Czech Legal Text Treebank 2.0

Vincent Kríž and Barbora Hladká
Charles University
Faculty of Mathematics and Physics
Prague, Czech Republic
{kriz, hladka}@ufal.mff.cuni.cz

Abstract

The Czech Legal Text Treebank 2.0 (CLTT 2.0) contains texts that come from the legal domain and are manually syntactically annotated. The syntactic annotation in CLTT 2.0 is more elaborate than in CLTT 1.0. In addition, CLTT 2.0 contains two new annotation layers, namely the layer of entities and the layer of semantic entity relations. In total, CLTT 2.0 consists of two legal documents, 1,121 sentences and 40,950 tokens.

Keywords: treebank, legal, long sentences, syntactic annotations, named entities, semantic relations

1. Introduction

We have been developing approaches and systems for detecting and extracting semantic relations from unstructured texts. We have developed the RExtractor system (Kríž et al., 2014; Kríž and Hladká, 2015). This system implements an extraction pipeline which processes input texts by linguistically-aware tools and extracts entities and relations using queries over dependency trees. The language used for testing RExtractor is Czech and the legal domain was chosen to be explored in detail.

We surveyed existing syntactically annotated corpora and only a few of them contain texts from the legal domain, e.g., the Universal Dependencies v2.[1] To have a gold-standard data for the RExtractor evaluation, we created the Czech Legal Text Treebank 1.0 (Kríž et al., 2016). In total, 1,121 sentences from the Collection of Laws of the Czech Republic were annotated morphologically and syntactically in accordance with the Prague Dependency Treebank annotation framework.

In this paper, we introduce the next version of CLTT with more elaborate syntactic annotations and enriched with two annotation layers. The remainder of this paper is organized as follows: Section 2. presents a brief description of CLTT 2.0. Modifications in the syntactic annotation are described in Section 3. Section 4. describes the layer of entities and Section 5. presents the layer of semantic relations. Finally, Section 6. provides more details about getting CLTT 2.0.

2. Czech Legal Text Treebank 2.0

We provide basic characteristics of CLTT 2.0 with a special attention paid to the differences between CLTT 1.0 and CLTT 2.0.

2.1. Annotation Layers

Both CLTT 1.0 and CLTT 2.0 annotation principles fit the framework originally formulated in the Prague Dependency Treebank project (PDT, (Hajič et al., 2018)).[2] According to this annotation framework, dependency trees are annotated on the three layers:

- **Word Layer** (`w-layer`)
  A text is segmented into documents and paragraphs and individual tokens are recognized and associated with unique identifiers.
- **Morphological layer** (`m-layer`)
  A sequence of tokens of the word layer is divided into sentences. Annotation of a sentence consists of attaching several attributes to the tokens of the `w-layer`, the most important ones are morphological lemma and tag.
- **Analytical layer** (`a-layer`)
  A sentence is represented as a rooted ordered tree with labeled nodes and edges. One token from the morphological layer is represented by exactly one node in the tree and the dependency relation between two nodes is captured by an edge between the two nodes. The actual type of the relation is given as an analytical function label of the edge.

There are two new layers in CLTT 2.0:

- **Entities Layer** (`e-layer`)
  We focus on entities from the accounting domain. Each entity detected in a text is represented by (i) unique entity identifier, (ii) reference to the dictionary of accounting entities (see below), (iii) identification of the document, the sentence and the tokens where the entity was detected, and (iv) text chunk with the given accounting entity form.
- **Semantic Relations Layer** (`r-layer`)
  A relation is defined as a triple of *subject*, *predicate* and *object*, where both subject and object are accounting entities and predicate is a token (typically a verb) which represents a semantic relation. Analogously to the annotation of entities, each relation has a unique identifier and we distinguish relations of three types, definitions, obligations, and rights.

[1] `http://universaldependencies.org/`
[2] `http://hdl.handle.net/11234/1-2621`

2.2. Data format

Both CLTT 1.0 and CLTT 2.0 use the Prague Markup Language (PML) defined as a main data format by Pajas and Štěpánek (2006). The PML is a generic XML-based data format designed for representation of a rich linguistic text annotation. Both CLTT versions come with a slight modification of the PDT PML Schema.
In CLTT 2.0, the PML files contain new node attributes for entity identification (if an associated token is a part of some entity). In addition, `e-layer` and `r-layer` are stored in separate JSON files which are easily readable by both human and machines.

3. Syntactic Annotation

The syntactic annotations in CLTT 2.0 differs from the ones in CLTT 1.0 in two main aspects: (i) we fixed several errors in the dependency trees, and (ii) we modified the existing naming convention of the node identifiers so it is more readable and easy to understand.

3.1. Fixed Dependency Trees

To make manual syntactic annotation comfortable, we split long and complex sentences into *segments*. A *complex sentence* is a sentence containing at least two segments. A *segment* is a part of a sentence between two numbering markers. It might not be a complete sentence nor even a complete clause. However, its manual annotation becomes more annotator friendly.

The syntactic annotation itself was provided as manual checking and correcting the output of an automatic parser by human annotators. They checked each segment individually – both the tree structure and the analytic function assignment. After that, annotators used inter-segment links to capture dependencies between the nodes from different segments. In fact, using inter-segment links presents a way of building a dependency tree from partial dependency trees. Finally, an automatic procedure joined segment annotations into the final dependency trees for complete complex sentences.

In CLTT 2.0 we checked the dependency trees manually. We fixed several errors that came from both manual inter-segment linking and automatic processing. Unfortunately, several sentences annotated with too erroneous dependency trees had to be removed from the treebank. Thus CLTT 2.0 contains valid dependency trees.

Each dependency tree has been checked three times. The human annotator checked (i) each segment individually, (ii) each final dependency tree (before publishing CLTT 1.0) and (iii) each final dependency tree once more (before publishing CLTT 2.0). All three annotation campaigns have been done by the experienced PDT annotator. Therefore we are not able to provide inter annotator agreement.

3.2. Naming Convention of Node Identifiers

As we mentioned above, the *complex* sentences in CLTT were split into *segments* to make the treebank easier for manual annotation and manipulation. To make searching the complex sentences even more comfortable, we modified the node identifiers in CLTT 2.0 so that the identifiers contain a hierarchical structure that helps to determine the segment position in the complex sentence.

`document`Doc_{id}–`sentence`$Sent_{id}$–[`section`Sec_{id}–[`subsection`Sub_{id}]]

Figure 1: Sentence identifier schema used in the CLTT 2.0

Typically, complex sentence segments depends on each other and so we can describe their hierarchical structure. Table 1 shows a an example of typical complex sentence. In our naming convention, we define *sections* to be complex sentence segments on the first level of numbering, i.e. segments that depend on the introductory segment (line 1 in Table 1). In our example, segments on lines 2, 3 and 6 in Table 1 are *sections*. Analogously, we define *subsections* as segments that depend on a *section* as segments on lines 4 and 5 do.

A sentence identifier schema is presented in Figure 1 and it consists of the following elements:

- **Document identification** – `document`Doc_{id}
  CLTT is distributed in several files. Each sentence identifier starts with Doc_{id} to determine the PML file where the sentence is stored.
- **Sentence identification** – `sentence`$Sent_{id}$
  This identifier provides a unique sentence identification in the PML file.
- **Section identifier** – `section`Sec_{id}
  If a given sentence is complex, then the Sentence identifier determines the first level of numbering used in the complex sentence. We assign the `section0` identifier to the segment where the numbering starts.
- **Subsection identifier** – `subsection`Sub_{id}
  If a given sentence is complex, then the Subsection identifier determines the second level of numbering used in the complex sentence. We assign the `subsection0` identifier to the segment where the numbering starts.

Table 1 presents an example of the naming convention in practice. In fact, two levels of numbering (i.e., section and subsection identifiers) cover all complex sentences in CLTT. However, this strategy could be easily extended to other numbering levels.

Out of 1,121 sentences, 92 sentences were identified as complex sentences and we segmented them into 507 segments. Using the complex sentence segmentation, the average sentence length decreased from 35.9 to 26.2 tokens per sentence.

4. Entity Annotations

In CLTT 2.0, we introduced a new annotation layer of entities. We exploited the dictionary of accounting terms

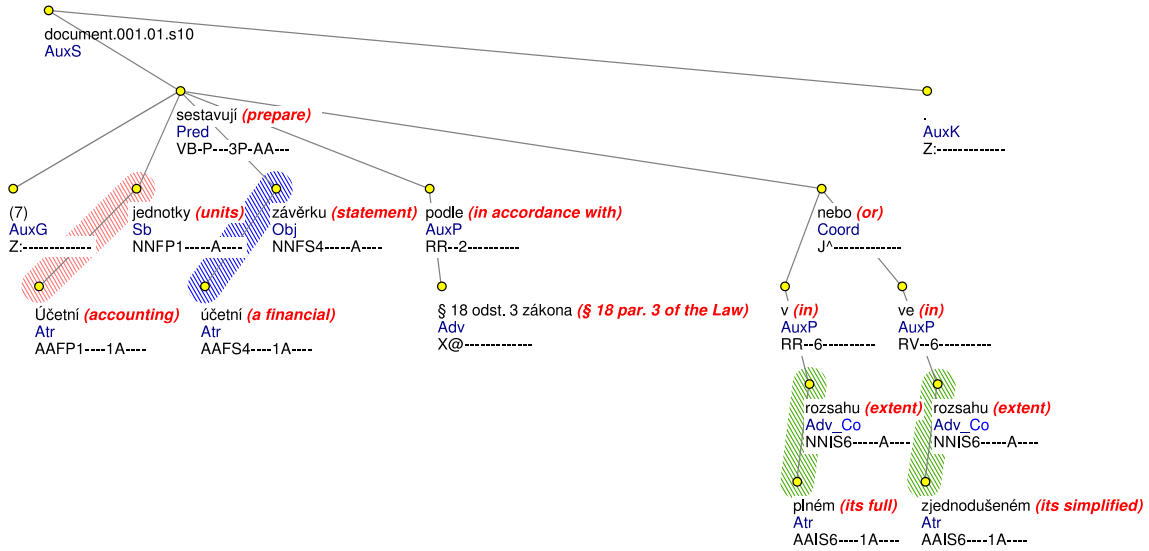

Figure 2: A sample sentence from CLTT 2.0 with highlighted accounting entities.

	Sentence sample	Node identifier prefix
1	(1) Complex sentence:	doc1-sent1-sect0
2	a) first section,	doc1-sent1-sec1
3	b) second section,	doc1-sent1-sec2-sub0
4	1. subsection,	doc1-sent1-sec2-sub1
5	2. subsection,	doc1-sent1-sec2-sub2
6	c) third subsection.	doc1-sent1-sec3
7	(2) Simple sentence.	doc1-sent2

Table 1: An example of the naming convention for the node identifiers in CLTT. The complete identifiers are abbreviated due to the lack of space, i.e., *doc* stands for *document* in the data.

that was created for the RExtractor system. Subsequently, we used the RExtractor system for automatic identification of entities in the CLTT dependency trees.

The dictionary of accounting terms consists of 1,733 different terms classified into 25 categories (see Table 2). The RExtractor system identified 7,332 occurrences in CLTT 2.0. Each detected entity is linked with the particular dictionary entry and its category.

account	general subject	obligation
accounting concept	general term	period
accounting report	incomes	regulation
activity	institution	revenues
agreement	legal person	right
assets	liabilities	state
costs	method	taxes
document	moment	
expenses	natural person	

Table 2: A list of categories in the Accounting Dictionary.

Technically, the detected entities are available in the PML files, namely see the `cltt_entity_id` attribute in the `e-layer`. It allows making tree queries with an entity specification as well as using their visual presentation in the TrEd editor (see Section 6. for more details). All detected entities are also listed in a standalone JSON file.

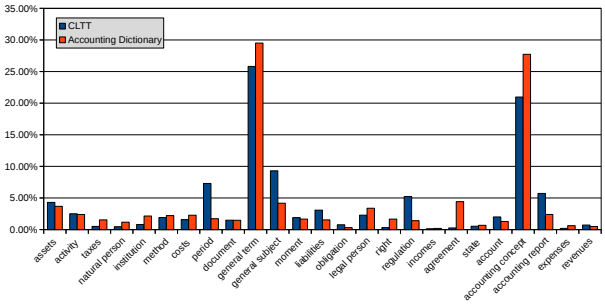

Figure 3: Accounting categories distribution in CLTT 2.0 data and in the Accounting Dictionary.

Figure 3 presents a distribution of different Accounting entities categories over the Accounting Dictionary entries as well as over the entities detected in CLTT 2.0 sentences.

5. Relations

The layer of semantic relations `r-layer` is newly introduced in CLTT 2.0. Relations are represented as (*subject*, *predicate*, *object*) triples, where *subject* and *object* have to be entities and *predicate* represents a relation. Three types of semantic relations were manually annotated in the CLTT texts:

- **Definitions**
  Relations link an entity (*subject*) and its definition (*object*).

- **Rights**
  Relations link an entity (*subject*) which have a given right (*object*) to do something.
- **Obligations**
  Relations link an entity (*subject*) which have a given obligation (*object*) to do something.

Technically, the annotated relations are available in a standalone JSON file with a simple, both human and machine readable structure. Each relation – *definition*, *right*, *obligation* – has a unique identifier. Subject and objects in the relation are represented using references to the entities in the `e-layer`. Predicates are represented by the node reference.

Relations in CLTT 2.0 have been manually annotated by one experienced annotator. As a result, CLTT 2.0 contains 483 manually annotated relations classified into 3 categories. Table 3 presents a relation types distribution and Table 4 lists the most frequent pairs of entity types that appear as relations subjects and objects.

Relation type	Frequency	
Definitions	79	16.36%
Obligations	347	71.84%
Rights	57	11.80%

Table 3: A distribution of different relation types in CLTT 2.0.

Relation	Subject type	Object type	Frequency
Oblig.	general subj.	general term	16.19%
Oblig.	general subj.	acc. concept	9.84%
Oblig.	general subj.	acc. report	8.40%
Oblig.	general subj.	acc. concept	7.17%
Oblig.	general subj.	liabilities	3.69%
Oblig.	general subj.	assets	3.48%
Oblig.	general subj.	account	3.07%

Table 4: The most frequent entity type pairs between subjects and objects.

6. Distributional Notes

CLTT 2.0 is distributed under the Creative Commons, Attribution-NonCommercial-ShareAlike 4.0 International Licence (CC BY-NC-SA 4.0).

6.1. Download

CLTT 2.0 can be downloaded from the LINDAT/CLARIN repository:

`http://ufal.mff.cuni.cz/czech-legal-text-treebank`

In addition, there are various tools for browsing and querying the treebank either locally or on-line, e.g., the TrEd graphical editor, the KonText KWIC search tool and PML TreeQuery:

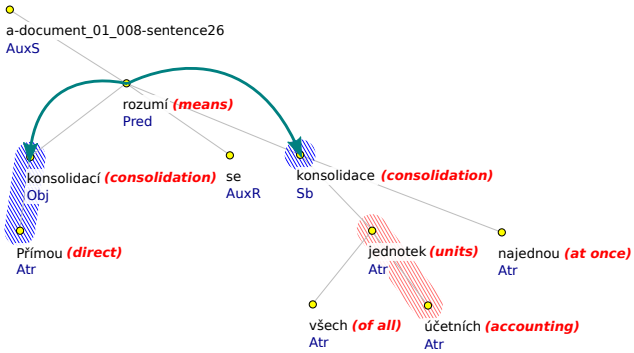

Figure 4: A sample sentence from CLTT 2.0 with the entities and relations highlighted: the *definition* relation between the *direct consolidation* and *consolidation of units*

6.2. TrEd editor

The users can view the treebank off-line using the TrEd editor[3] that we used for the manual annotation of the CLTT. We implemented a new TrEd extension *CLTT* that can be installed directly in TrEd using *Setup* → *Manage Extensions* → *Get New Extensions*.

6.3. KonText

KonText[4] is a web application for querying corpora on-line within the LINDAT/CLARIN project. Users can evaluate simple and complex queries, display their results as concordance lines, compute frequency distribution, calculate association measures for collocations and do further work with the data.

6.4. Tree Query

Tree Query[5] is a powerful open-source search tool for all kinds of linguistically annotated treebanks available on-line within the LINDAT/CLARIN project. Users can evaluate complex tree queries and display their results graphically highlighted in the dependency trees. Tree Query can be run in the TrEd editor.

7. Conclusions

The Czech Legal Text Treebank contains texts from the legal domain. Sentences in legal texts are typically long and very complex. This fact makes the treebank unique and interesting language resource.

We introduced the new version 2.0 of the treebank. It contains 1,121 sentences annotated syntactically using the Prague Dependency Treebank annotation guidelines. In addition, two annotation layers were added, namely the layer of accounting entities and the layer of semantic relations of three types – *definitions*, *rights*, and *obligations*.

CLTT 2.0 is available for free for non-commercial and academic purposes.

[3] `http://ufal.mff.cuni.cz/tred/`

[4] `https://lindat.mff.cuni.cz/services/kontext/first_form?corpname=legaltext_cs_a`

[5] `https://lindat.mff.cuni.cz/services/pmltq`

Acknowledgments

We gratefully acknowledge support from the Charles University project No. SVV 260 453. This work has been using language resources and tools developed and/or stored and/or distributed by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2015071).

Bibliographical References

Kríž, V. and Hladká, B. (2015). RExtractor: a robust information extractor. In Matt Gerber, et al., editors, *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 21–25, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kríž, V., Hladká, B., Nečaský, M., and Knap, T. (2014). Data extraction using NLP techniques and its transformation to linked data. In *13th Mexican International Conference on Artificial Intelligence, MICAI 2014, Tuxtla Gutiérrez, Mexico, November 16-22, 2014. Proceedings, Part I*, volume 8856 of *Lecture Notes in Computer Science*, pages 113–124, Switzerland. Instituto Tecnológico de Tuxtla Gutiérrez, Springer International Publishing.

Pajas, P. and Štěpánek, J. (2006). XML-based representation of multi-layered annotation in the PDT 2.0. In *Proceedings of the LREC Workshop on Merging and Layering Linguistic Information (LREC 2006)*, pages 40–47.

Language Resource References

Hajič, Jan and Bejček, Eduard and Bémová, Alevtina and Buráňová, Eva and Hajičová, Eva and Havelka, Jiří and Homola, Petr and Kárník, Jiří and Kettnerová, Václava and Klyueva, Natalia and Kolářová, Veronika and Kučová, Lucie and Lopatková, Markéta and Mikulová, Marie and Mírovský, Jiří and Nedoluzhko, Anna and Pajas, Petr and Panevová, Jarmila and Poláková, Lucie and Rysová, Magdaléna and Sgall, Petr and Spoustová, Johanka and Straňák, Pavel and Synková, Pavlína and Ševčíková, Magda and Štěpánek, Jan and Urešová, Zdeňka and Vidová Hladká, Barbora and Zeman, Daniel and Zikánová, Šárka and Žabokrtský, Zdeněk. (2018). *Prague Dependency Treebank 3.5*. Institute of Formal and Applied Linguistics, LINDAT/CLARIN, Charles University.

Kríž, V., Hladká, B., and Urešová, Z. (2016). Czech legal text treebank 1.0. In Nicoletta Calzolari, et al., editors, *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*, pages 2387–2392, Paris, France. European Language Resources Association.