectives of Environmental Policy/Principles of Environmental Policy/Integration Principle'. As a human classifier, we can easily see that §3 is about objectives, principles, and integration. However, we cannot determine that this paragraph is about environmental policy. If we look at the references, in this case §2, however, it becomes clear that this is about environmental policy.

> **§3** "The *objectives* and *principles* mentioned in §1 and §2 should be *integrated* with other fields in the determination and execution of the policy of the Flemish District. In execution of the policy, social-economical aspects, the international dimension and the available scientific and technical data are taken into account."
>
> **§2** "On the basis of an assessment of the various civil activities, the *environmental policy* strives to a high level of protection. It is based, among others, on the precautionary principle and the principle of the preventive action, the principle that environmental damage should be contained at the source, the standstill principle, and the principle that the polluter pays."

To exploit these references, first, they need to be recognized and parsed. After that, they can be used in the classification process.

#### 2.2.1. *Parsing of References*

Since references in legal texts follow a quite regular pattern, the automatic parsing of these references is, although definitely not trivial, a feasible task [6]. The parser of references used in this paper is an adapted version of the parser proposed by de Maat et al. [7]. On top of this parser, a module was built that searches the EMIS Navigator database for the legislative text referred to by the reference.

#### 2.2.2. *Use of References in Classification*

References can be used in two ways in the classification process [8]. In the first approach, when classifying a certain document $x$, all neighbours of $x$ (the documents referring to or referred by $x$) are concatenated to $x$. Optionally, all words in the neighbours can be preceded by a label, for example transforming 'soil' into 'REF_soil'.

In the second approach, the classification of $x$ is based on the class labels of the neighbours of $x$. If the class labels of the neighbours are unknown, a two-phase procedure is needed, in which at first the neighbours are automatically classified and thereafter their class labels are used in the classification of $x$.

## 3. Data

As data set, a part of the EMIS Navigator database, consisting of 1600 text objects, is used. The text objects, called 'law objects', are all part of Flemish, Belgian and European environmental and energy legislation, and are written in Dutch. The size of the law objects ranges from very small (parts of articles) to very large (complete chapters).

The data are categorized into an index tree with a maximum depth of six levels, and 230 leaf nodes. Law objects are classified into leaf nodes; they cannot be classified into internal nodes. To make the classification task more feasible, the depth of the classification tree is often constrained to a certain level, below which the hierarchical structure is collapsed into the nodes at the specified level. This is always indicated in the experiments.

The index tree consists of rather abstract legal subjects. The top level subjects are:

- Principles and Objectives of Environmental Policy
- Government
- Enforcement
- Procedures
- Instruments of Environmental Policy
- Instruments of Energy Policy

## 4. Experimental Results

A linear SVM (see subsection 1.2) with standard parameters was used as the classification algorithm. As document representation, a bag of words with tf-idf features was used. Stop word removal was performed, i.e. common, unimportant words as 'the' and 'an' were removed. Only words that occur at least five times in the training set were included in the bag of words. Tenfold cross-validation was applied in all experiments. Results are

compared with a simple baseline that assigns every object to the largest class in the training set. The *WEKA* toolkit [9] was used for preprocessing. For the SVM algorithm, the *LibSVM* [10] package was used.

### *4.1. Generalizing Across Codes*

Here the ability of SVMs to build a model that generalizes across codes is assessed. Two experiments are performed, each with the same standard setup discussed above, but with a different form of cross-validation.

In the first, standard tenfold cross-validation is used on all law objects. This implies that for every law object in the test set, on average 90% of the code containing the law object is in the training set.

In a second experiment, an adapted tenfold cross-validation procedure is used. Here, for each cross-validation split of the data in training and test set, the law objects of a code are either all in the training set, or all in the test set. This way, all objects in the test set are parts of codes that are unseen during the training phase, and the ability of the SVM model to generalize across codes is assessed.

The results are visualized in Figure 1, on different levels of the hierarchy. The figure shows that, as expected, accuracy drops when no training data of the same code is available, yet the drop being remarkably small (7% on the top level). This indicates that it is possible for a classifier to transfer knowledge across different codes, despite the different authors and publication dates of, and differences in vocabulary between, these codes. In the following experiments, the adapted cross-validation is always used.

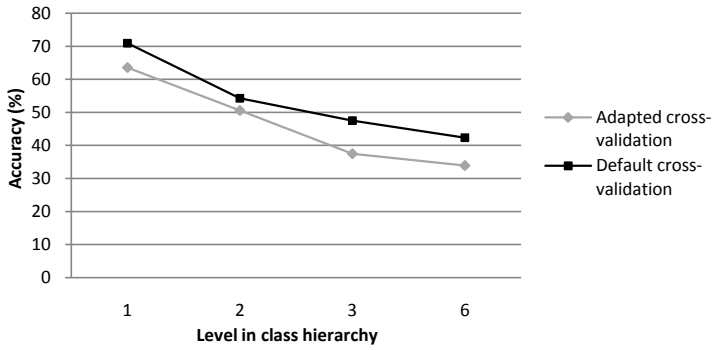


**Figure 1.** Accuracy at different depths of index hierarchy for the title and content of law objects, and their combination.

### *4.2. Exploiting the Structure of Legislative Documents*

Here, the structure of codes is exploited, as discussed above. Figure 2 shows the accuracy of the SVM classifier on different levels of the index hierarchy. The more sophisticated classifiers all clearly outperform the baseline. The addition of the title to the content of

the law object increases accuracy on all levels of the index hierarchy with 3% to 6%. It is stressed that the adapted cross-validation is used here. This implies that the classifier does not merely remember these titles from the training set and maps them one-to-one to legal subjects, but instead really uses the words from the titles in the determination of the index. This in turn indicates that it is very important for legislators to choose appropriate titles for codes and parts of codes to facilitate the automated classification process.

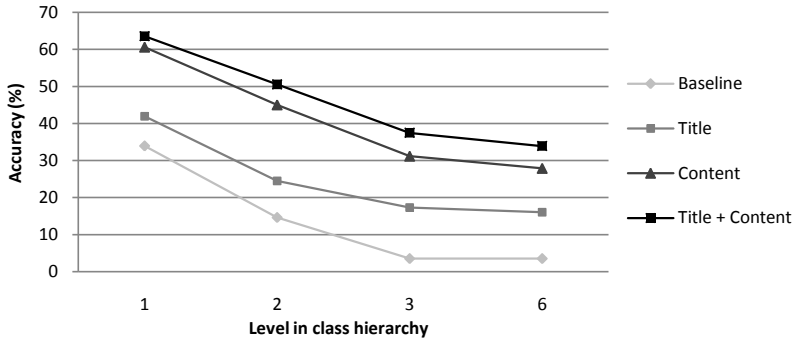


**Figure 2.** Accuracy at different depths of index hierarchy for the title and content of law objects, and their combination.

### *4.3. Legal References*

References are very well identified, parsed and linked to the part of the code they refer to. Estimates show that at least 85% of the references are identified, with a precision of at least 95%. These very good results can be explained by a combination of the already very good parser of de Maat et al. [7], the adaptations made to this parser in order to prepare it for Flemish environmental legislation, and the good performance of the module that searches the law objects corresponding to the references, as this module acts as a filter for faulty identified references. Because almost all references point to codes within the EMIS Navigator database, this parser enables the automated creation of hyperlinks between law objects. This could greatly enhance the usability of the Navigator web site.

To assess the value of references in the classification process, first, the value of the references per se is determined, after which they are combined with the original contents of the law objects. In order to do this, first, in a preprocessing step, all references are extracted, and for each law object, a list of neighbours is created. After this, either the content and the title, or the class labels of the neighbours are fetched. Finally, the original contents and titles of the law objects are deleted or merged with the fetched information from the references.

Figure 3 shows the results of the use of references per se in the top level classification process, for ingoing, outgoing and both directions of references. It can be seen that references do contain information. Both the classifier using the class-label of neighbours and the one using the contents and titles of neighbours outperform the simple baseline.

Furthermore, the use of the class label of the neighbours yields better results than the use of the contents and titles of the neighbours. These findings are consistent with the results of Chakrabarti et al., who performed similar experiments on web sites and patent databases [8]. However, it should be noted that it was assumed in our experiments that the class labels of the references were always known, which is obviously not always the case in practice.

Despite this stand-alone value of references, it turned out to be extremely difficult to exploit this value to outperform the classifier only using titles and contents of the original law objects. Various approaches in combining original and neighbour law objects were pursued, including labelling the neighbour contents, using class labels of neighbours, and setting different weights to neighbour and original contents or titles. The only improvements were obtained by combining the class labels of the neighbours with the original contents and titles. However, these improvements were very small (about 1%), and would probably disappear as soon as the unrealistic assumption of the known neighbour class labels was removed.

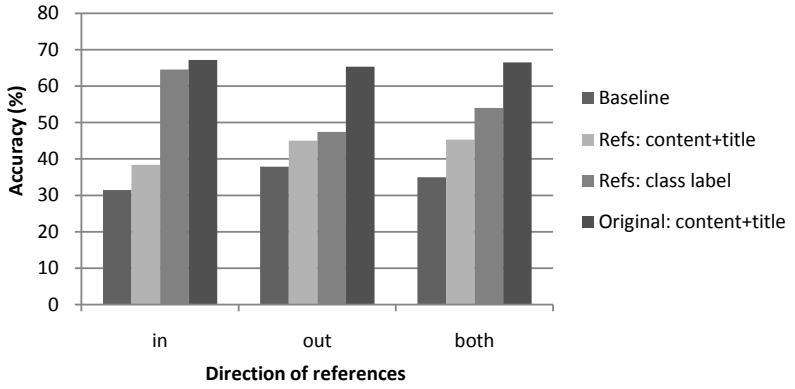


**Figure 3.** Accuracy for use of references in classification at top level of index hierarchy, compared to baseline and original data.

### *4.4. Hierarchical Classification*

It becomes immediately clear from Figure 4 that the hierarchical classifier does not work well in the concerned problem. The hierarchical classifier performs worse than the flat classifier; and the more the hierarchical structure of the classes is exploited (i.e., the deeper in the class hierarchy), the worse the results get. This is not in line with previous research on similar classifiers [4], and indicates that the class hierarchy in our problem is not very consistent. Of course, if the classes on higher levels do not group the lower level classes in a logical way, 'exploiting' the hierarchy leads to a deterioration of the results.

## 5. Discussion

The best result obtained was 63.6% accuracy on the top level task with six indexes. While this is definitely better than the simple baseline (33.9% accuracy) and random

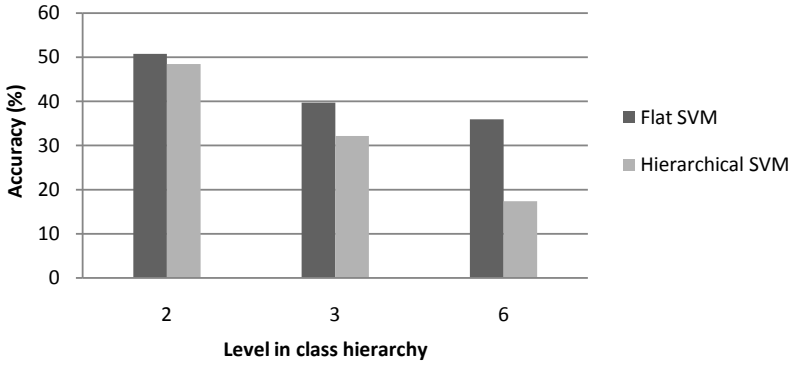


**Figure 4.** Accuracy at different depths of index hierarchy for standard, flat SVM and hierarchical SVM

(16.7% accuracy on a task with six indexes), it is not enough to be of practical use. We can claim, however, that we have employed state of the art text classification techniques to solve this problem, and, on top of that, have exploited the structural properties of the legislative texts. So why, then, are the results what they are? In previous research, Moens mentioned the abstract concepts legal classification hierarchies contain, and the high variety of words and phrasal expressions these concepts are expressed by, as an important cause for the disappointing results in legal classification tasks [11].

Analysis of the results obtained in this paper indicate, however, that a large part of this complexity in legal classification tasks is not inevitable, and could instead rather easily be avoided by closer cooperation between legal and machine learning experts. Now, in the Navigator project, as is all too often the case, there was a simple two-step procedure in which at first legal experts designed an index hierarchy and did some manual classification, after which the task was handed over to machine learning experts. The legal experts did the first part of the process without ever thinking about machine learning - indeed, initially even without knowing about the automation efforts. In the remainder of this section, we discuss three important causes for the disappointing results, which could all be easily solved by cooperating machine learning and legal experts.

A first and important cause is found in the lack of semantic consistency in the grouping of lower level indexes into higher level indexes, especially at the top level. This statement is supported by various observations. A first observation is the disappointing result of the hierarchical classifier. Second, the accuracy on the top level (six classes), 63.6%, is low compared to the accuracy at the second level (40 classes), which is 50.5%. So, while the second level has almost seven times more classes than the top level, the accuracy at the second level only drops with 22% relative to the accuracy at the top level, while the error rate on the top level is a much larger 36.4%. A third indication is the confusion matrix at the top level, which is shown in Table 1. As can be seen, class nr. 5, or 'Instruments of Environmental Policy', causes severe confusion in the other classes. It is assumed that class nr. 5 contains so many and such a variety of law objects, that it becomes extremely hard for the classifier to distinguish between this and the other classes. A preliminary test, in which class nr. 5 was removed from the classification task, lead to an increase of accuracy to 74.8%. Obviously, 74.8% accuracy for a five class task is

more satisfying than 63.6% accuracy for a six class problem. Of course, such a coarse solution would probably be impossible in practice, but it indicates the jump in accuracy that can be achieved by changing the top level classes.

**Table 1.** Confusion matrix for top level of index hierarchy.

| | | Real class | | | | | |
|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 |
| Predicted class | 1 | **180** | 3 | 2 | 3 | 32 | 6 |
| | 2 | 10 | **175** | 25 | 16 | 27 | 10 |
| | 3 | 2 | 11 | **160** | 13 | 11 | 2 |
| | 4 | 1 | 18 | 8 | **93** | 31 | 2 |
| | 5 | 113 | 94 | 27 | 95 | **533** | 74 |
| | 6 | 7 | 5 | 6 | 1 | 18 | **107** |

A second, important factor with regard to the classification results is the partial inability of the classifier to learn a model that generalizes across different codes. On the top level, a 7% accuracy decrease was observed when the classifier only had to its disposal training data of codes different from the one it had to classify.

Finally, the size of law objects can have a negative influence on the classification accuracy. Analysis has shown that classification accuracy is inversely correlated to the size of law objects, and that up to 7% accuracy gain could be achieved by only using small chunks of texts as law objects. Large law objects (e.g. complete chapters) are assumed to contain information that is too diffuse to be classified into one class.

## 6. Future Work

In future work on the EMIS Navigator, the problems discussed above should be dealt with. Machine learning experts should work close together with legal experts to change the index hierarchy, based on the insights explained above. For example, it could be proposed to either split the top level class 'Instruments of Environmental Policy' into smaller, crisper defined classes, or to spread the subclasses over the other top level classes. Also, legal experts could be advised against the use of large law objects. These measures will not only improve classification accuracy significantly, but also improve usability of the Navigator web site. Both a more logical and consistent index hierarchy and a larger number of small, crisply defined law objects will lead to greater usability. Besides these measures, active learning could be introduced to alleviate the problems the classifier still has with transferring knowledge from one code to another. For SVMs, training with the help of active learning on only 10-20% of the original training set size could produce results as good as with the complete training set [12]. So, in our case, 10-20% manual classification of new codes, guided by active learning, could lead to an accuracy gain of 7% at the top level.

Furthermore, it would be interesting to look at other legal classification tasks, to see in which degree the same problems reoccur in other tasks. We believe that in any classification of legislative texts, there will be an initial mismatch between the manual index-building and classification on the one hand, and machine learning on the other hand. As shown above, these problems could be solved by closer cooperation between machine learning and legal experts.

## 7. Conclusions

It was shown that automated classifiers using machine learning can learn the properties of different classes of legislation, and transfer this knowledge across different codes, although not with an accuracy high enough to be of practical use. The structure of legal codes was successfully exploited in the classification process. References were parsed very well. However, these references could not be successfully used in the classification context. The findings were discussed and weighed in view of future work. Closer cooperation between legal and machine learning experts was encouraged.

## Acknowledgments

Chris Cornelis would like to thank the Research Foundation–Flanders for funding his research.

## References

[1] V. Vapnik. The nature of statistical learning theory. Springer-Verlag, 1995.

[2] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In ECML '98: Proceedings of the 10th European Conference on Machine Learning, pages 137-142, London, UK, 1998. Springer-Verlag.

[3] F. Sebastiani. Machine learning in automated text categorization. ACM Computing Survey, 34(1):1-47, 2002.

[4] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning, pages 170-178, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[5] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. Information Processing and Management, 24(5):513-523, 1988.

[6] M. Palmirani, R. Brighi, and M. Massini. Automated extraction of normative references in legal texts. In ICAIL '03: Proceedings of the 9th international conference on Artificial intelligence and law, pages 105-106, New York, NY, USA, 2003. ACM.

[7] E. de Maat, R. Winkels, T. van Engers. Automated detection of reference structures in law. In T. M. van Engers, editor, Legal Knowledge and Information Systems. Jurix 2006: The Nineteenth Annual Conference, volume 152, pages 41-50, Amsterdam, The Netherlands, 2006. IOS Press.

[8] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. SIGMOD Rec., 27(2):307-318, 1998.

[9] I.H. Witten and E. Frank. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Fransisco, 2nd edition, 2005.

[10] C.C. Chang and C.J. Lin. LIBSVM: a library for support vector machines. 2001, Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[11] M.-F. Moens. Innovative techniques for legal text retrieval. Journal of Artificial Intelligence and Law, 9(1):29-57, Springer, Netherlands.

[12] G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning, pages 839-846, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.