Using a Czech Valency Lexicon for Annotation Support

Václav Honetschläger

Center for Computational Linguistics, Malostranské náměstí 25, 118 00 Praha, Czech Republic
honet@ufal.mff.cuni.cz

Abstract. While assigning sentences their so-called tectogrammatical representation, annotators of the Prague Dependency Treebank are also creating a valency lexicon of Czech verbs, nouns and adjectives. Until now, the information contained in it has only been used for visual checking of consistency of valency frame assigned by the annotators. We have developed an automatic procedure for pre-annotation of verbal modifications using this valency lexicon. When adding nodes into the tectogrammatical representation of sentences, our tool substantially increases the baseline recall, at the cost of only small decrease of precision.

1 Introduction

The Prague Dependency Treebank[1] (PDT, [1], [2]), being developed at the Center for Computational Linguistics[2] and Institute of Formal and Applied Linguistics[3], is a long-term research project, whose aim is a complex, linguistically motivated manual annotation of a small part of the Czech National Corpus.[4] It is being developed as a core resource for further linguistic research and for building automatic tools such as taggers, parsers, and text generators. Although the treebank is annotated manually, we do not exclude some degree of automation, especially in the later stages of annotation, when part of the truly manual annotation is already available and can be exploited in various ways.

In this paper, we would like to show how the information contained in a *valency lexicon* (the main variant of which is being created by the annotators as they go) can be used for effective automatic pre-annotation of verbal modifications, typically the most difficult part of the tectogrammatical-level annotation of a sentence.

In Sect. 2, we introduce the structure and content of PDT and the valency lexicon. We also briefly describe the annotation procedure that we want to positively influence by our tool. The algorithm on which our tool is based will be described in Sect. 3. Experiments and their results follow in Sect. 4, and finally, Sect. 5 contains some closing remarks.

1 http://ufal.mff.cuni.cz/pdt/
2 http://ckl.mff.cuni.cz
3 http://ufal.mff.cuni.cz
4 http://ucnk.ff.cuni.cz

2 The PDT, Its Valency Lexicon and the Annotation Procedure

2.1 The Prague Dependency Treebank

The PDT is being annotated on three levels: morphological, analytical (surface syntax), and tectogrammatical ("deep" syntax). The Functional Generative Description theory ([3]) has been the starting point of its annotation.

On the *morphological* level, the morphological lexical entry (represented by a *lemma*) and values of morphological categories (a *tag*, i.e. the combination of person, number, tense, gender, voice, ...) are assigned to each word.

At the second and third (*analytical and tectogrammatical*) levels of the PDT, a sentence is represented as a rooted tree. Edges represent relation of dependency[5] between two nodes—the dependent and its governor.[6]

Every token (word, punctuation) from the original text becomes a node at the analytical level of annotation. A label (*analytical function*) is assigned to every node, describing the type of surface dependency relation of the node to its parent. The original word order position of the corresponding token is also kept.

The highest, tectogrammatical level of annotation captures the deep (underlying) structure of a sentence. Nodes represent only autosemantic words; synsemantic (auxiliary) words and punctuation marks can only affect values of attributes of the autosemantic words which they are (invisibly) attached to. At this level, certain nodes might be inserted for several types of reasons. We are interested here in the case when nodes are added when valency dictates so (see 2.2 and below). Several attributes (labels) are assigned to each node; one of the most important ones is the *(deep) functor* capturing the tectogrammatical function of a child relative to its parent, i.e. the type of the modification.

2.2 The Valency Lexicon

Generally, the term *valency* indicates the capability of (autosemantic) lexical units to bind other (autosemantic) units onto itself (in the "deep" syntactic *dependency* sense); their number and type is determined for each lexical unit separately. We only deal with valency of verbs here.

Expressions (sentence constituents) which can depend on a verb are called *modifications* (thus the verb is a governor for its modifications). Modifications can be *obligatory* or *optional*. The criterion for a decision about this distinction is the so-called *dialogue test* ([7]).

Every entry in the lexicon (corresponding to one autosemantic word) contains one or more *valency frames*. Each frame usually corresponds to one meaning of the word. Each valency frame contains several *valency slots*, one for each obligatory (or, under certain conditions, also optional) modification that is supposed

[5] called 'immediate subordination' in some other theories

[6] Sometimes we denote pair of adjacent nodes as 'child–parent' since not all the edges correspond to the "proper" relation of dependency—some of them have rather technical character (e. g. edges adjacent to nodes representing punctuation marks).

associated with the particular meaning. Each slot has a label denoting the appropriate functor for the modification occupying this slot in a particular utterance. A set of possible *surface morphosyntactic forms*[7] is associated with each slot.

Example of a verb with three meanings (i.e. three valency frames):[8]

jednat {*negotiate with sb./proceed/treat sb.*}

ACTor(1) PATiens(o{about}+6) ADDRessee(s{with}+7)
- example: *jednal s nimi o smlouvě* {*he negotiated the contract with them*}

ACTor(1) MANNer[ ]
- example: *začal jednat* {*he started to proceed*}

ACTor(1) PATiens(s{with}+7) MANNer( )
- example: *jedná s ní špatně* {*he treats her badly*}

The valency lexicon we work with is called PDT-VALLEX. It captures only those meanings (and thus those frames) of verbs which occur in the annotated data. It currently contains 4457 verbs (as well as 1425 nouns and 21 adjectives).

2.3 The Tectogrammatical Annotation

The tectogrammatical annotation proceeds essentially manually, starting with a "half-baked" tectogrammatical structure as a result of a conversion from the manually annotated analytical representation performed by the AR2TR[9] tool ([6]). It seems natural to apply our tool right after AR2TR and we are doing so.

A tool using the valency lexicon can help here in two ways. First, it can determine functors of verb modifications as required by its valency frame slots, based on the morphosyntactic form of these modifications. Second, it can add missing obligatory modification(s) of the verb.

3 The Algorithm

For a given verb and its meaning all its obligatory modifications *has to be present* in the tectogrammatical representation of every sentence where this verb occurs. An optional modification need not be present there, but its entry in a valency lexicon contains information about its surface morphosyntactic form which can help us to determine its functor. However, even an obligatory modification does not need to be expressed in the surface form of a sentence (an extreme case is that one can reply to a question just with a bare verb with no modifications). This fact, i.e. the possibility of not seeing some of the modifications expressed in the original sentence, is what makes the task non-trivial.

Moreover, the valency frames corresponding to the individual meanings of the verb usually overlap; however, it is impossible to choose the correct frame

[7] Referred to by some as the *subcategorization information*

[8] The notation: a slot is described by its functor and its morphosyntactic forms in parentheses (for an obligatory modification) or in brackets (otherwise). Numbers denote Czech morphemic cases (1 is nominative, 2 is genitive, etc.) Lexical items are fully specified where required. English translations are in braces.

[9] it removes certain auxiliaries, assigns functors in clear-cut cases etc.

first and then simply deal only with the slot-to-modification alignment. Instead, we align, match and score all possible frames and try to put together pieces of information from those ones with the maximal score. The match score is based on the alignment of the (form of) possible modifications as found in the text with the morphosyntactic form(s) associated with slots in a valency frame from the lexicon. This measure has two desirable properties: (1) when no modification is expressed, the scores of all the frames are equal; (2) when there is only one frame with all modifications present, such a frame has the highest score.

The algorithm works as follows.

1. Get the morphosyntactic forms of modifications (as they appear in the data).
2. For every lexicon frame of the verb compute the alignment (and from it the match score) between slots of this frame and the modifications present in the sentence (using the surface morphosyntactic forms). Retain only the frame(s) with the maximum score.
   e. g. frame: ACTor(1) PATiens(4) ADDRessee[3] MEANS[7]
   expressed modifications: 1 (nominative), 3 (dative), 4 (accusative), v+6 (preposition "v" with locative)
   alignment: ACTor, ADDRessee, PATiens, none
   the score (total number of matches) is 3
3. Assign functors to the modifications according to the computed alignment. If more than one frame is retained, assign functors according to all such frames. Assign no functor to a modification if there are conflicting functors (but treat all these conflicting functors as if they were assigned).
   e. g. verb *připravit* {*prepare/steal*} has two frames
   ACTor(1) PATiens(4) for *prepare*
   ACTor(1) ADDRessee(4) PATiens(o+4) for *steal*
   expressed modifications: 1 (nominative), 4 (accusative)
   matches: ACTor, none (PATiens/ADDRessee conflict)
4. Add nodes (with appropriate functors) not present in the tree but matched as obligatory in (all of the) frame(s) into the tree.
5. Assign the rest of the functors.

4 Experiments and Results

We have made a series of experiments with our algorithm, using various features and parameters, evaluating them on the same test data.

Test data consists of 1641 both analytically and tectogrammatically manually annotated sentences. Since valency frames in PDT-VALLEX are updated during tectogrammatical annotation, only data more recent than PDT-VALLEX have been used to ensure fair evaluation.

We report precision, recall and F-measure results for adding nodes into the (original) tectogrammatical structure. A node is considered to be added correctly iff it is attached to the correct node *and* its functor is determined correctly. Since our tool is and always will be applied after the AR2TR procedure, we always report cumulative results obtained by serially applying both tools.

4.1 The Basic Experiments (Table 1)

On top of the baseline—the AR2TR tool—(row 1) and the algorithm described in Sect. 3 (row 2), we present results of our tool enhanced by various features. We have incorporated all the features into our tool and we report further results using them.

(a) The match between modifications appointed by certain frame and the expressed modifications is computed using obligatory modifications only.
(b) Conflicts of functors corresponding to a modification are solved by random selection of one of the conflicting functors. (We recall that *none* functor was being assigned initially.)
(c) In the valency lexicon, there are sometimes no constraints on the morphosyntactic form of a frame slot; therefore our tool could not assign the appropriate functor. In this case the forms extracted from [4], where lists of possible functors for several morphosyntactic forms are defined, are used.[10]
(d) When a verb was not found in the lexicon, the default frame containing the only modification—obligatory actor expressed by a nominative case—is assigned to it.

Table 1. Results of the basic experiments

Experiment	precision (%)	recall (%)	F-measure
AR2TR alone	86.7	17.3	28.8
basic implementation	67.5	48.3	56.3
match according to obligatory (a)	67.4	48.2	56.2
random functor when conflict (b)	69.5	48.5	57.1
extracted morphosyntactic forms (c)	69.0	48.4	56.9
default frameset (d)	68.2	49.4	57.3
the “final” method	72.5	49.5	58.8

4.2 Experiments with Another Valency Lexicon (Table 2)

Besides PDT-VALLEX, there also exists another valency lexicon, called VALLEX ([5]). It contains 1102 most frequent verbs (as found in the Czech National Corpus) with 3333 frames capturing all their meanings. It is also hand-crafted (but more thoroughly checked for consistency) and it also captures some additional syntactically relevant features of verbs. Now we want to compare the contribution of the two apparently different lexicons.

According to our expectations when using VALLEX instead of PDT-VALLEX precision increased and recall decreased (VALLEX is hand-checked, contains more meanings of individual verbs, but contains less entries). We conclude that using PDT-VALLEX is better for our purpose.

[10] From those, only those forms corresponding to prepositional phrases have been used.

Table 2. Results of experiments with another valency lexicon

Experiment	precision (%)	recall (%)	F-measure
PDT-VALLEX	72.5	49.5	58.8
VALLEX	78.6	44.0	56.4

5 Closing Remarks

We have tried to ease annotation of PDT using information from valency lexicons. When adding nodes into tectogrammatical structures of sentences, recall has substantially improved over the baseline from 17.3% to 49.5% while precision has decreased from 86.7% to 72.5% (the F-measure gain has been 30.0).[11] Our tool is currently tested by the annotators of PDT with a positive initial feedback. Its former version was used in the project of machine translation ([8]) for partial improvement of the automatically generated tectogrammatical structures, too.

One way of improving the quality of our tool is a complete understanding of the information contained in the valency lexicons—our tool cannot handle e.g. compound prepositions. Obviously, the better quality and completeness of the valency lexicon, the better results produced by our tool can be expected.

This research was supported by a grant of the Grant Agency of the Czech Republic No. 405/03/0913 and a project of the MŠMT ČR No. LN00A063.

References

1. Jan Hajič: *Building a Syntactically Annotated Corpus: The Prague Dependency Treebank*, Festschrift for Jarmila Panevová, Karolinum, Charles University Press, Prague, 1998, pp. 106–132
2. Eva Hajičová, Jan Hajič, Martin Holub, Veronika Řezníčková, Petr Pajas, Barbora Vidová Hladká, Petr Sgall: *The Current Status of the Prague Dependency Treebank*, TSD 2001, Železná Ruda, 2001, pp. 11–20
3. Petr Sgall, Eva Hajičová, Jarmila Panevová: *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects*, Academia – Kluwer, Praha – Amsterdam, 1986
4. Eva Hajičová, Jarmila Panevová, Petr Sgall: *A Manual for Tectogrammatic Tagging of the Prague Dependency Treebank*, ÚFAL/CKL Technical Report TR-2000-09, Charles University, Prague, 2000
5. Markéta Lopatková, Zdeněk Žabokrtský, Karolína Skwarska, and Václava Benešová: *Tektogramaticky anotovaný valenční slovník českých sloves*, ÚFAL/CKL TR-2002-15, Charles University, Prague, 2002
6. Alena Böhmová: *Automatic Procedures in Tectogrammatical Tagging*, in *The Prague Bulletin of Mathematical Linguistics 76*, Charles University, Prague, 2001
7. Jarmila Panevová: *On verbal frames in functional generative description*, in *The Prague Bulletin of Mathematical Linguistics 22*, pages 3–40, 1974
8. Jan Hajič et al.: *Natural Language Generation in the Context of Machine Translation*, Workshop '02 Final Report, CLSP Technical Reports, Baltimore, USA, 2002

[11] We have also evaluated functor assignment, but we have observed only insignificant increase of precision and recall.