MASTER

Mining Local Process Models from Dutch Legislative Texts

van der Veen, M.F.

Award date:
2021

Department of Mathematics and Computer Science
Process Analytics Group

Mining Local Process Models from Dutch Legislative Texts

Master Thesis

M.F. van der Veen

Supervisors:
dr. N. Sidorova (TU/e)
R.W. Lucas, MSc (Deloitte)

Assesment Committee
dr. N. Sidorova (TU/e)
R.W. Lucas, MSc (Deloitte)
dr. I. Kurtev (TU/e)

26 November 2021

Abstract

With an increase in (textual) data nowadays, methods to analyze such data are developing as well. One of the domains that could benefit from automated analysis of text data is the legal domain, since this domain is built on written texts. In this work we focus on legislative texts, as these are information-dense and complex by their nature. Due to the complexity of these texts, it is challenging for readers to comprehend them. Therefore, mining the patterns and rules and visualizing them could assist users in their tasks. Previous work has not been able to address this problem on a multi-sentence level yet. In this work we propose a method that uses relations between sentences to extract rules and patterns. We also propose a visualization method that provides an intuitive and understandable view. We present our framework as a proof-of-concept feature within Deloitte's Moonlit platform, which aims to increase the quality and efficiency of legal services. Our approach consists of three modules. First we extract phrases, specifically for the legal domain, indicating relations. Then we use these phrases to extract rules and connect them. Finally, we compare four visualization methods and select the most suitable one for visualization. Our results demonstrate the feasibility of extracting these rules from legislative texts using domain expert knowledge, automated processes and additional assumptions. Some limitations need to be encountered to improve the extracted rules in terms of correspondence with legislative texts. Moreover, we proposed a method to select the most suitable visualization method. Our proposed method opens up possible future opportunities in terms of enhancement through case law and compliance checking.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, we live in a digital world in which the volume of both structured and unstructured data is increasing. Consequently, methods for handling and analyzing data are evolving as well. Specifically, the analysis of textual data through Natural Language Processing (NLP) and text mining techniques are developing [1, 2]. NLP is a research area that investigates the ability of computers to understand and manipulate natural language text or speech through automated techniques [3]. Such techniques can be applied to domains ranging from the medicine [4] to finance [5]. One of the areas that could benefit from the application of NLP techniques is the legal domain, since knowledge in this domain is built on textual data. Therefore, automatic analysis of legal texts is an active research field with applications in different areas ranging from document annotation and legal text generation [6] to verdict prediction [7]. These NLP techniques can be applied to multiple types of legal texts, such as case law or legislative documents.

1.1 Context

In this work, we focus on legislative texts, as these are complex and information-dense by their nature. Due to the complexity of these texts, it is challenging for readers to comprehend them. Tackling this challenge could help attorneys and legal practitioners improve their efficiency in analyzing and using these texts. Moreover, it could make legislation more transparent for non-experts, as it makes complex legislative texts easier to comprehend and thus more accessible. Therefore, the aim of our work is to demonstrate the feasibility of extracting patterns and rules from legislative texts and the visualization of these patterns.

These extracted patterns and rules are then connected to form conditional relation diagrams. These diagrams describe the control flow within the legislative text in the form of a flowchart or a decision tree. In the remainder of this work we call such control flow diagrams, *local process models*.

Besides an academic contribution, this work also delivers a practical solution by means of a proof-of-concept feature within Deloitte's Moonlit platform[1]. This platform's purpose is to increase the quality and efficiency of legal services, relieving stress on the legal system and improving access to justice across Europe. By extracting and visualizing control flow within legislative texts, our work contributes to improving efficiency for legal practitioners and opening up access to legislation for non-experts.

For our work, we require some prerequisites before we can start mining local process models from legislative texts. As human beings comprehend text and language through learned patterns and structures, it is important to understand what those patterns mean and how these are presented in a text. One type of pattern that indicates relations in a text is the usage of *signal phrases*. These signal phrases are words or phrases that indicate a certain relation in a text. For example, in the sentence "If it rains, we will get wet", we marked the relevant signal phrase by underlining it. Here we see that the word "if" is underlined and that it indicates a relation. Specifically, the word "if" is indicating a condition, namely that in the situation where it rains, we will get wet.

These signal phrases provide extremely valuable information about a text. For human beings, recognizing such relations seems trivial, but for computers this is not the case. Therefore, it is important for a machine to recognize such signal phrases. As we are focusing our work on the legal domain, we would like to find signal phrases specifically used in this domain. Since signal words and phrases in the legal domain differ from regular language, we cannot simply take a school's textbook and use those signal phrases. Hence, we need to utilize legal texts to extract these phrases before we can start the extraction of local process models.

In the process of extracting patterns and local process models, we utilize dependency parsers. The goal of dependency parsing is to construct a labeled dependency graph that illustrates the semantic dependencies within a sentence [8]. Figure 1.1 illustrates an example of a parsed English sentence [9]. Such parsers could assist in understanding sentence structures.

[1] `https://www.moonlit.ai/`

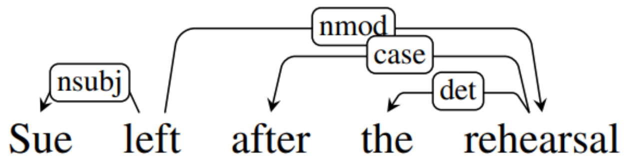

Figure 1.1: Dependency graph for an English sentence. Nsubj indicates the nominal subject; nmod indicates a nominal modifier; case indicates prepositions, postpositions and other case markers; and det indicates a determiner

1.2 Research Questions

Current state-of-art rule and pattern mining techniques for textual data are able to mine the structure within a single sentence [10, 11]. In this work, we try to extract additional information from texts by also looking at the relation between sentences. Previous works have shown promising results in the extraction of these relations [12, 13]. The authors of [13] noted, however, that model quality could be improved using more heuristics and rules. Model quality can be measured in terms of syntactic quality, how the model is modelled according to the syntax; and semantic quality, which states how well a model describes the modelled domain. An important focus point is hence to investigate how sentences refer to one another. By identifying and extracting these relations, modelling full legislative texts becomes more feasible. To assist legal practitioners and other users in understanding legislative texts, it is important to present the knowledge gained from legislative texts in an intuitive manner. Hence, this work also focuses on the visualization of the processes within legislative texts.

Our goal is to develop a framework that could solve a two-fold of problems, namely to extract local process models from legislative texts and to visualize the underlying local processes in a meaningful way. Therefore, we answer two main research questions in this work, which we call **RS1** and **RS2** respectively.

> **RS1:** "How to mine local process models from legislative texts?"
> **RS2:** "In what way can we provide the user with knowledge about the local processes in legislative texts?"

1.3 Challenges

In answering **RS1** and **RS2**, some challenges need to be encountered. The first challenge ties to **RS1**. This challenge is the recognition of relevant events and activities in a text and the relation between these events and activities. An *activity* can be defined as a trigger in the process. We call the following result of an activity the *event*. To find the relation between events and activities we could utilize the signal phrases that we presented earlier.

A constraint to the representation of the events and activities in our local process models is the language. As legal practitioners often do not have a strong mathematical background, we cannot use formal mathematical notations such as linear temporal logic (LTL) [14], Quanitfied Regular Expression (QRE) [15] and computation tree logic (CTL) [16]. Having to use another notation presents another challenge in answering **RS1**, as it constraints the way in which the model will be expressed.

Another challenge is the visualization of underlying processes in legislative text. This challenge is related to **RS2**. The visualization needs to be straightforward and comprehensible for its users. It needs to be as intuitive as possible and this could be challenging, as legislative texts are known to be complex.

1.4 Approach

To answer the stated research questions **RS1** and **RS2**, we propose a framework that consists of several modules. In order to answer **RS1**, we introduce modules 1 and 2. For answering **RS2**, we introduce module 3. Figure 1.2 shows the modules in our framework and the steps taken within each module. We demonstrate the use of our approach through a case study using Dutch legislative texts.

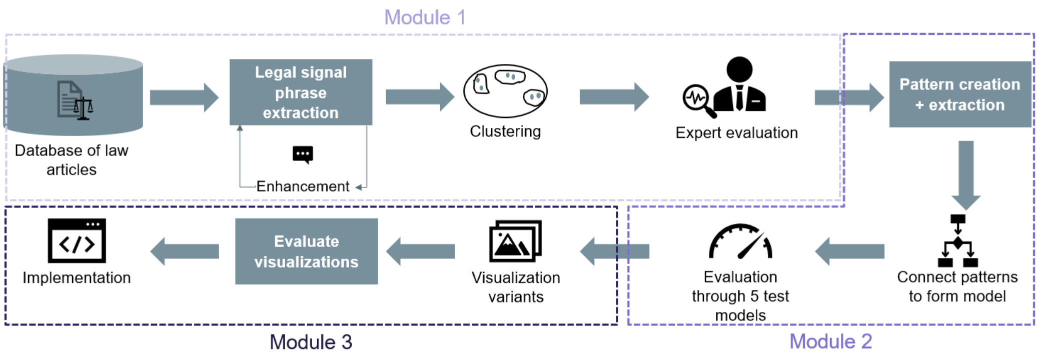

Figure 1.2: General framework to mine local process models from legislative text

In the module 1, we extract the relevant signal phrases that are required to mine the patterns and rules in later modules. Extraction of relevant signal phrases is done semi-automatically using domain expert knowledge. The extraction process is iterative where in each iteration, domain knowledge is utilized. This ensures that the number of false positives and false negatives are minimized. After extraction, signal phrases are clustered to their respective category. Finally, interview sessions with domain experts are conducted to remove false positive phrases. Within the interview sessions, consistency of the answers is checked to ensure validity of the interview results.

Module 2 extracts local process models from legislative text using the signal phrases mined in the first module. Patterns were created in this module based on how signal phrases were used in legislative text. To narrow the scope of our project, we focus only on one type of signal phrases based on domain expert input and analysis on usage of signal phrase types. After the creation of these patterns, the conditions are mined and connected to one another to form local process models. These local process models are then evaluated through semi-structured domain expert interviews.

Visualization takes place in module 3. Here, we use the knowledge gained from the local process models in module 2 to visualize them in an intuitive manner. Four candidate visualizations were made and evaluated with domain experts that can be considered as potential users. Based on this evaluation, a final visualization method is selected and implemented. The final implementation is then presented as a proof-of-concept tool.

1.5 Outline

To achieve our goal of mining local process models from legislative text, we present the modules of our framework in the remainder of this work. Chapter 2 discusses our signal phrase extraction module and its evaluation. After the extraction of signal phrases, patterns and local process models will be created and evaluated as discussed in Chapter 3. Thereafter, we use four visualization methods to display the mined local processes and evaluate these in Chapter 4. Finally we draw conclusions and provide recommendations for future work in Chapter 5.

Chapter 2

Signal Phrase Extraction

This module aims at the semi-automatic identification and categorization of words and phrases indicating conditions and causal or temporal relationships between activities in legislation documents. We call these words and phrases *signal phrases*. This module consists of 3 steps, as shown in Figure 1.2. In this chapter we first present preliminary knowledge required for understanding this module. Then we discuss related works. Thereafter, we introduce the methodology used in our module. Then we evaluate our module steps and present the results. Finally, we draw some conclusions on the results and discuss implications.

2.1 Preliminaries

In this section we introduce the terminology and knowledge that is required for this module.

- **N-gram:** An N-gram is a slice consisting of N words from a longer sentence string [17]. For example, when we have the sentence "I see Joe", we can split it to the following 1-grams: "I", "see", "Joe". When we split it to bi-grams we get the slices "I see", "see Joe". In case we split this sentence to tri-grams we get the slice "I see Joe". Slice sizes continue to increase with increasing N.

- **Part-of-speech (POS) tag:** POS tags indicate the semantic tag of a word in a sentence [18]. Examples of POS-tags are ADJ (adjective), NUM (numeral) and PUNCT (punctuation). These examples are derived from the Universal POS tag database[1].

[1]https://universaldependencies.org/docs/u/pos/

- **Token:** A token is an atomic (hence it will not be broken down to smaller parts) sub part of a input sentence [19]. In this work we define a token as one word.

- **Word Embedding:** Word embeddings are vector representations for words [20]. Vector representation for words enables the computer to also include context related information of a word.

- **Corpus:** A corpus is a collection of written or spoken material stored digitally and it is used to find out how language is used[2]. In this work we refer to the corpus as the database of text documents.

2.2 Related Works

Signal words and phrases are language specific. Efforts were made in the past to provide support in mining them for specific languages [21, 22]. The authors in [21] created an NLP pipeline to annotate Bulgarian legislative texts. This pipeline is able to annotate POS tags, utilize universal dependency parsing, annotate Noun phrases, Named Entities and Interactive Terminology for Europe (IATE) terms. The authors in [22] created a semantic extraction method for German legal documents. Here the authors were able to extract information such as the year of dispute and the extraction of legal definitions and contexts of legal terms in judgements. Besides efforts in specific languages, multi-lingual approaches were also made [23]. The authors in [23] created a legal knowledge graph that covers some jurisdiction and gives a coverage for a few languages. It shows the potential of creating multi-lingual legal information extraction tools. However, the developed method only covers very specific domains, is extended to limited jurisdictions and covers only a small number of languages.

Ontologies could be a helpful tool in information retrieval because these provide a knowledge base that could be used for the extraction of signal phrases. Several efforts have been made in creating ontologies specifically for the legal domain [24, 25]. These ontologies do not contain signal phrases yet, however, our work could possibly contribute to such ontologies as well.

2.3 Methodology

Figure 2.1 shows the steps taken in this module to extract legal signal phrases.

[2] `https://dictionary.cambridge.org/dictionary/english/corpus`

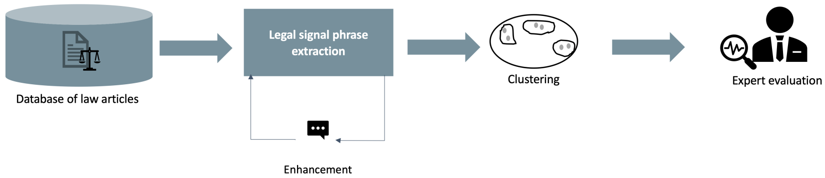

Figure 2.1: Steps taken in module 1

The goal of step 1 is to extract signal phrases from legislative texts. In step 2, we want to cluster these phrases to their respective phrase type. Finally, in step 3 we evaluate the results with domain experts to remove false positives and verify cluster quality.

2.3.1 Extract potential phrases

Our first step is to extract potential signal phrases. The goal of the extraction is to obtain signal phrases that are typical for the legal domain, but are not specific for a subdomain within the legal domain, e.g., tax law, civil law. To achieve this goal often occurring (high frequency), subdomain aspecific phrases were extracted. It is desirable to extract subdomain aspecific phrases because these signal phrases, i.e., phrases that describe a process, should be common for all legal subdomains. Whenever a phrase is too specific for a legal subdomain, it is not general enough to describe a process. This subsection explains how we attempted to extract these signal phrases.

Before extraction we created a dataframe, a table storing the data, with N-grams as rows and laws at the columns. The values within the cells of this dataframe are the counts of occurrence of this N-gram within a certain law. Our N-grams were extracted using the `CountVectorizer` function from the scikit-learn package[3].

The extraction is based on thresholds for *total frequency* of N-grams with $n \leq 3$ and the *coverage*, this is the percentage of laws in which a signal phrase occurs. Sufficiently high coverage ensures that signal phrases are general for legal texts and span across multiple legal domains and laws. This reduces the number of false positives in the form of expressions frequent within certain legal areas but not used in other areas. To illustrate false positives, we take the example phrases "omzet" (revenue) and "indien" (if). The former is a phrase that would have a high frequency in legal subdomains such as sales law or administrative law. However, in other subdomains this would not be a frequent phrase. This is a typical false positive phrase that will

[3] `https://scikit-learn.org/stable/index.html`

be excluded from our search when considering coverage. The latter phrase "indien" (if), is an example of a phrase with both a high frequency and a high coverage, as this phrase is used both often and in many laws. We do not include false positive expressions, such as "omzet" (revenue), as they do not necessarily carry the temporal or conditional relation required for our work. Additionally, stop words were discarded from our initial extraction results. Stop words were derived from the NLTK package[4]. Besides the stop words sourced from NLTK, we also discarded phrases such as "Koning der Nederlanden" or "wij Beatrix bij". Phrases like these were discarded, as these are noun phrases, specifically indicating persons, and therefore do not provide relevant insights for our work.

To enhance our extraction method, experts were enquired to curate a list of signal phrases that are typically found in legislative texts. This list was then analyzed on properties such as total frequency, coverage (within our dataset of legislative texts) and POS tags. These tags were selected on top of total frequency and coverage, as these provide insights into how phrases are used in sentences since POS tags denote their grammatical meaning. For N-grams where $n > 1$, POS tags were generated by doing a majority vote on the POS tags for each 1-gram. In case of no majority, the POS tag of the first 1-gram was selected. The enhancement process is an iterative process where in each iteration additional constraints on the extraction are imposed.

In our first iteration, we analyzed the properties of total frequency and coverage. The goal of this first iteration is to do a global initial search, which we can enhance by adding POS tags in later iterations. We took the total frequency and coverage for the phrases in the expert curated list and calculated μ (mean) and σ (standard deviation) for each property. Assuming that the distribution of the data describing the total frequency and coverage is normal, we can take the values of $\mu - 2\sigma$ as threshold to ensure that at least 95% of our population of signal phrases would be included in our search. To test normality, we used the Shapiro-Wilk test, as has been shown that this the most powerful normality test [26].

In our next iterations, we added constraints on POS tags per iteration to make the search more specific, i.e., decreasing the number of false positives. First we analyzed which POS tags are occurring the most in our expert curated list. We used the most frequent POS tag as additional constraint. Then we take the second most frequent POS-tag and add an additional constraint. We continue doing this until the remaining POS-tags have a count that is not representative of the tags in the expert curated list. We set the threshold on 10% of the curated list. For illustration, when the curated list consists of

[4] `https://www.nltk.org/`

40 phrases, then the POS-tags that occur less than 4 times are not included. On top of adding only POS tags as constraints, we made combinations of POS tag constraints and threshold or coverage constraints. For example, including phrases with POS tag VERB AND coverage > 0.9. This process was repeated until not too many phrases (< 400) are selected, while maintaining an adequate (> 0.6) recall score. Recall can be calculated as shown in Equation 2.1, where tp are true positives and fn are false negatives. The maximum of 400 phrases has been decided as it will otherwise take too many man-hours to evaluate the phrases in later stages.

$$\text{Recall} = \frac{tp}{tp + fn} \tag{2.1}$$

2.3.2 Embedding and Clustering

After the extraction step, we first generalize the extracted phrases before embedding them. Generalization of extracted phrases is done by identifying similar phrases, e.g., "van de" and "van het" (both translated to "from the"). Such phrases were then generalized to "van + DET", where DET is the POS tag for determinant. Generalization of such phrases was done automatically by keeping the first token in a bi-gram, whenever the last token has the POS-tag "DET".

Embedding of the phrases is done using a language model. Three embedding methods were used, namely Tensorflow's Multilingual Universal Sentence Encoder presented in [27], Dutch Word embeddings introduced in [28] and embeddings trained on our corpus of Dutch laws using fastText[5]. The choice of embedding is made based on experiments where it is checked whether a sample of words that are known to be synonyms from[6], receive a high similarity score to one another. Two words can be considered similar to one another when the similarity score is > 0.5. Similarity is calculated using the cosine similarity as given in Equation 2.2, where $\mathbf{A}$ and $\mathbf{B}$ are vector representations of text.

$$\text{similarity} = cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} \tag{2.2}$$

Then after embedding, we cluster the embedded phrases. Clustering is done by first defining a fixed number of labelled clusters and predefined centroids, which is a central point of a cluster, of these clusters. The clusters and its labels are based on signal phrase types found in [29]. These phrase

[5]`https://fasttext.cc/`

[6]`https://synoniemen.net/`

types are: "opsommend verband" (summation), "tijdsverband" (temporal relation), "tegenstelling" (contrasting), "vergelijking" (comparison), "toelichting" (explanation), "voorwaardelijk" (conditional), "verklaring" (causation), "samenvatting" (summarization) and "conclusie" (concluding). On top of these phrase types, we add the type "overig" (other). A centroid is calculated by taking the mean of embeddings of the phrases for each phrase type [29]. To assign the phrases to its corresponding cluster, we calculate the similarity with each cluster centroid. To calculate similarity, we again use the cosine similarity in Equation 2.2. The phrase is assigned to the cluster that is most similar.

2.3.3 Expert interviews

Domain experts, which are legal consultants at Deloitte's Tax and Legal department, were interviewed to detect false positive phrases. Besides false positive phrase removal, cluster assignment was checked as well. The interviews were taken through Microsoft Teams. While sharing our screen, we asked for each presented phrase whether a phrase was suitable for legislative texts, whether a phrase was a connecting phrase and whether the phrase was assigned to the correct cluster. Moreover, interviewees could suggest alternative or additional clusters. The interview template used can be found in Appendix A.

To check the consistency of responses in this step, we designed corresponding measures. Each interviewee received a set of phrases consisting of two parts: one is the same for all of them and the other is distinct. On top of these constraints we ensured in our sample creation that each phrase is evaluated by two interviewees. In our evaluation phase we could evaluate whether the sample part that is the same for each interviewee is consistent. Whenever this is consistent enough, we could assume that this consistency propagates through the other phrases and that the provided answers by the interviewees are sound.

The phrases were presented to the interviewees sorted by the cluster. Therefore, all phrases that were assigned to the same cluster - and hence have more or less the same semantic meaning - were presented close to one another. This ensures a shorter response time when validating the cluster of a phrase [30]. A risk of using this method is that information bias could be introduced and it poses the risk of losing focus and flat-lining (consistently providing one answer). However, the benefit of a shorter response time outweigh the risks of bias, as shorter response times are beneficial for the feasibility of this method. Presenting the phrases sorted on cluster enables using a greater number of samples during the limited time span of the interviews.

2.4 Evaluation and Results

This section covers the evaluation methods and results for each step in this module. The signal phrases were extracted from 1413 Dutch laws originating from the Dutch government website[7].

2.4.1 Extract potential phrases

In our first step, we initially enquired 3 domain experts to curate a list of signal phrases. With their assistance we found a combined list of 36 phrases. These 36 phrases were analyzed and used for our initial extraction attempt. We call this list of 36 phrases our current "ground truth". We analyzed our "ground truth" list on its total frequency and coverage in our dataset of 1413 laws. Moreover, we analyzed their dominant POS tags. Figure 2.2 shows a boxplot for the total frequency values for our "ground truth" list. It can be seen that some outliers exist and therefore it seems that these scores are not normally distributed. In Figure 2.3, we can see no outliers and that the box is more balanced. This indicates that coverage values seem to be normally distributed. To verify these initial impressions, we run the Shapiro-Wilk test on both variables. The results are shown in Table 2.1. The Shapiro-Wilk test tests for a null hypothesis, which states that the distribution of the sample is normal [26]. It can be seen that we reject the null hypothesis for total frequency, as the p-value is extremely small. Therefore, we can state that the total frequency is not distributed normally. For the coverage, however, we do not reject the null hypothesis. Hence, we can conclude that the coverage is distributed normally.

Based on these statistics, we cannot simply take the $\mu - 2\sigma$ as minimum threshold for total frequency. For the coverage we found that $\mu - 2\sigma$ is extremely close to 0. This is also undesirable as the risks of false positives that are only used in some legal domains, but not in others, increases. To prevent including too many false positives in our initial search attempt, we decided to set our initial thresholds for total frequency to 1000 and for coverage to 0.25. These values include values greater than the first quartile and therefore ensure at least 75% of the desired phrases to be included in our search. This trade-off is made as having too many false positives could decrease the feasibility of the evaluation phase in step 3.

After this initial search attempt, 1453 phrases were found. As next step, we analyzed the frequency of POS-tags in our "ground truth" list as seen in Figure 2.4 It can be seen that the adposition (ADP) POS-tag occurs the

[7] `https://wetten.overheid.nl`

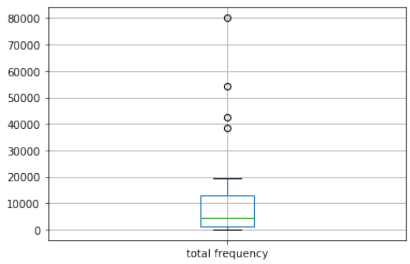

Figure 2.2: Boxplot for total frequency

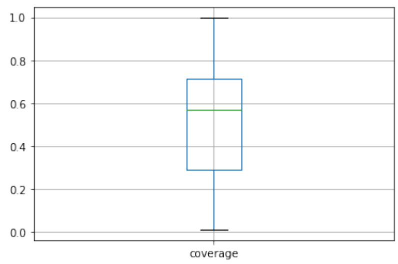

Figure 2.3: Boxplot for coverage

Variable	**Statistic**	**P-value**
Total frequency	0.63	0.13
Coverage	0.95	$2.8 * 10^{-8}$

Table 2.1: Shapiro-Wilk test results for total frequency and coverage

most. Hence, this is the first filter that we add to our search. After the inclusion of only selecting ADP tags, we found 273 phrases. Then we add an additional constraint of also including adverb (ADV) tags, this resulted in 342 phrases found. From Figure 2.4, we see that the third most occurring tag is VERB. However, when we simply add all phrases with tag VERB, we get too many false positives. Thus, we decided to add a nested constraint of VERB and coverage > 0.9, as this includes the most occurring VERBs from our ”ground truth” list. We did not use a lower coverage because then the risk of capturing false positives increases. After adding this constraint, we found 349 phrases. The next most occurring tag was SCONJ, again we combined this tag with a coverage constraint. The combined constraint was SCONJ, coverage > 0.35, as all SCONJ phrases in our ”ground truth” list are above this coverage level. Finally this resulted in 369 phrases. We stopped with adding more tag constraints as the counts for these tags were too low (< 2).

After this enhanced search, we calculated the recall score as stated in Equation 2.1. With the search result of 369 phrases, we had a recall score of 0.63. This recall score is higher than our set threshold of 0.6. The recall could have been higher, with the risk that more false positives are included as well. The recall score can be explained because the POS tag constraints ensured that not all ”ground truth” phrases were included. To continue with a complete list of phrases, we added the remaining phrases to our list of extracted phrases.

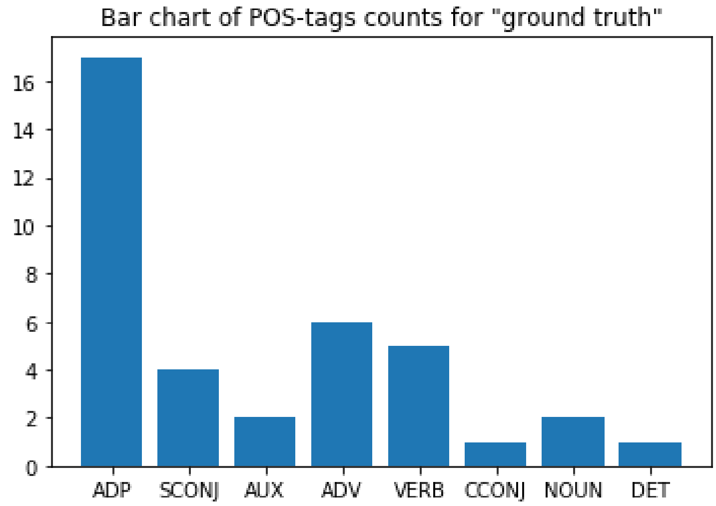

Figure 2.4: Bar chart illustrating counts of POS-tags for our "ground truth" list. ADP is the most occuring POS-tag.

2.4.2 Embedding and Clustering

The prerequisite of this step is to generalize the extracted signal phrases first. After generalization, the list of N-grams was reduced to 322 phrases.

Before clustering, an embedding method needed to be selected. To facilitate the selection, 3 embedding methods were evaluated. We evaluated the 3 embedding methods using the word "indien" (if) and its 5 synonyms from [8], namely: "zo", "mits", "als", "ingeval", "wanneer". We tested the similarity for the 3 embedding methods, where for each synonym it is checked whether the cosine similarity is > 0.5. If that is the case, the synonym is classified correctly. Hence, an optimal score will be achieved if for all 5 synonyms, the cosine similarity results in > 0.5 as defined in our approach. Table 2.2 shows the accuracy scores for each embedding method. Here we see that the Multilingual Universal Sentence Encoder scores best using this sample. Thus, we continue with clustering using the embeddings generated by this language model.

Using the Multilingual Universal Sentence encoder, we embedded the phrases and assigned them to their corresponding cluster. To check the cluster quality, external evaluation was required to see whether phrases were assigned to the correct cluster. For this evaluation we used a subset of extracted phrases that are synonyms to one another. To determine the synonyms we used the same synonym database as mentioned above. Phrases

[8] `https://synoniemen.net`

Embedding method	Score
Multilingual Universal Sentence Encoder	0.8
Self-trained embeddings	0.6
Dutch embeddings	0.6

Table 2.2: Accuracy scores for three embedding methods

that are synonym to one another were assigned to the same class. Then to check the cluster quality, we would expect that all synonyms, i.e., phrases in the same class, would get the same cluster assigned. To measure this, we used the Purity measure in Equation 2.3 [31], where M is the set of classes and D the set of clusters. N denotes the number of data points used.

$$\text{purity} = \frac{1}{N} \sum_{m \in M} \max_{d \in D} |m \cap d| \tag{2.3}$$

We found that our clustering resulted in a perfect purity score of 1.0. This indicates an adequate embedding and cluster quality. To do an additional check, we analyzed whether not all phrases were assigned to one cluster as this inherently will result in a purity score of 1.0. This was not the case, however, we found that cluster 5 and 6 were the biggest clusters with size 63 and 49 respectively. This may have influenced the high purity score as many classes could fall in either cluster.

2.4.3 Expert Interviews

We conducted interviews with 5 experts. Each of them received either 72 or 73 phrases from all clusters found in [29]. In the subset creation, we ensured that the consistency amongst the responses of the interviewees could be assessed by including the same 10 randomly selected phrases to the sample set of each interviewee. Due to time constraints, we were not able to ensure that each phrase was evaluated twice as was initially planned in our methodology of this module. To verify the consistency of the K interviewees we used the lower bound on the error relative to the (unknown) ground truth using Equation 2.4 [32]. In Equation 2.4, N is the number of phrases in the overlapping subset, which is 10 in this work. Moreover, Y_n is the number of interviewees that labelled phrase n as a true positive value. When the error rate is lower than 0.10, we can assume that the results consistently propagate to the non-overlapping phases [32].

$$\bar{e} \geq \frac{1}{KN} \sum_{n=0}^{N-1} \min\{K - Y_n, Y_n\} \tag{2.4}$$

The results of this consistency experiment can be found in Table 2.3. As shown in Table 2.3, we see that for the evaluation of true positives the error rate is 0.08. The error rate for the assignment to the correct cluster is 0.24. Based on this table, we can conclude that the classification of true positives is reliable. However, the evaluation of clusters is less sound. In future work, an experiment setting where at least 2 experts evaluate each phrases is thus required. Whenever these two experts are in conflict, more analysis on the context and semantics could prove useful. One of the phrases where the experts were in conflict was "op basis van" (based on). Some experts denoted this phrase as an explaining phrase, while other stated that this was a referencing phrase. Both explanations are possible, depending on the context in which this phrase is used. This shows the importance of including contextual information in our analysis.

Error rate	**Value**
True positives	0.08
Correct cluster	0.24

Table 2.3: Error rates for agreement

The experts selected 204/322(0.634) phrases as true positives. A phrase was marked a true positive when it was either marked as suitable for legislative text or marked as a connecting phrase. Several true positives were close to our predefined thresholds in Section 2.4.1. This indicates that some potential false negatives exist and these were missed by our search method. False positives were mostly phrases that are commonly used, but not specific enough for our work. Examples of such phrases are "door" (by) and "bedoeld" (meant). In the case of "door", we found that this phrase indicates a resource. Resources are not considered in the scope of our work as we focus on process relation, such as causal and temporal relations. However, it could prove useful in future endeavours as it could be valuable information to be extracted from legislative texts. In the case of "bedoeld", we found that this N-gram is too short to be recognized as relevant by our experts. This conclusion was made since the phrase "als bedoeld" was considered as a true positive.

From the selected true positives, 104/204(0.510) were assigned automatically to the correct cluster. The clusters indicating examples and conditions were misclassified most often. This misclassification is probably due to the context-dependent nature of phrases in these clusters. Another reason for misclassification could be the fact that our language model was trained on a regular corpus rather than a corpus specific for the Dutch legal domain. Such phrases could have different meanings in regular language as opposed to legal language. Moreover, our experts indicated that a cluster was missing. 3/5 experts suggested to add a cluster for "referencing".

2.5 Conclusion

This module demonstrated the ability to semi-automatically mine signal phrases from legislative texts. We demonstrated that we could combine domain knowledge with automated processes, such as extraction, embedding and clustering. Furthermore, we established that we could successfully filter out false positives with a relatively small number of domain experts.

The presented module is applicable on any domain for any European language. This can be done by following the steps introduced in this module. These steps are summarized and listed below.

- **Extract potential phrases:** In this step potential phrases are extracted. To assist extraction, domain expert curated list of signal phrases are required to enhance the search.

- **Embedding and clustering:** The extracted phrases are embedded using a language model and then clustered. Before clustering, predefined phrase types are determined and centroids are defined.

- **Expert evaluation:** Expert evaluations aim to remove false positives and evaluate cluster quality. The minimum number of experts required depends on the number of extracted phrases.

The remainder of this section discusses some possible improvements that could be made in future work. In our extraction phase, we stopped adding more POS-tags to prevent too many false positives. However, adding more POS-tags also broadens our search. Thus in future work it could be worthwhile to investigate ways or metrics for which search is still broad, without including too many false positives. Tackling the risk of capturing false positives is important as having many false positives decreases the feasibility in the expert evaluation phase.

We saw in our results that the classification of clusters into categories requires more domain experts. This could provide consistent responses in terms of clustering. In future work, experiments could be designed with a greater focus on the context.

Embedding quality is crucial for the classification of the right cluster categories. To improve the embedding quality, an improved custom embedding would prove useful. The model that we used was trained on a general corpus, while the self-trained model used in this module did not find synonyms well enough. The self-trained model could have performed better if trained on a bigger legal text corpus. More legal text sources could have been added to the corpus, such as case law and other legislative documents. This could be an improvement in future work as this improves the context awareness of our language model. As discussed in the evaluation, context is of great importance for the categorization of signal phrases.

In the remainder of this work we attempt to extract patterns and local process models. To facilitate the extraction of these, we utilize the signal phrases mined in this module.

Chapter 3

Pattern Extraction and Local Process Model Creation

This module uses the signal phrases extracted in Chapter 2 to extract local process models from text. First, preliminary knowledge required for understanding this chapter will be discussed. Then we discuss related works. Thereafter, we present the steps taken in this module in the methodology section. After the methodology is explained, we discuss the evaluation of these steps and present the results. Finally, we draw conclusions, discuss implications and suggest future work.

3.1 Preliminaries

This section describes preliminary terminology and concepts required for understanding the remainder of this chapter.

- **Deontic rules:** Deontic logic is a type of logic that covers permission, obligation and related concepts [33]. Deontic rules are therefore obligatory or permissive rules. This type of rule is typically found in legislative texts.

- **Process template:** We define process templates as blueprints for patterns that indicate a process [34].

- **Implication:** An implication denotes a conditional relationship. In regular language this can be seen as If a, then b.

- **Boolean algebra:** Boolean algebra is a form of algebra in which variables can either be true or false [35]. True and false are also often denoted as 1 and 0 respectively.

3.2 Related Works

Previous works has been focusing either on extracting rules or extracting dependencies. The authors in [36] present a linguistically-oriented, rule-based approach to identify and extract deontic rules from textual data. This approach is in contrast to other methods that are based on machine-learning [37]. The results presented by the authors seem promising in terms of precision and recall. However, limitations, due to parsing issues, exist for more complex cases.

In the study presented in [38], the authors focused on dependency extraction and visualizing extracted dependencies rather than decision logic. The extracted dependencies were presented in the Decision Model and Notation (DMN) standard. This study presented a framework for extracting Decision Requirements Diagrams (DRD) to visualize textual data. The work succeeded in presenting a successful use case. Moreover, the authors presented a NLP pipeline that could be useful for related works.

Another work on rule extraction demonstrates the task of generating rules from legal text documents [39]. One of the steps taken in this work involves relation extraction where tuples are formed that describe relations within predicates. To assist in this task, OpenIE [40] was used. One of the main challenges posed in this work is the representation of the meaning of complex sentences.

The techniques used in [41] where the authors extracted annotations from texts seem relevant for this module as well. In this work, the authors extracted ATDP [42] elements from textual descriptions of business processes. One of the NLP tasks used in this method is the extraction of dependency trees using a parser. Since legislative texts can be seen as texts consisting of processes, the method proposed in this work could be utilized for our module. The authors in [43] also applied the technique of parsing to assist in extracting formal rules from legal texts. Their approach uses language-agnostic components. They showed the utility of their presented modules using text documents of the zoning map of the city of Vienna.

3.3 Methodology

This module consists of three steps. The first step is the pattern creation and extraction. Then after individual patterns are created, models are formed by connecting the patterns. Finally, the extracted models are evaluated with domain experts. These steps are illustrated in Figure 3.1

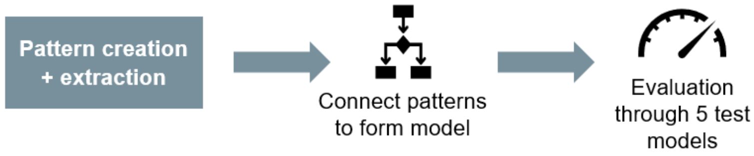

Figure 3.1: Steps taken in module 2

3.3.1 Pattern Creation and Extraction

Our first step in this module is the creation of patterns based on the relevant signal phrase types. This step aims to first create templates for process patterns and then to extract these patterns from legislative texts.

Cluster selection: To narrow the scope of this work, a selection of signal phrase types is made. We decided to select one signal phrase cluster based on analysis of cluster usage in the corpus of Dutch legislative texts and domain expert input. Cluster usage was analyzed by investigating which clusters were used most per sentence. To accomplish this, we split all law texts into sentences. Splitting the sentences is done using the sentence tokenizer function build into nltk, a NLP library for Python[1]. Then we checked per sentence which signal phrase and corresponding cluster type was used. The most occurring and most favorable cluster amongst domain experts were used for the remainder of this work. To illustrate the method above, we take an installation guide as an example document. Using our method, we then will probably see that most sentences contain a temporal relation, e.g., "first this, then that". In this example, when we analyze this qualitatively we could also conclude that temporal relations are the most important for installation guides.

Pattern creation for selected cluster: Patterns should be defined before they can be extracted. Once a suitable cluster is selected, we created patterns for this cluster, we call these patterns *process templates*. In case of more clusters, we take this step for each cluster. Process templates are created based on text analysis and linguistic literature. Textual analysis was done manually for each signal phrase: multiple sentences that use this phrase were selected. Based on the usage of signal phrases within these sentences, process templates were created for each signal phrase specifically. Additionally, literature research was done to find whether literature covered the usage of such signal phrases to create process templates. Typical queries that could be used during literature research are "Dutch [signal phrase type]

[1] `https://www.nltk.org/`

phrase" or "Dutch [signal phrase type] pattern".

Pattern extraction: After the creation of process templates, deeper understanding of the sentence structure is required to extract these templates. It is important to gain this knowledge to determine how to extract the templates. Knowledge could be acquired in multiple ways, such as pattern matching [44] or the usage of dependency parsers. In this work we decided to use dependency parsers as these are able to extract the syntactic structure of a sentence in the form of a parse tree [8]. As we handle Dutch texts, we used the parser introduced in [45], which is implemented in Spacy, a Python package for NLP tasks[2]. We selected this specific parser as it was both implementable in our Python module and its state-of-the-art performance in terms of accuracy. Using this parser we found the relevant sub-sentences that are related to components within the process template. Assumptions on the sentence structure were made after the signal phrase type was selected.

3.3.2 Pattern connection

After the extraction of single patterns, we connected the patterns to one another to form local process models. This was done by combining three methods. The methods are introduced and explained in this subsection.

Method 1 - Nested patterns: The first method to connect these patterns was by extracting nested patterns. We define nested patterns as patterns that occur within sub patterns. To illustrate this, we use the following example. Suppose we have an implication pattern where A implies B, i.e., $A \to B$. Then in case of a nested pattern, another implication exists in B. So we suppose that $B = (C \to D)$. Using this, the original $A \to B$, can be rewritten as $A \to (C \to D)$. This way of nesting could occur n times within an implication $A \to B$, where $n \in \mathbb{N}$. The extraction of such nested patterns was done using a recursive algorithm. In each recursion, the algorithm recursed on the result of a pattern. Whenever a pattern exists within the result, the algorithm continues its recursion. In case no patterns exist anymore, the algorithm stops. This recursive mechanism is illustrated in pseudo-code in Algorithm 1. It can be seen that this algorithm assumes that new patterns only exist in the result of a pattern. This assumption was made to reduce the complexity of this model given the limited time-span of this project.

Method 2 - Independent pattern connection: Besides extracting nested patterns to connect patterns in forming local process models, we con-

[2] `https://spacy.io/models/nl`

Algorithm 1 Mine_nested_patterns(resulting_pattern)

if a pattern exists in resulting_pattern.result **then**
    Call mine_nested_patterns(resulting_pattern.result)
end if
return resulting_pattern

nected independent patterns. This connection of patterns rests on the assumption that the order of independent assumptions is arbitrary. In this module we also tested this assumption both theoretically and practically.

Method 3 - Utilize text structure: Our third method uses the structure of legislative texts. These texts have the benefit that references are made in a structured manner. In this work, we utilized the structured method of referencing to connect multiple sentences to one another. Within law articles paragraphs refer to one another by referencing their paragraph number. In the example below, we underlined the explicit reference to paragraph 1. For the scope of this work, we only focused to references to paragraphs within the same law article, rather than references to other articles or laws.

> The contracting authority which supplies the information referred to in paragraph 1 shall request the tenderers or candidates in the contract award procedure to indicate that they have taken account, when drawing up their tender, of the obligations relating to employment protection provisions and the working conditions which are in force in the place where the works are to be carried out or the service is to be provided.[3]

In our approach we used assumptions to scope the referencing of paragraphs. These assumptions are based on analyses done on the text. We measured the occurrences of references within all sentences from our legislative corpus. Based on these analyses, we were able to draw assumptions to scope our work. An example of an assumption that can be drawn from this analysis is: "Based on the analysis we assume that sentences in legislative texts have X references". To connect the content of multiple paragraphs or sentences, we replace the paragraph reference with the relevant information from the paragraph to which the reference refers. We denoted relevant information as the full sentence or paragraph in case no pattern exists. In the situation where a pattern does exist within the referred text, we only take the result of the pattern. The result is taken as otherwise the full pattern

[3] `https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:C:2003:147E:0001:0136:EN:PDF`

will be extracted. This causes duplicated nested patterns which will result in an incorrect resulting local process model. To denote that a reference is made, the inserted information from the referred text starts and ends with **.

3.3.3 Expert Evaluation

Once the local process models are created, evaluation of these models is required to validate whether the models correctly represent the legislative text. For the evaluation, domain experts were interviewed in a semi-structured interview format.

For the semi-structured interviews, a selection of 5 arbitrary law articles was made. These law articles originate from multiple laws. From these law articles, the local process models were extracted using the previous steps of this module. For feasibility reasons we narrowed the scope of the user tests. A selection of 10 signal phrases and their corresponding patterns was made. The process models extracted from the sample articles were created based on these 10 signal phrases. The sample is therefore selected on the inclusion of at least one of these selected signal phrases.

An interview template was generated automatically based on each extracted local process model. For each element in this model, questions were generated on whether the element is correct considering the text. Moreover, for each model it is asked how this model should be rated. The automatic generation of interview templates could potentially be used to evaluate process models for a wider range of domains besides the legal domain. The generated template used in this work can be found in Appendix B.

3.4 Evaluation and Results

The evaluation methods and results of the steps presented in this module are covered in this section. For the first two steps results are presented. In the final step the quality is evaluated.

3.4.1 Pattern Creation and Extraction

The results of the pattern creation and extraction step of this module can be summarized as follows. Within this step three substeps were taken.

Cluster selection: In the first step we first selected the best signal phrase type for this work. Figure 3.2 shows the top 5 results of analysis on signal phrase types used in sentences of our legislative corpus. It can

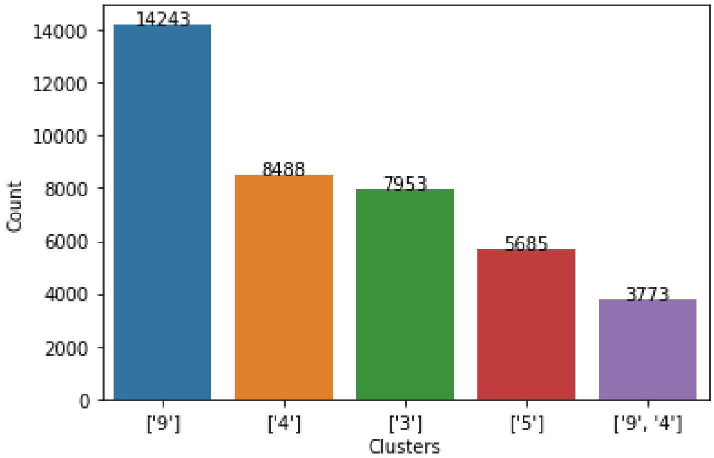

Figure 3.2: Top 5 most used cluster combinations and their respective counts. The x-axis outlines the cluster combinations and the y-axis shows the count.

be seen that cluster 9, which is the other cluster has the highest count. Since cluster 9 is least relevant we could discard this cluster. Then clusters 4 (explanatory), 3 (comparative) and 5 (conditional) make the top 3. Based on the results in Figure 3.2 and domain experts discussions, we selected cluster 5, the conditional cluster as most suitable cluster to utilize in this work.

Pattern creation: In the pattern creation step we first analyzed literature on Dutch conditional phrases to find how such phrases could be modelled as patterns. Example formulae for the phrase "mits" (provided that), can be found in Equations 3.1a, 3.1b [46]. In these equations, p represents the condition (PROTASIS, in traditional logical terminology). Moreover, q represents the expression which the condition is appended to (the APODOSIS) [46]. In this work, the apodosis is also referred to as the result of a condition.

$$q, \text{"mits"}\ p \tag{3.1a}$$

$$\text{"mits"}\ p, q \tag{3.1b}$$

Equations 3.1a, 3.1b can be simplified and generalized to the following logical formula $p \rightarrow q$, which indicates that condition p implies result q. The remainder of the conditional phrases were rewritten in formulae like these. For example, the phrase "tenzij" (unless) could be modelled as $\neg(p \iff q)$ or $\neg(p \rightarrow q)$ depending on the context.

Pattern extraction: The extraction of patterns was done using the dependency parser introduced in [45]. This parser was used on a sample of

```
          haal
   _______|__________________
  |   |       studeer      tentamen
  |   |    ______|_____        |
  ,   ik Als     ik   goed   mijn
```

Figure 3.3: Parse tree for example sentence "Als ik goed studeer, haal ik mijn tentamen" (If I study well, I will pass my exam)

sentences containing a conditional phrase. Based on the resulting parse trees for this sample, we found that the condition p can be found in the same sub-tree where the conditional phrase occurs. Therefore, the sub-tree where our signal phrase occurs could be marked as p. Moreover, we assumed that the remainder of a sentence s that is not p can be automatically classified as q [47]. Figure 3.3 shows the parsing result for the example sentence "Als ik goed studeer, haal ik mijn tentamen" (If I study well, I will pass my exam). In this example, "als" (if) is our conditional signal phrase. When we manually define p and q for this sentence, we would set "Als ik goed studeer" (If I study well) as the condition and "haal ik mijn tentamen" (I will pass my exam) as the result. When we look at the parse tree in Figure 3.3, we see that the subtree containing "als" (if), is equivalent to our manually defined p.

In our training sample of conditional sentences we noticed that long sentences were not parsed correctly. Two problems occurred during parsing. The first problem was that the condition was not captured within the correct sub-tree. As a result, the mined condition was not complete and therefore too short. The second problem was that the word-order of the condition was not maintained correctly in some situations. This caused the model to fail in finding the correct q as this is found by removing p from sentence s.

To resolve these two problems, we implemented additional rules to aid the condition extraction. To fix the parsing errors, a solution was created for each problem. The first solution is based on the ratio of the length of condition p and a full sentence s. Whenever this ratio is low, we can assume that the parser did not parse the full condition. This assumption was backed by analyzing sample sentences were the sub-sentence that was marked as condition was very short (~ 3 tokens). We set the threshold value for the condition-sentence ratio to 0.25. This ratio was selected as analysis of sample sentences where conditions were marked correctly showed that a ratio of 0.25 was the minimum ratio in our sample. Whenever the condition-sentence ratio was lower than 0.25 we used regular expressions to naively extract the condition. Our regular expression used a look ahead solution to greedily take the sub-sentence that ranges from our signal phrase until the last occurrence of "," or ".". To fix the second problem, we checked whether the extracted condition occurs directly in the sentence by checking whether the literal condition exists within the sentence. If this is not the case, we used a regular expression to lazily search until the first comma occurrence. Using these two solutions we were able to more accurately parse conditions from sentences in our sample set.

Recap substeps taken: These substeps above were taken to extract conditional patterns. To expand the extraction to other signal phrases, one should first create the pattern based on the signal phrase type. For pattern creation, literature research with example queries "[signal phrase type] linguistic model" could be used. Moreover, patterns can be found by analyzing how signal phrases are used in the corpus. Then to extract the patterns, dependency parsers are used. It may differ per signal phrase type how the subtrees of the dependency tree should be utilized.

3.4.2 Pattern connection

This subsection covers the pattern connection step. Within this step, three methods were utilized to connect patterns to form local process models.

Nested conditions: The first method of pattern connection is through the extraction of nested conditions. These nested conditions were defined as conditions that can be found within results as illustrated in Equation 3.2.

$$p_1 \rightarrow (p_2 \rightarrow ...(p_n \rightarrow q)) \tag{3.2}$$

Using Boolean algebra, we can prove that these nested conditions can be flattened to a formula where the result holds when all condition properties are satisfied. For the proof we use Equation 3.2, where $n = 2$. Below we proved how this equation results in Equation 3.7. Using this resulting equation, we modelled our local process models that contain nested conditions.

$$\begin{align} p_1 \rightarrow (p_2 \rightarrow q) &= \neg p_1 \vee (p_2 \rightarrow q) && \text{(Implication law)} \tag{3.3}\\ &= \neg p_1 \vee (\neg p_2 \vee q) && \text{(Implication law)} \tag{3.4}\\ &= (\neg p_1 \vee \neg p_2) \vee q && \text{(Associativity law)} \tag{3.5}\\ &= \neg (p_1 \wedge p_2) \vee q && \text{(De Morgan's law)} \tag{3.6}\\ &= p_1 \wedge p_2 \rightarrow q && \text{(Implication law)} \tag{3.7} \end{align}$$

Independent pattern connection: Another method to connect patterns is the assumption that the order of independent patterns is arbitrary. By this assumption, independent patterns were placed next to one another on the order that the patterns are extracted, i.e., the first extracted patterns is placed before the second extracted pattern. This assumption can be proved with the following example. Suppose we have two independent patterns X and Y. Then under this assumption, our model can be modelled as $\neg X \rightarrow Y$. Since the implication rule indicates that $\neg X \rightarrow Y \iff X \vee Y$. We can also model independent patterns X and Y as $\neg Y \rightarrow X$, then the following relation follows $\neg Y \rightarrow X \iff Y \vee X$. By the associativity rule, $X \vee Y \iff Y \vee X$. This demonstrates that our assumption holds theoretically.

Patterns are connected to form a model by creating nodes for conditions and results. After each condition, a choice node is created. This choice node has two outgoing edges, one for when the condition is satisfied and one for when it is not. The edge for the satisfied condition leads to the result node. In case of nested conditions, this edge leads to the next condition within the nested condition set. The edge when the condition is not satisfied leads to the following condition whenever an additional condition exists. In case no additional conditions exist, the edge leads to the result "niet van toepassing" (not applicable). This latter option is based on the assumption that whenever a condition is not satisfied within a sentence, the sentence is not applicable.

Utilize referencing style: The third method is the utilization of explicit referencing within legislative texts. For creating models, some assumptions were made on this referencing method. The first assumption is that within a sentence at most one reference is made. This assumption is supported by analysis where we found that the majority of sentences that contain references have one explicit reference, namely 89%.

3.4.3 Expert Evaluation

As a prerequisite to sample creation, a subset of size 10 of the conditional phrases mined in Chapter 2 was selected. The selection of conditional phrases was based on their occurrence in the legislative corpus. Moreover, the shortest n-grams were selected for implementation purposes. Properties of the selected phrases can be found in Table 3.1.

The selected phrases are "indien" (if), "wanneer" (when), "mits" (provided that), "tenzij" (unless), "ingeval" (in case), "krachtens" (by virtue of), "behoudens" (subject to), "voorzover" (to the extent that), "ingevolge" (as a result of), "op grond" (based upon). The sample used for the expert evaluation consisted of five models extracted from their corresponding law articles.

Signal phrase	Frequency	Coverage
indien	54223.0	0.863304
krachtens	17376.0	0.715643
op grond	16456.0	0.717836
ingevolge	8084.0	0.606725
tenzij	6836.0	0.559211
wanneer	4375.0	0.368421
voorzover	2848.0	0.336257
behoudens	2197.0	0.392544
mits	2146.0	0.341374
ingeval	2043.0	0.295322

Table 3.1: Selected sample phrases and their corpus properties sorted on frequency

The law articles for extraction are selected arbitrarily with the condition that it should only contain at least one of the 10 selected conditional phrases. The final selection of law articles is given in Table 3.2.

For the expert evaluation, four domain experts were interviewed. All interviewees are consultants at Deloitte's Tax & Legal department. The interviews were conducted in a semi-structured manner and took place in real-life. For the interviews, experts were presented a law article together with the local process model. Color-coding was used to annotate the law text and the corresponding local process model component. This aided experts in quickly matching the text with its corresponding component. Figure 3.4, 3.5 show examples of the views presented to the interviewees. It can be sen that the law text is illustrated, including its original source (Figure 3.4) and the generated local process model (Figure 3.5). It can also be seen that the color-coding is utilized to match the text and the model. The full sample of extracted models can be found in Appendix C.

For each component in a local process model, experts could indicate whether this component was corresponding with the legislative text where the model is based on. After surveying the components, we requested the experts to rate the full local process model on a scale from 1 to 10 in terms of correspondence. Table 3.3 shows the average scores for the sample of models and the standard deviation of these scores, which describes the variation [48]. This table shows that based on the domain expert interviews, the majority (0.8) of the models score sufficiently (> 5.5). Moreover, the table shows that the standard deviation in responses is low for the first two models and higher for the remaining three models. The high variation indicates that the

Model number	Law	Article
1	Wet op de vennootschapsbelasting (Enterprise income tax law)	12ag
2	Wet op de omzetbelasting (Turnover tax law)	9
3	Wet op de loonbelasting (Wage tax act)	5
4	Wet op de omzetbelasting (Turnover tax law)	35c
5	Algemene wet bestuursrecht (General administrative law act)	5:10

Table 3.2: Sample of selected law articles for expert evaluation sessions

Wet op de omzetbelasting – Artikel 35c

1 De ondernemer bewaart kopieën van de door hemzelf dan wel, in zijn naam en voor zijn rekening, door zijn afnemer of een derde uitgereikte facturen, en alle door hemzelf ontvangen facturen in zijn administratie.

2 Wanneer een ondernemer de door hem verzonden of ontvangen facturen elektronisch bewaart, waarbij een online toegang tot de gegevens wordt gewaarborgd, heeft de inspecteur met het oog op de toepassing van deze wet het recht de facturen ter controle in te zien, te downloaden en te gebruiken, indien de ondernemer is gevestigd in Nederland, dan wel de ondernemer de belasting in Nederland verschuldigd is.

3 Ingeval facturen elektronisch worden opgeslagen, worden de gegevens die de authenticiteit van de herkomst en de integriteit van de inhoud waarborgen, eveneens opgeslagen.

Figure 3.4: Example of text view presented to interviewees

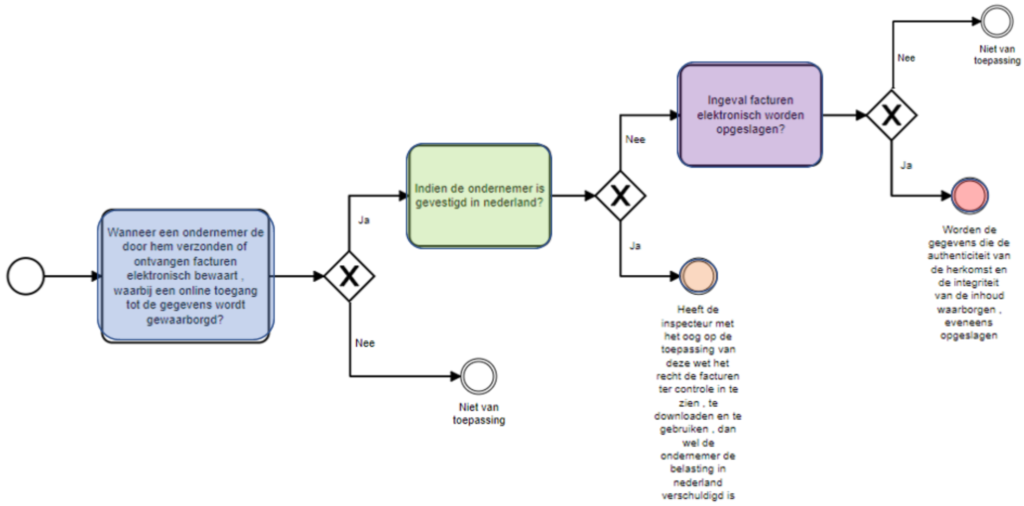

Figure 3.5: Example of model view presented to interviewees

resulting ratings may not be consistent enough to be reliable.

Model number	**Average rating**	**Standard deviation**
1	7.25	0.87
2	4.25	0.5
3	6.50	1.91
4	5.63	1.38
5	6.73	1.45

Table 3.3: Domain expert ratings on the scale from 1 to 10 and consistency for tested models

Besides looking at the raw scores, we aggregated the scores to create rankings of the models. By looking at the rankings we can verify whether domain expert responses vary in terms of the ranking. This also helps to counter bias, e.g., when an individual user rates all models systematically lower, which in return could cause high variation. Table 3.4 shows the rankings of the models. It can be seen that the agreement is highest on model 2 as this was rated lowest in all interviews. For models 1, 2 and 5 experts agreed in some terms. In model 3, no agreement was found as this model was ranked differently in each interview. These results indicate that in terms of ranking, some variation still exists as well.

During the semi-structured interviews, some discussion points arose as well. In the first model, interviewees indicated that the not applicable result could be more defined specifically to the legislative text to make it clearer

Model	Rank interview 1	Rank interview 2	Rank interview 3	Rank interview 4	Most occurring rank	Count of rank
1	1	3	2	1	1	2
2	5	5	5	5	5	4
3	1	2	5	3	-	-
4	2	5	3	2	2	2
5	2	1	1	4	1	2

Table 3.4: Ranking of models from 1 to 5 in four interviews

for the user. Moreover, domain experts indicated that references to other paragraphs should be also applied to the condition. For the second model, all interviewees agreed that this was not a correct representation. The main problem was that the incorrect condition was extracted. In this particular case, 'voor' (before) should be used as an if condition rather than 'mits' (provided that) . Here, 'voor' (before) was not flagged as a conditional phrase and even domain experts would not flag this as a conditional phrase in isolation. Therefore, in future work we recommend that whenever experts are not satisfied with a model, they should indicate how, i.e., based on which phrases, they would make a new model. For the third model, domain experts noted that the not applicable result could be more specified to the specific text. In this case the majority of the experts (3/4) indicated that instead of the not applicable result they preferred "dienstbetrekking van toepassing" (employment applicable). In the fourth model, the biggest remarks were on the incompleteness of one of the conditions. Here the experts noted that the parser failed to parse the full condition correctly. In our extracted model, only half of the condition was parsed, while the other half was not included. Another remark in this model was that the condition and result in paragraph 3 of the corresponding law article should be an independent model in itself. This indicates that one of our assumptions on model creation does not hold. For the last model, domain experts noted that the condition "tenzij bij wettelijk voorschrift anders is bepaald" (unless otherwise provided by legal regulation) is too generic since any law article could be provided by legal regulation. Consequently, the context of this condition is lacking in this situation. The context could have been created by analyzing the preceding articles. Hence, this could be possibly investigated in future work.

A general remark of one of the domain experts was that legislation contains words like "kan" (can/may). In legislation this word denotes may, and indicates that a result does not necessarily happen. This is an interesting finding and the usage of this type of modal verbs could be investigated in future work.

3.5 Conclusion

In this module we demonstrated that we can extract conditional patterns from Dutch legislative text and that we can connect them to form local process models. In our test sample of extracted models, 4/5 were rated sufficiently (> 5.5) by the interviewed domain experts. However, due to the high variation in the expert ratings, the results are not reliable yet. To improve reliability of results, more domain experts are required to validate the extracted models.

The presented module is applicable on more signal phrase types besides the conditional phrase type. This can be done by following the steps introduced in this module. These steps are summarized and listed below.

- **Pattern creation and extraction:** In this step patterns are created for the selected signal phrase type. The patterns are based on linguistic literature and usage of signal phrases in the corpus. After creation of the patterns, pattern extraction occurs, which is assisted by dependency parsers.

- **Pattern connection:** Patterns are connected using three methods to form local process models. These methods could differ per pattern type.

- **Expert evaluation:** Experts are enquired to evaluate the extracted local process models. Suggestions of these sessions could be used for further improvements.

A possible limitation of this work is the small test set of models. In future work, a greater test set originating from more domains using more signal phrases could be used. Another limitation of this work arose during the user tests. To create local process models we made assumptions on independent patterns. However, domain experts indicated that in our current approach, independence of patterns is not handled correctly. This is an interesting and unexpected finding. For future work more research is required in this specific assumption. Furthermore, the assumption that references only occur in results did also not hold. During evaluation, experts noted that conditions could also contain references and these are required to have a complete model.

In the creation of models we assumed that whenever a condition does not hold, we would simply reach the result of not applicable. The expert evaluation sessions indicated that this assumption holds, but that the description should be more specific for the concerned law article that is covered. Researching what happens in case a condition does not hold is another possible research direction for future work.

Besides the limitations in the assumptions made for this work one of the limitations in this work is the dependency on of-the-shelf methods such as dependency parsers. In the expert evaluation sessions we saw that parsing was not always correct, despite our attempts to assist the Dutch dependency parser used. As not much research has been done in Dutch parsing, it is possible that this problem would not arise when testing our methodology on other languages. Therefore, future work could focus on testing this methodology on a language, e.g., English, where the quality of parsers is better and where more parser methods exist to compare to one another.

Chapter 4

Visualization and Implementation

The goal of this module is to find the most suitable visualization method for the information extracted in Chapter 3. We first present preliminary knowledge and related works. Then we introduce the methodology of the four visualizations and the expert interviews. Following that, we introduce the evaluation and results. Consequently, we present the final implementation based on the results. Finally, we draw conclusions and discuss recommendations for future work.

4.1 Preliminaries

This section introduces the knowledge and terms required for understanding this module.

- **BPMN:** Business Process Management Notation (BPMN) is a standard notation with the goal to be understandable for all types of business users. BPMN defines a Business Process Diagram, which illustrates graphical models of business process operations in the form of a flowchart [49].

- **Activity:** An activity is represented by a rounded-corner rectangle. In this work, activities denote conditions that are either satisfied or not satisfied.

- **Event:** An event is represented by a circle and denotes the result of a condition in this work.

- **Sequence Flow:** A sequence flow is represented by an arrow. This denotes the order (the sequence) of activities performed in a process [49].

- **XOR Gate:** The Exclusive Or (XOR) gateway controls the sequence flow. In XOR gates, exactly one of multiple outgoing sequence flows must be selected [50].

- **Screen real estate:** Screen real estate indicates the available space for displaying output[1].

- **Deadlock:** A deadlock occurs when the process gets obstructed and cannot terminate in an event [51].

4.2 Related Works

After the extraction of local process models, it is important to visualize these in an intuitive manner. One of most common methods to visualizes processes is BPMN. This method has also been used in the legal domain as the authors of [52] utilized BPMN to visualize the Italian Family Reunification Law. In their work, the authors demonstrated how ontology and business process modelling tools can be combined to model legal texts. The authors acknowledged that the integration between tagging the text and automatically generating a model is lacking. In other works, BPMN was used to model legal knowledge for GDPR compliance checking [53, 54]. In [54], BPMN was selected as target model to integrate GDPR representation into business processes. These previous works have shown that BPMN is a plausible method to visualize legal texts and the processes within them.

In the past research has been done on the effect of color-coding in visualizations. In a review study, it was found that that color could be a very effective performance factor in search and identification tasks [55]. These findings are nowadays still relevant as these are utilized in modern design books [56].

Another method to visualize legal information is by means of a question-answering visualization method. This has been done in the past by means of legal question answering tools [57]. Additionally, online dispute resolution (ODR) tools, e.g., `https://magontslag.nl/` have been using the question-answer format [58]. These tools aim to assist users, through an user friendly environment, to resolve their disputes and go to court if necessary.

[1]`https://www.usabilityfirst.com/glossary/screen-real-estate/index.html`

Figure 4.1: Steps taken in module 3

To compare multiple visualization techniques, the authors in [59] used tasks that needed to be carried out by interviewees. The visualization technique with the lowest execution time was selected as most favorable visualization. Moreover, authors in [60] build on the previous work by introducing heuristics to capture the value of a visualization. This is a metric that goes beyond task accuracy or execution speed.

4.3 Methodology

In this section, we present the approach of the three steps of this module. The first step is a selection of four visualization methods. Then the second step is the evaluation of these methods. In the final step, the most suitable visualization is selected and implemented. Figure 4.1 shows the described steps.

4.3.1 Four visualization methods

For this module we created four visualization methods to display the information mined in Chapter 3. The first visualization is the created BPMN model as mined in previous chapter. The second and third visualizations are variations on this mined BPMN model with additional functionality to improve easy-of-use. The fourth visualization completely departs from the "model view" as introduced in the other visualization methods. The four visualization methods are illustrated in Figures 4.2 - 4.6

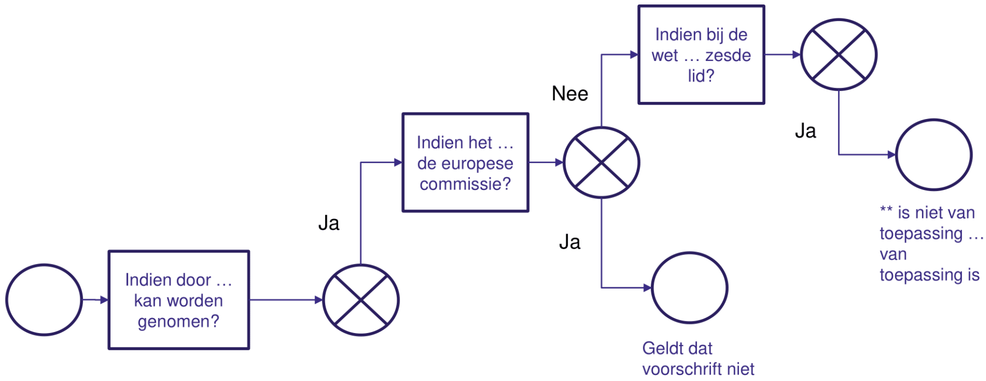

Figure 4.2: Visualization method 1: Mock-up illustrating the visualization method in BPMN. Note that full text is truncated to improve readability on paper.

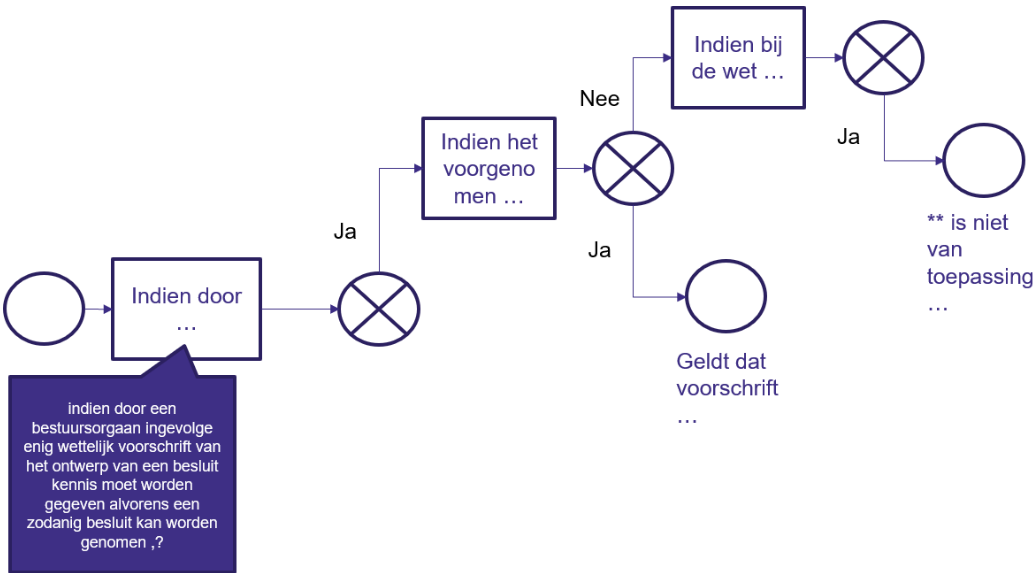

Figure 4.3: Visualization method 2: Mock-up illustrating the visualization method where users can hover over the activity and event labels to reveal the full text.

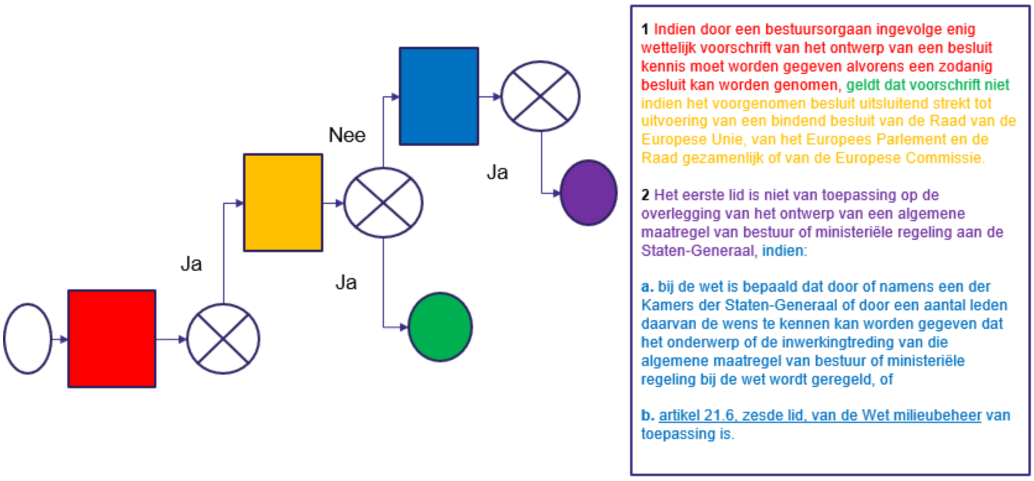

Figure 4.4: Visualization method 3: Mock-up illustrating the visualization method where color-codings were used to match the text and the process model

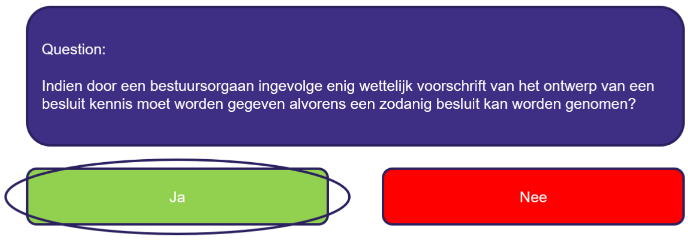

Figure 4.5: Visualization method 4: Mock-up illustrating a "ja" (yes) response to a question in the question-answer visualization method

Figure 4.6: figure
Visualization method 4: Mock-up illustrating a resulting answer to a question in the question-answer visualization method

Visualization method 1: Our first visualization method is the raw BPMN model mined from the text. Figure 4.2 shows an example of such BPMN model. In this model some adaptions are made to increase readability. Instead of the raw text, each label starts with a capital letter. Moreover, activity boxes are automatically adjusted based on the amount of text in the real visualization method, in the illustration presented in Figure 4.2 this adjustment was not shown for visibility reasons. Ideally, the event label size are also adjusted.

Visualization method 2: The second visualization method is a variation on the first method. The difference in this visualization is the fact that activities and event labels are not fully disclosed to reduce the size of the visualization. Reducing the size of the visualization is useful when the displayed BPMN models become extremely large in case of long articles or when full laws will be visualized in the future. To reveal the full textual descriptions, the user can hover over the activity or event description. This method ensures that all necessary information is captured, without taking too much screen real estate. Figure 4.3 shows a mock-up of this visualization method. This mock-up illustrates the situation in which the user is hovering its mouse over the left-most activity. As can be seen, a pop-up appears with the full text description for this activity.

Visualization method 3: The third visualization method also aims to maximize the information provided in the given screen real estate. Instead of using hovering to reveal more information, this visualization method presents both the text and the model structure of a legislative text at once. The full original text is provided together with the model. On the left side, the model is presented with unlabelled activities (rectangles) and events (circles). The original text is provided at the left side of the visualization. In order to connect the relevant parts of the text with the corresponding element in the model, we used color-codings to create this match. In the color coding procedure, each activity and event is colored with a distinct color. In the original text, the corresponding text that matches a activity or event is also colored with the color of its corresponding process element. This enables users to read the original text as usual, while directly seeing how the text should be represented in a model scheme. Thus, the relation between parts of texts can be seen in this way. A mock-up of this visualization method is shown in Figure 4.4. This figure shows that each part of the text is matched with a process element in the model.

Visualization method 4: The last visualization method differs substantially from the three previous visualization methods. Instead of presenting the user a model view and hence insight into the control flow of the text, this method is on a question-answer basis. In essence, the question-answer method follows the model mined in Chapter 3. The user is presented with questions, directly corresponding with the activities in the process. At each question, the user is able to either select yes or no. Based on the user's answer, this method shows either the next question (next activity) or the resulting answer (event nodes). The benefit of using this method is that the user is less easily overwhelmed by a plethora of information as only one question is shown at a time. Figure 4.5 shows a mock-up for this visualization method. This mock-up illustrates the first question and it can be seen that in this case, the user selected the answer "ja" (yes). In this mock-up it becomes clear that the user is posed a question as this is both indicated in the text box and by the "ja" (yes) and "nee" (no) buttons. The "ja" (yes) button is made green as green is the conventional color for a positive response [61]. The "nee" (no) button is made red as this is typical for a negative response [61].

Once the user reaches a result, the result screen is presented to the user. Figure 4.6 illustrates a mock-up where a result screen is shown. This screen is clearly a result as no option buttons are present. Furthermore, the text box header also denotes that the user reached a result.

4.3.2 Evaluation method

For the evaluation phase, we enquired domain experts to evaluate the four visualization methods presented above. Domain experts were interviewed in a semi-structured interview format. The interviewees were presented each visualization one at a time and they did not have a time limit to inspect the visualization. Each expert was asked to rate each visualization on a scale from 1 to 10. Then we gave the interviewees the opportunity to elaborate on the assigned score by asking why they gave a certain score. Moreover, we gave the interviewees the opportunity to suggest improvements to the visualizations. Besides the questions per visualization, we asked how to visualize situations where only one result holds, e.g., what happens if the first activity does not hold. As concluding question, we asked after the interviewee has seen all visualizations, which visualization was their preferred method. The interview template used can be found in Appendix D.

After the results of the evaluation interviews were processed, we selected the visualization that has both the highest average score and was voted most often as favourite visualization. In case of a tie, we would look into the qualitative responses of the domain experts to understand what distinguishes the tied visualizations. Then based on these responses, pro's and con's will be created and compared to select the most suitable visualization method.

4.4 Evaluation Results

This section covers the evaluation results of the four visualization methods. Five experts were interviewed for the evaluation of our visualizations. All interviewees are consultants at Deloitte's Tax & Legal Department. The interviews took place in real-life, where we demonstrated the visualizations on a screen, which the expert could interact with.

Each visualization was ranked from 1 to 10 by the interviewed domain experts. We took the average score for each score to determine which visualization scores highest. Table 4.1 shows the resulting average scores. It can be seen that visualization 1 scores highest with an average score of 7.1. It is also clear that visualization 2 is the lowest rated visualization with an barely sufficient (> 5.5) average score of 5.55.

Visualization number	**Average score**	**Standard deviation**
1	7.10	0.89
2	5.55	1.07
3	6.80	0.57
4	6.75	0.90

Table 4.1: Average scores of visualization evaluation

The experts indicated that visualization 1 enabled them to see the overall structure of the text well, while it also maintains the original legislative text in the activity and event labels. One expert indicated that a legend could be helpful as they did not immediately understand the syntax of the XOR-gates. The experts noted that visualization 2 provides less information as the full text is not shown directly. Moreover, they found the revelation of text while hovering less useful as this does not enable them to compare the full texts of activities. The experts noted, however, that hovering allows for bigger process models within the given screen real-estate, which is useful for longer texts. For visualization 3, the experts noted that the color-coding helps them to link the text directly to the graph. However, they stated that the usage of colors could make the text less understandable. Especially with larger texts, having many colors could confuse the user. One expert noted that they felt that the model was incomplete, since they found that the text was missing in the labels of the model. Visualization 4 was perceived as the most easy-to-understand visualization. All experts noted that this visualization is most useful for non-experts in the legal domain. However, they noted that a lack of providing an overview of the flow of a legislative text is a great pitfall of this method. Therefore, one expert recommended to combine the question-answer view with an overview of the process model to indicate the user's progress in the control flow.

For situations where only one answer is available, all experts suggested to add an additional sequence flow (arrow) that leads to "niet van toepassing" (not applicable). This ensures that the user will always receive a result and that a dead-lock will not occur. In determining the best visualization method, after the majority vote, it was found that 4/5(0.8) of the experts rated visualization 1 as the most useful visualization. This majority vote result is also in line with the average scores in which visualization 1 was rated highest. The qualitative responses also indicate that visualization 1 was the most favorable method as this method did not suffer from issues that arose in other visualization methods.

4.5 Implementation at Deloitte

In the previous section we selected visualization 1 as the most suitable visualization technique. Moreover, all experts indicated that for activities with only one event, the "niet van toepassing" (not applicable) event should be added. This section covers the implementation of the visualization method for the mined local process models from legislative text. As a proof-of-concept for this research, we built an implementation within Deloitte's Moonlit platform[2].

Moonlit is a legal research tool that aims to automate and improve efficiency in tasks that legal professionals typically conduct. This platform contains document analysis features ranging from recommending relevant documents and summarizing legal text documents to predicting outcomes and durations of cases based on facts provided by the user. The implementation of the modules presented in this work is used as a beta feature within this platform.

The implementation of the proof-of-concept is done using Python 3.7.5 to extract the local process models. Visualization is done using bpmn-js, a JavaScript library for visualizing BPMN models[3] from an XML input. The XML input was generated automatically using Python 3.7.5. Figures 4.7, 4.8 show screenshots of how such proof-of-concept would look like within the Moonlit platform. Figure 4.7 shows the text view when a user selected a legislative document. Within this text view, the user can select the visualization button to go to model view presented in Figure 4.8.

[2] `https://www.moonlit.ai/`

[3] `https://bpmn.io/toolkit/bpmn-js/`

Figure 4.7: Displaying a legislative text in the Moonlit platform

Figure 4.8: Displaying an extracted BPMN model in the Moonlit platform

4.6 Conclusion

In this module we researched the most suitable visualization method and implemented this within Deloitte's Moonlit platform. We proposed four visualization methods from which one was selected for further implementation. Domain expert interviews demonstrated that both displaying the original text and the control flow of the text result in the most valuable representation for legal practitioners. However, whenever the target user is a non-expert in the legal domain, the question-answering visualization was found to be the most recommended representation. Hence, we can conclude that the most useful visualization is highly dependent on the target user. Furthermore, the experts noted that no dead-ends should exist in the model. Therefore, we added additional events representing "niet van toepassing" (not applicable) to activities with only one result.

A possible future work could focus on improving how information is represented in this module. One aspect that could be further discovered in future work is how text is represented. At the moment the original text is used for the creation of questions. This representation could present risks in terms of understandability. Hence, the formulation of questions could be improved in future work by looking at how text could be rewritten to a question automatically. This could be a challenging task, since this will be highly language-dependent.

Another future work could focus on the method of referencing other paragraphs or other laws. Referencing could be improved by indicating hyperlinks. Further research is required to find whether such method would be useful and relevant for legal practitioners.

Combining visualization methods could be another interesting future research topic. Visualization 1 and 4 were both highly regarded by the domain experts, due to their ability to provide an overview and simplicity respectively. Combining these methods and testing whether this combination will be perceived worthwhile is another option that may be pursued in the future.

Chapter 5

Conclusion

In this work we introduced a framework that aims to extract local process models from legislative text and to visualize the underlying process models in a meaningful way. To verify whether our proposed framework succeeds, we try to answer the research questions **RS1** and **RS2** introduced in Chapter 1. For reference, **RS1** and **RS2** are stated below.

> **RS1:** "How to mine local process models from legislative texts?"
> **RS2:** "In what way can we provide the user with knowledge about the local processes in legislative texts?"

Section 5.1 answers the stated research questions and covers the main conclusions of this work. Then the implications that arose during this work and recommendations for future work are presented in Section 5.2.

5.1 Main conclusions

The framework proposed in this work consists of three modules. Module 1 and 2 were introduced to answer **RS1** and Module 3 was introduced to answer **RS2**.

5.1.1 Answering RS1

To answer the question how we can mine local process models from legislative text, we used two modules to answer this research question. Each module solves a separate subproblem required to answer the general research question.

Module 1: In module 1, we extracted signal phrases that are typical for the legal domain. In this module we introduced an approach where these signal phrases can be extracted semi-automatically using the support of five domain experts. Since each domain, e.g., the legal- or medical domain, have their typical jargon, finding phrases specifically for these domains is not trivial. The main academic contribution of this module is a method to find aspecific and frequent signal phrases for the legal domain. The proposed method does not rely on any specific properties related to the Dutch language or the legal domain and could therefore be applied to other domains and other languages.

Module 2: Using the extracted signal phrases from module 1, we extracted local process models using these phrases in module 2. In this module, we focused on the extraction of conditional patterns to form local process models. We have demonstrated that it is feasible to extract conditional patterns from legislative texts with the help of dependency parsers. After the extraction of conditional patterns, we connected these patterns to form local process models. The connection of these patterns was done using assumptions that we have proven as well. In the evaluation of our test sample, 4/5 test models were approved by domain experts. The main academic contribution of this module is that we could extract local process models from texts that consist of more than one sentence. This contribution is novel as current state-of-art techniques focused on the extraction of process models on the sentence level.

Research question: The proposed two modules in this work have demonstrated that it is feasible to extract local process models using knowledge on signal phrases, the help of dependency parsers and additional assumptions. Moreover, the extraction of signal phrases in module 1 could be a gateway to other tasks such as text annotation or local process extraction from text as is the goal of our work. Therefore, the research question **RS1** could be answered successfully.

5.1.2 Answering RS2

To answer the question in what way we can provide the user with knowledge about the local processes in legislative texts, we proposed one module to investigate this problem and answer the research question.

Module 3: Module 3 aims to visualize the process information extracted in module 2 in a meaningful and user-friendly way. In this module, four visualization methods were created and evaluated by domain experts. Domain expert evaluation showed the visualization method where both the original text and the control flow of the text were presented was the most favorable solution for legal practitioners. However, for users that are not active in the legal sector, a visualization method where the control flow is hidden was more favorable. Consequently, the visualization method is highly dependent on the target users. The main academic contribution of this module is a method to test multiple visualization techniques to determine the most suitable technique for visualizing process models.

Research question: The module introduced in this work have demonstrated that the best way of providing users with knowledge about local processes in legislative texts is dependent on the type of user. For the target audience of this work and the software platform in which our proof-of-concept is implemented, a visualization that both shows content and control flow was determined as the best approach.

5.2 Limitations and future work

This section covers the limitations of the steps taken in this work. Moreover, recommendations on future work are made. The limitations and recommendations for future work are presented per module. Finally, we present general recommendations for future work.

5.2.1 Module 1

One of the limitations that arose in module 1 concerned the search strategy for candidate signal phrases. In our approach, we set thresholds variants on constraints consisting of combinations of frequency, coverage and POS-tags. A limitation of this approach is that we could not search broader once we set our thresholds without adding too many undesired false positives. Therefore, we suggest for future work that more metrics could be used to make the search process more detailed. This could potentially lead to having less false positives and false negatives.

Another limitation of this module was the categorization of signal phrases. Out of the true positive signal phrases, just more than half (0.510) were categorized correctly using our automated process. After analysis of the misclassifications it was found that the clusters that are context-dependent were misclassified most often. Therefore, we could conclude that our current approach is not context-aware enough. Our current approach takes some context into account as embeddings inherently take the context of the training corpus into account. However, the Multi-lingual Universal Sentence Encoder model is trained on a normal language corpus rather than legal language. Hence, the context awareness could be improved by using a model trained on a legal corpus. Furthermore, to tackle the problem of context-awareness, more analysis on context is required in future work. Information on context could be acquired by methods that could analyze a signal phrase not only in isolation, but in its context. This could for example be done by providing phrases or their features, such as POS-tags or coverage, that are typically found around signal phrases.

5.2.2 Module 2

One of the limitations of module 2 is in the results of the domain expert evaluation sessions. In these evaluation sessions, 4/5 extracted models were rated sufficiently. However, the standard deviation, hence the variation, of these ratings was relatively high for 3/5 models from this test set. Furthermore, the results of the ranking also showed that variation still exists. This demonstrates that current results in terms of scores and ranks could be unreliable. A possible solution to tackle this problem is to enquire more experts. This enables us to use the wisdom of the crowd. By this principle the "true" rating unfolds itself when enough experts cast their opinions on the extracted models. In terms of insights to improve the model, enquiring more experts could help when legal practitioners that are enquired originate from a broader range of expertise.

At the moment we limited our scope to conditional patterns. During the evaluation sessions we learned that some articles go beyond this scope. Therefore, we recommend for future work to include additional signal phrase types and their corresponding patterns to capture more information from the legislative texts. Our proposed approach also depends on of-the-shelf tools such as dependency parsers. This is a limitation as our results rely on the quality of such tools. This limitation is illustrated in our expert evaluation sessions. In these sessions we saw that parse errors still occur for Dutch legislation. Since Dutch is a relatively small language, research in parsing this language is not as extensive as, e.g., for a much more common language such as English. A possible future work would be to apply our framework to other languages where more research is done in the parsing method.

Our work also has limitations on the assumptions used. In creating the local process models we assumed and proved that the order of independent conditionals does not matter for the end-result. However, during expert evaluations we learned that for the perception of users, the order of patterns does matter. This is a surprise finding and therefore we recommend to conduct further research in collaboration with domain experts on the perception of models. Another possible future research direction could be into the transformation of complex models.

5.2.3 Module 3

During this module we focused on the representation of information to prospective users. One of the limitations of our current visualization methods is the way conditions are presented. Both in the BPMN and the Question-Answering visualization methods, we present conditions as questions. However, at the moment these questions are "forced" by taking the original text and placing a question mark behind it. This method of presenting could be improved to improve readability for users. Future research is required in rewriting a sentence to a question. Possibly, language models like GPT-3 [62], which can generate language, could be utilized to accomplish this. Another limitation for visualization is the way that paragraphs references are displayed. More research is required to investigate how these references could be presented in a meaningful and intuitive way.

In this work, one visualization method was selected to be implemented as a proof-of-concept. This visualization method is most suitable for legal practitioners, but not for non-experts. Therefore, in future work, it could be valuable to research how the benefits of the selected visualization can be combined with an easier view to accommodate non-expert users.

5.2.4 General recommendations

This work has provided a solid foundation for the extraction of local process models from legislative texts. Through a case study on Dutch legislation, we demonstrated the feasibility of extracting these models. In the sections above we stated limitations that arose during the development of our framework. In future work we recommended to tackle these limitations first, before proceeding to further developments.

Possible future directions for our framework could lie in the extension of extracted process models from one article to multiple articles. In our domain expert interviews we learned that for legal practitioners the most added value lies in being able to connect several law articles. To extract process models that cover multiple articles creates additional challenges in referencing. Moreover, additional assumptions are required to connect these articles.

Another extension on our framework could be the enhancement of current legislation with knowledge from case law. It would be interesting to investigate whether conditions and other patterns could be mined from case law. These conditions could then be compared to current process models in legislation. Whenever the interpretation and application of current legislation in case law changes, these changes could be used to enhance the process models that are made of current legislation.

Besides providing visualization of legislative documents, our framework could also be used for compliance checking. Once the extracted process models are sufficiently representing the text, these models could be used on case law facts, contracts or other legal documents to verify whether the facts are compliant to legislation. This could increase efficiency of legal practitioners significantly as they could focus on more complex problems non-automated problems.

Bibliography

[1] Cambria E, White B. Jumping NLP curves: A Review of Natural Language Processing Research. IEEE Computational intelligence magazine. 2014;9(2):48-57.

[2] Otter DW, Medina JR, Kalita JK. A Survey of the Usages of Deep Learning for Natural Language Processing. IEEE Transactions on Neural Networks and Learning Systems. 2020;32(2):604-24.

[3] Chowdhury GG. Natural Language Processing. Annual review of information science and technology. 2003;37(1):51-89.

[4] Sorin V, Barash Y, Konen E, Klang E. Deep Learning for Natural Language Processing in Radiology—fundamentals and a Systematic Review. Journal of the American College of Radiology. 2020;17(5):639-48.

[5] Mishev K, Gjorgjevikj A, Vodenska I, Chitkushev LT, Trajanov D. Evaluation of Sentiment Analysis in Finance: from Lexicons to Transformers. IEEE Access. 2020;8:131662-82.

[6] Dale R. Law and Word Order: NLP in Legal Tech. Natural Language Engineering. 2019;25(1):211-7.

[7] Medvedeva M, Vols M, Wieling M. Using Machine Learning to predict decisions of the European Court of Human Rights. Artificial Intelligence and Law. 2020;28(2):237-66.

[8] Nivre J, Scholz M. Deterministic Dependency Parsing of English Text. In: COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics; 2004. p. 64-70.

[9] De Marneffe MC, Dozat T, Silveira N, Haverinen K, Ginter F, Nivre J, et al. Universal Stanford dependencies: A cross-linguistic typology. In: LREC. vol. 14; 2014. p. 4585-92.

[10] van der Aa H, Di Ciccio C, Leopold H, Reijers HA. Extracting Declarative Process Models from Natural Language. In: International Conference on Advanced Information Systems Engineering. Springer; 2019. p. 365-82.

[11] Shafiee F. Local Process Model Extraction from Textual Data. MSc Thesis. 2019.

[12] Friedrich F, Mendling J, Puhlmann F. Process Model Generation from Natural Language Text. In: International Conference on Advanced Information Systems Engineering. Springer; 2011. p. 482-96.

[13] Riefer M, Ternis SF, Thaler T. Mining Process Models from Natural Language Text: A state-of-the-art analysis. Multikonferenz Wirtschaftsinformatik (MKWI-16), March. 2016:9-11.

[14] Manna Z, Pnueli A. The Temporal Logic of Reactive and Concurrent Systems: Specification. Springer Science & Business Media; 2012.

[15] Olender KM, Osterweil LJ. Cecil: A Sequencing Constraint Language for Automatic Static Analysis Generation. IEEE Transactions on Software Engineering. 1990;16(3):268-80.

[16] Clarke EM, Emerson EA, Sistla AP. Automatic Verification of Finite-state Concurrent Systems using Temporal Logic Specifications. ACM Transactions on Programming Languages and Systems (TOPLAS). 1986;8(2):244-63.

[17] Cavnar WB, Trenkle JM, et al. N-gram-based Text Categorization. In: Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval. vol. 161175. Citeseer; 1994. .

[18] Brown RW. Linguistic Determinism and the Part of Speech. The Journal of Abnormal and Social Psychology. 1957;55(1):1.

[19] Webster JJ, Kit C. Tokenization as the Initial Phase in NLP. In: COLING 1992 Volume 4: The 14th International Conference on Computational Linguistics; 1992. .

[20] Mikolov T, Chen K, Corrado G, Dean J. Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:13013781. 2013.

[21] Koeva S, Obreshkov N, Yalamov M. Natural Language Processing Pipeline to Annotate Bulgarian Legislative Documents. In: Proceedings of The 12th Language Resources and Evaluation Conference; 2020. p. 6988-94.

[22] Waltl B, Landthaler J, Scepankova E, Matthes F, Geiger T, Stocker C, et al. Automated Extraction of Semantic Information from German Legal Documents. In: IRIS: Internationales Rechtsinformatik Symposium; 2017. .

[23] Doncel VR, Ponsoda EM. LYNX: Towards a Legal Knowledge Graph for Multilingual Europe. Law in Context A Socio-legal Journal. 2020 Dec;37(1):175-8.

[24] Hoekstra R, Breuker J, Di Bello M, Boer A, et al. The LKIF Core Ontology of Basic Legal Concepts. LOAIT. 2007;321:43-63.

[25] Sartor G, Casanovas P, Biasiotti M, Fernández-Barrera M. Approaches to Legal Ontologies: Theories, Domains, Methodologies. Law. Governance and Technology series Springer. 2011.

[26] Razali NM, Wah YB, et al. Power Comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling Tests. Journal of statistical modeling and analytics. 2011;2(1):21-33.

[27] Yang Y, Cer D, Ahmad A, Guo M, Law J, Constant N, et al. Multilingual Universal Sentence Encoder for Semantic Retrieval. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations; 2020. p. 87-94.

[28] Tulkens S, Emmery C, Daelemans W. Evaluating Unsupervised Dutch Word Embeddings as a Linguistic Resource. In: Chair) NCC, Choukri K, Declerck T, Grobelnik M, Maegaard B, Mariani J, et al., editors. Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016). Paris, France: European Language Resources Association (ELRA); 2016. .

[29] Bovenhoff M, Zeijl W. Basisboek Taal. Pearson Education; 2009.

[30] McKoon G, Ratcliff R. Priming in Item Recognition: The Organization of Propositions in Memory for Text. Journal of verbal learning and verbal behavior. 1980;19(4):369-86.

[31] Sanderson M. Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press. 2008. ISBN-13 978-0-521-86571-5, xxi+ 482 pages. Natural Language Engineering. 2010;16(1):100-3.

[32] Smyth P. Bounds on the Mean Classification Error Rate of Multiple Experts. Pattern Recognition Letters. 1996;17(12):1253-7.

[33] McNamara P, Van De Putte F. Deontic Logic. In: Zalta EN, editor. The Stanford Encyclopedia of Philosophy. Spring 2021 ed. Metaphysics Research Lab, Stanford University; 2021. .

[34] Awad A, Goré R, Thomson J, Weidlich M. An Iterative Approach for Business Process Template Synthesis from Compliance Rules. In: International Conference on Advanced Information Systems Engineering. Springer; 2011. p. 406-21.

[35] Boole G. The Mathematical Analysis of Logic. Philosophical Library; 1847.

[36] Wyner AZ, Peters W. On Rule Extraction from Regulations. In: JURIX. vol. 11; 2011. p. 113-22.

[37] de Maat E, Winkels R. Automated Classification of Norms in Sources of Law. In: Semantic processing of legal texts. Springer; 2010. p. 170-91.

[38] Etikala V, Van Veldhoven Z, Vanthienen J. Text2Dec: Extracting Decision Dependencies from Natural Language Text for Automated DMN Decision Modelling. In: Del Río Ortega A, Leopold H, Santoro FM, editors. Business Process Management Workshops. Cham: Springer International Publishing; 2020. p. 367-79.

[39] Ferraro G, Lam HP, Tosatto SC, Olivieri F, Islam MB, van Beest N, et al. Automatic Extraction of Legal Norms: Evaluation of Natural Language Processing Tools. In: Sakamoto M, Okazaki N, Mineshima K, Satoh K, editors. New Frontiers in Artificial Intelligence. Cham: Springer International Publishing; 2020. p. 64-81.

[40] Angeli G, Premkumar MJJ, Manning CD. Leveraging Linguistic Structure for Open Domain Information Extraction. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers); 2015. p. 344-54.

[41] Quishpi L, Carmona J, Padró L. Extracting Annotations from Textual Descriptions of Processes. In: International Conference on Business Process Management. Springer; 2020. p. 184-201.

[42] Sànchez-Ferreres J, Burattin A, Carmona J, Montali M, Padró L. Formal Reasoning on Natural Language Descriptions of Processes. In: International Conference on Business Process Management. Springer; 2019. p. 86-101.

[43] Recski G, Lellmann B, Kovacs A, Hanbury A. Explainable Rule Extraction via Semantic Graphs. 2021.

[44] Jacobs PS, Krupka G, Rau L. Lexico-semantic Pattern Matching as a Companion to Parsing in Text Understanding. In: Speech and Natural Language: Proceedings of a Workshop Held at Pacific Grove, California, February 19-22, 1991; 1991. .

[45] Bouma G, van Noord G. Increasing Return on Annotation Investment: The Automatic Construction of a Universal Dependency Treebank for Dutch. In: Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017). Gothenburg, Sweden: Association for Computational Linguistics; 2017. p. 19-26. Available from: `https://aclanthology.org/W17-0403`.

[46] Daalder S. Conditional Constructions: The Special Case of Modern Dutch mits. Journal of Germanic Linguistics. 2009;21(2):231-48.

[47] Funk WP. On a Semantic Typology of Conditional Sentences. 1985.

[48] Bland JM, Altman DG. Measurement error. BMJ: British medical journal. 1996;312(7047):1654.

[49] White SA. Introduction to BPMN. Ibm Cooperation. 2004;2(0):0.

[50] Allweyer T. BPMN 2.0: Introduction to the Standard for Business Process Modeling. BoD–Books on Demand; 2016.

[51] Van Dongen BF, Mendling J, van der Aalst WM. Structural Patterns for Soundness of Business Process Models. In: 2006 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC'06). IEEE; 2006. p. 116-28.

[52] Ciaghi A, Weldemariam K, Villafiorita A. Law Modeling with Ontological Support and BPMN: A case study. In: Proceedings of the Fifth

International Conference on Digital Society–CYBERLAWS; 2011. p. 29-34.

[53] Palmirani M, Governatori G. Modelling Legal Knowledge for GDPR Compliance Checking. In: JURIX; 2018. p. 101-10.

[54] Bartolini C, Calabró A, Marchetti E. GDPR and Business Processes: An Effective Solution. In: Proceedings of the 2nd International Conference on Applications of Intelligent Systems; 2019. p. 1-5.

[55] Christ RE. Review and Analysis of Color Coding Research for Visual Displays. Human factors. 1975;17(6):542-70.

[56] Proctor RW, et al. Human Factors in Simple and Complex Systems. CRC press; 2018.

[57] Bennett Z, Russell-Rose T, Farmer K. A Scalable Approach to Legal Question Answering. In: Proceedings of the 16th edition of the International Conference on Articial Intelligence and Law; 2017. p. 269-70.

[58] Reiling D. Beyond Court Digitalization With Online Dispute Resolution. In: IJCA. vol. 8. HeinOnline; 2016. p. 2.

[59] Kobsa A. An Empirical Comparison of Three Commercial Information Visualization Systems. In: IEEE Symposium on Information Visualization, 2001. INFOVIS 2001. IEEE; 2001. p. 123-30.

[60] Wall E, Agnihotri M, Matzen L, Divis K, Haass M, Endert A, et al. A heuristic approach to value-driven evaluation of visualizations. IEEE transactions on visualization and computer graphics. 2018;25(1):491-500.

[61] Levkowitz H. Color Theory and Modeling for Computer Graphics, Visualization, and Multimedia Applications. vol. 402. Springer Science & Business Media; 1997.

[62] Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, et al. Language Models are Few-Shot Learners. arXiv e-prints. 2020:arXiv-2005.

Appendix A

Module 1: Interview template

moonlit

Invoeropties
j
n

Vraag	Uitleg
Passend?	Past deze frase in een wetstekst?
Verbindend?	Wordt deze frase gebruikt om zinnen te verbinden?
Proces?	Wordt deze frase gebruikt om een proces te beschrijven?
Cluster?	Staat deze frase in het juiste cluster?

Clusters	Beschrijving	Voorbeelden
0	opsommend	ten eerste, verder
1	tijdsverband	terwijl, nadat
2	tegenstelling	maar, echter, ondanks
3	vergelijking	evenals, eveneens
4	toelichting, vo	bijvoorbeeld, zoals
5	voorwaardelijk	als, indien
6	verklaring	ten gevolge van, omdat
7	samenvatting	kortom, samengevat
8	conclusie	dus, daarom
9	overig	

Frasen	Context	Cluster	Passend?	Verbindend?	Cluster?	Nieuw/additionel	Waarom?
in artikel tweede		0.0					
van artikel tweede		0.0					
in artikel derde		0.0					
te doen	te doen blijken	1.0					
ingevolge		1.0					
te nemen	in aanmerking te nemen	1.0					
daarop	daarop berustende	1.0					
omtrent		1.0					
tot stand	tot stand komen/ tot stand brengen	1.0					
boven	boven de vergoeding	1.0					
al dan niet		2.0					
dan niet		2.0					
zo spoedig	zo spoedig mogelijk	2.0					
niet is		2.0					
niet of		2.0					
waaronder		2.0					
in andere gevallen		3.0					
onderscheidenlijk		3.0					
in werking treedt		5.0					
met	met waarde/ met dien	5.0					
voor de toepassing		5.0					
al dan		5.0					
van het in		5.0					
op of	op of na	5.0					
met betrekking tot		5.0					
in geval van		5.0					
en met van		5.0					
in verband		5.0					
in overeenstemming		5.0					
mits		5.0					
of	bij of krachtens/ of een vijfde	5.0					
van toepassing op		5.0					
ter	ter beschikking/ ter voorkoming	5.0					
binnen	binnen de vastgestelde	5.0					
in aanmerking		5.0					

Figure A.1: Interview template for semi-structured domain expert interviews from Module 1

Appendix B

Module 2: Interview template

Question	Model	Activity element	Question	Answer	Notes
1.1	1	1	Considering the law text; is the question posed in the textbox correct?		
1.2			Considering the law text; is the "no" answer option correct? If not, what happens with the N.V.T?		
1.3			Considering the law text; is the "yes" answer option correct?		
1.4		Whole model	How would you rate the correspondence of the law text and the model from 1 to 10?		
2.1	2	1	Considering the law text; is the question posed in the textbox correct?		
2.2			Considering the law text; is the "no" answer option correct? If not, what happens with the N.V.T?		
2.3			Considering the law text; is the "yes" answer option correct?		
2.4		Whole model	How would you rate the correspondence of the law text and the model from 1 to 10?		
3.1	3	1	Considering the law text; is the question posed in the textbox correct?		
3.2			Considering the law text; is the "no" answer option correct? If not, what happens with the N.V.T?		
3.3			Considering the law text; is the "yes" answer option correct?		
3.4		Whole model	How would you rate the correspondence of the law text and the model from 1 to 10?		
4.1	4	1	Considering the law text; is the question posed in the textbox correct?		
4.2			Considering the law text; is the "no" answer option correct? If not, what happens with the N.V.T?		
4.3			Considering the law text; is the "yes" answer option correct?		
4.4		2	Considering the law text; is the question posed in the textbox correct?		
4.5			Considering the law text; is the "no" answer option correct?		
4.6			Considering the law text; is the "yes" answer option correct?		
4.7		3	Considering the law text; is the question posed in the textbox correct?		
4.8			Considering the law text; is the "no" answer option correct? If not, what happens with the N.V.T?		

Figure B.1: Interview template for semi-structured domain expert interviews from Module 2

Appendix C

Module 2: Sample set of extracted models

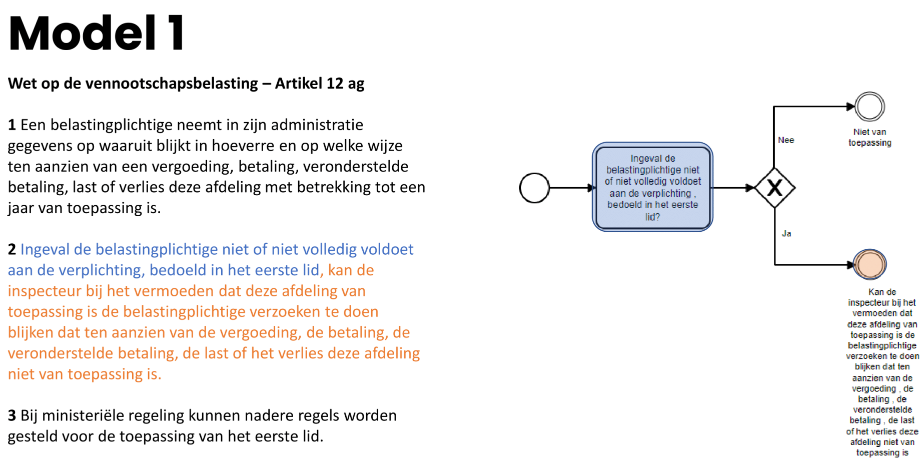

Figure C.1: Extracted model 1 from Module 2

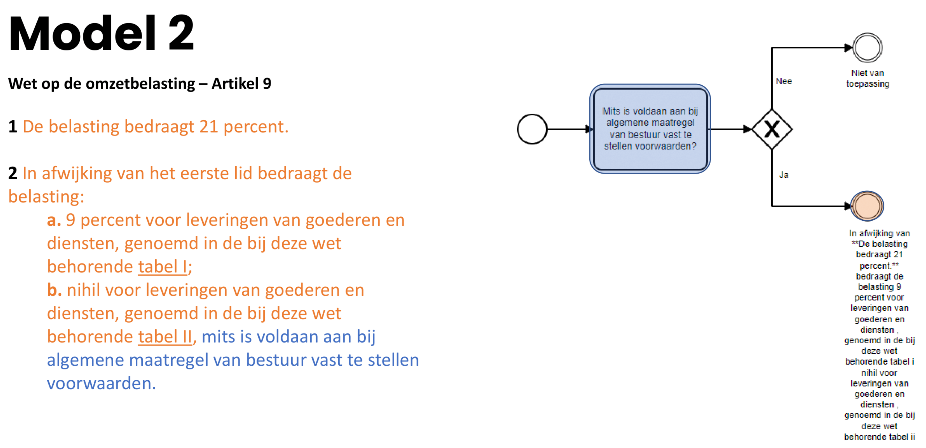

Figure C.2: Extracted model 2 from Module 2

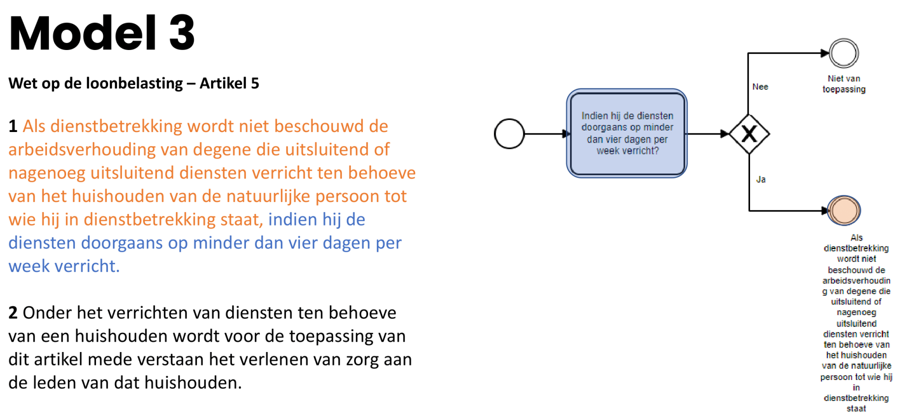

Figure C.3: Extracted model 3 from Module 2

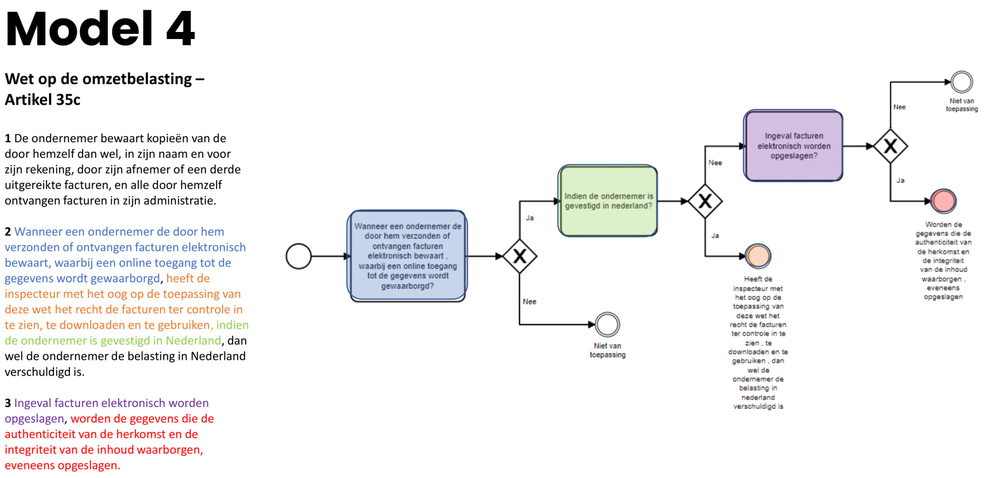

Figure C.4: Extracted model 4 from Module 2

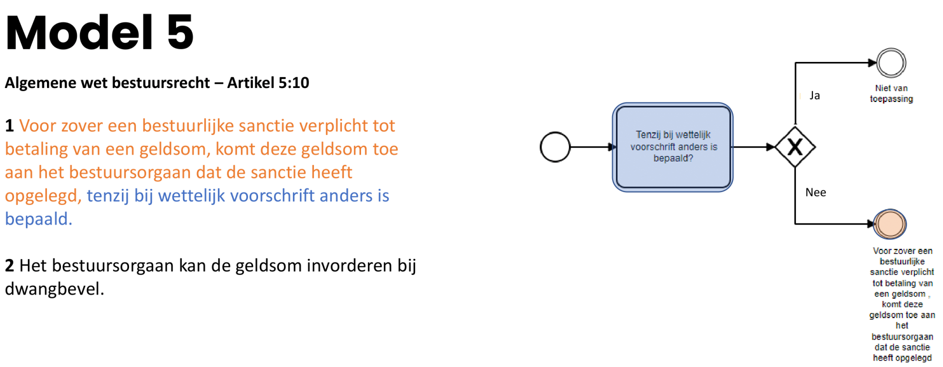

Figure C.5: Extracted model 5 from Module 2

Appendix D

Module 3: Interview template

moonlit

Type	Question	User 1	User 2	User 3	User 4	User 5
Visualization 1	Rating 1 to 10					
	Why?					
	What could be improved?					
Visualization 2	Rating 1 to 10					
	Why?					
	What could be improved?					
Visualization 3	Rating 1 to 10					
	Why?					
	What could be improved?					
Visualization 4	Rating 1 to 10					
	Why?					
	What could be improved?					
Other questions	What happens when the decision tree stops?					
	Should we visualize NVT?					
	To conclude: which visualization do you like the most and why?					

Figure D.1: Interview template for semi-structured domain expert interviews from Module 3