Human Language Technologies – The Baltic Perspective
A. Utka et al. (Eds.)

doi:10.3233/978-1-61499-442-8-219

Constructions in Latvian Treebank: the Impact of Annotation Decisions on the Dependency Parsing Performance

Lauma PRETKALNIŅA[a,1] and Laura RITUMA[a]
[a]*Institute of Mathematics and Computer Science, University of Latvia*

Abstract. In this paper, we analyze the impact of various dependency representations for various constructions on the general parsing accuracy and on the parsing accuracy of these constructions. We focus on the analysis of coordination constructions, complex predicates, and punctuation mark attachment. We use Latvian Treebank as a dataset, thus, providing insight for an inflective language with a rather free word order. Experiments with MaltParser, a transition-based parser, show clear difference in learnability of various representations for the considered constructions. Future work would include carrying out comparable experiments with a graph-based dependency parser like MSTParser.

Keywords. Dependency parsing, coordination, punctuation, multiword expressions, complex predication, annotation decisions, Latvian Treebank

Introduction

Dependency parsers are among basic language processing tools. Considering a formal dependency model from the linguistic point of view, it offers a diversity of ways how to represent various constructions: while linguists tend to agree how dependency analysis should be performed on core phenomena, there are several important linguistic phenomena with no general consensus available. Various studies e.g. [1], [2] show that different dependency representations influence both parser accuracy and accuracy of the tools relaying on the parser, thus, the annotation decisions have far-reaching consequences. We explore effects of varying dependency representations for three language phenomena: coordination constructions, punctuation mark attachment and multiword predicates. When considering coordination constructions we include coordinated clauses. When considering multiword predicates we include compound tense forms, compound predicate, and predicates with modifiers. We do intrinsic parser evaluation with the focus on these constructions—we train dependency parsers on data where these constructions have been annotated in different ways and then compare accuracy scores for tokens involved in these constructions.

As a dataset we use Latvian Treebank. The native annotation model for Latvian Treebank is a dependency based hybrid model [3], [4]. The constructions considered in

[1] Corresponding Author: lauma@ailab.lv, Institute of Mathematics and Computer Science, University of Latvia; Raiņa bulv. 29, Rīga, LV-1459, Latvia

this paper are represented in Latvian Treebank by phrase structures interlinked with dependency relations, thus, making it convenient for obtaining varied dependency-only representations by use of various transformations.

For the purposes of this paper, we use MaltParser [5] due to the easy availability of implementation for both the parser and the parameter optimizer MaltOptimizer [6].

Related work includes research of Schwartz et al [7], where emphasis is on overall accuracy comparison for various parsers and various dependency representations. Compared to this study, we provide an insight for a language with a rather free word order and information about more complex dependency choices as [7] considers conjunctions with two conjuncts and a conjunction. We consider punctuation mark attachment and a broader scope of complex predicates as well. Our work can be seen as continuation of the earlier experiments [2] where extrinsic evaluation for similar dependency representations choices in similar setting is done. While other research such as [1] and [8] seek ways to eliminate influence of the representational variations on the parser evaluation, we want to compare the representational variations and find whether some of them are superior to others.

1. Latvian Treebank

Latvian is a morphologically rich inflective language with a relatively free word order. Latvian Treebank is the biggest known syntactically annotated corpus for Latvian. It is an unbalanced (approx. 1/3 newswire texts and more than 1/3 fiction texts) corpus currently containing 3,882 sentences and 53,225 tokens.

Latvian Treebank is annotated according to SemTi-Kamols hybrid grammar model [3], [4]. In this model, each sentence is represented by a dependency tree. Each node of such tree is either a token or a phrase formed from multiple units who in turn can be either tokens or phrases themselves. Thus, for each dependency link both its head and dependent can be tokens or phrases. Moreover, each phrase can consist from tokens or other phrases or a mix of them. Latvian Treebank contains detailed labeling for dependency links (26 roles), phrases (24 subtypes) and their elements (9 roles). All phrases are divided in three main categories—coordination, punctuation mark attachment and so-called x-words depicting fixed order word groups with stable structures—each consisting of several subcategories. In Latvian Treebank ellipsis is annotated within bounds of a single sentence, however, this is more like an investment for the future as this information is not used by any parsers currently. Further we shall describe in more detail the constructions whose dependency representation will be experimented with.

The coordination construction has two subtypes: coordinated parts of sentence, e.g., *milti un sviests* ‘butter and flour’ and coordinated clauses, e.g., *ūdens vārās, un tvaiki ceļas* ‘water boils and steam rises’. Coordination construction constituents are not only coordinated parts but also conjunctions and punctuation marks (very common in Latvian). For the experiments, both kinds of coordination constructions are considered.

For attaching punctuation marks to the tree, a special phrase type is introduced: punctuation mark construction. Such phrases group together one or more punctuation marks with the a token, or sometimes another phrase, invoking the usage of these punctuation marks, e.g. *Anna, spēlējot vijoli, neievēroja troksni* ‘Anna [while] playing the violin didn’t notice the noise’—the underlined elements constitute a punctuation mark construct as participle *spēlējot* ‘playing’ causes the usage of the commas. Such

constructions also include a subordination conjunction, if there is any. In the treebank, punctuation mark constructs are divided in several subtypes. In this study, we shall consider all punctuation mark constructs regardless their subtype.

The third phrase type in Latvian Treebank is an x-word. This construction is used for multiword units that have a relatively fixed word order and fixed element roles, for example, prepositional constructions and similes, named entities, expressions in other languages. In this study we consider a single x-word subtype—multiword predicates. This construction is used for annotating compound tense forms like *viņš ir bijis Latvijā* 'he has been in Latvia', compound predicates like *viņš ir skolotājs* 'he is [a] teacher', predicates with modifiers like *viņš var skriet* 'he can run' and any combination of the before mentioned. Each multiword predicate is considered as consisting of one semantically main word (*base element*) and one or more modal or auxiliary verbs.

2. Transformations

In order to use Latvian Treebank data to train a dependency parser, it must be transformed to dependency-only annotation. Hybrid-to-dependency transformations are formed as follows:

- Phrases are transformed to single rooted tree-form dependency substructures. Phrase dependents are attached to the root of the created substructure.
- Dependency links between tokens are kept intact.
- Each dependency link with a phrase as a child (or a parent) is transformed to a dependency link to the root (from the root in case of a parent) of the substructure representing that phrase.

To reduce the complexity of data, nested coordinations of the same type are substituted with a single coordination of the same type. After transforming a tree into pure dependencies, ellipsis nodes without a corresponding token are removed. During the transformation to dependency representations and ellipsis removal, labels are enriched to keep track of the information removed from the tree structure by transformations:

- All dependency labels are augmented with a prefix indicating if the dependency ark's head is a phrase or a token.
- Labels for tokens representing an ellipsis, e.g. dashes, are augmented with a prefix identifying that there is an ellipsis.
- A new label is constructed for the phrase element chosen as the root of the dependency substructure representing the given phrase in the dependency tree. It consists of dependency role, dependency type (i.e., token dependency or phrase dependency), phrase type, and the element's role within the phrase. If this element can fulfill the same dependency role without being a part of the phrase, then phrase type and the element's role within the phrase are omitted from the final label.
- Other phrase elements are labeled only with the phrase type and the element's role within the phrase.

This procedure leaves a space for decisions about specific phrase transformations—what is the best way to interconnect the constituents. In this paper,

we consider different transformations for multiword predicates, coordination constructions and punctuation mark constructs.

For coordination constructions (*coord*), we consider four transformation strategies representing three major coordination representation families (in the parenthesis, dependency representation's classifications are given according to [9]).

- *3_LEVEL*. The overall structure is three nodes (two dependency links) deep regardless the number of coordinated parts. First coordinated part is chosen as the root of the corresponding dependency subtree. Other coordinated parts are made direct children of the root. Conjuncts and punctuation marks are made direct children of the following coordinated part. (Stanford family, *fShLsHcFpFdU*)
- *DEFAULT*. Essentially, Prague Dependency Treebank [10] annotation style; it is two nodes (one dependency link) deep regardless the number of coordinated parts. The conjunction between the first two coordinated parts is chosen as the root, if there is a conjunction, otherwise—the punctuation mark between the first two coordinated parts. Other constituents are made direct children of the phrase root. (Prague family, *fPhLsHcHpBdU*)
- *ROW*. Depth of the structure is equal to the element count in the structure minus the count of the conjunctions before the first coordinated part. Each linearly next constituent after the first coordinated part is added as the children of the previous coordinated part. If there are any conjunctions before the first coordinated part, they are made children of the first coordinated part. (Moscow family, *fMhLsHcBpBdU*)
- *ROW_NO_CONJ*. Depth of the structure is equal to the coordinated part count. The first coordinated part is chosen as the root. Each linearly next coordinated part is added as a child of the previous part. Conjuncts and punctuation marks are made direct children of the following coordinated part. (Moscow family, *fMhLsHcFpFdU*)

For punctuation mark constructions and multiword predicates, there are two essentially different transformation strategies. For punctuation mark constructions (*pmc*), they are as follows:

- *BASELEM*. The word that invokes the usage of punctuation marks and/or subordination conjunction is chosen as the root of the corresponding dependency subtree. Other constituents are made direct children of the root.
- *DEFAULT*. Linearly first conjunction or punctuation mark if there are no conjunctions is chosen as the root. Other constituents are made direct children of the root.

For multiword predicates (*xpred*), it must be kept in mind that transformation to dependencies will be followed by ellipsis removal.

- *BASELEM*. If semantically main verb or nominal is not an empty ellipsis node, it is chosen as the root of the corresponding dependency subtree. Otherwise, the linearly first constituent that is not an empty ellipsis node is chosen as the root. Other constituents are made direct children of the root.
- *DEFAULT*. The linearly first constituent, which is not an empty ellipsis node, semantically main verb or nominal, is chosen as the root. If there is no such

node, the linearly first constituent that is not an empty ellipsis node is chosen as the root. Other constituents are made direct children of the root.

By combining these different construction transformations, we obtain 16 different transformations for the whole treebank. Further we shall denote treebank transformations by specifying the transformation choice for each of the above described constructions, e.g., *coord3_LEVEL & pmcBASELEM & xpredDEFAULT*.

3. Experiments

For the experiments, we use MaltParser [5]—a transition based dependency parser induction system. To get out the best of each corpus variant, we use MaltOptimizer [6] for obtaining the best configuration for each set of training data. For MaltOptimizer feature selection, we use built-in cross-validation, except for *coord3_LEVEL & pmcDEFAULT & xpredBASELEM* and *coordROW_NO_CONJ & pmcDEFAULT & xpredBASELEM*, where the final feature selection (last phase for MaltOptimizer) was done without cross-validation due to MaltOptimizer crashes. We use development corpus of 3,290 sentences, and test set drawn proportionally by genre of 369 sentences. We use both a proportional test set and a newswire test set to obtain better understanding about parser's everyday behavior as the treebank itself is not balanced.

We transform the treebank with each of 16 different transformations, and do the parser training, optimization, and testing with the obtained data sets. We evaluate each parser by three performance aspects—the overall accuracy, accuracy of tokens for each construction of interest, and accuracy of phrase dependents for each construction of interest. We report labeled accuracy score (LAS), unlabeled accuracy score (UAS) and label accuracy (LA) for each performance aspect. Tokens included in each of the constructions, we obtain with the help of native hybrid annotations in the treebank—for each phrase of interest we consider its constituents. For each constituent we consider how the subtree under this constituent is transformed to dependencies and use a token that is the root for this subtree after dependency transformations. Tokens corresponding phrase dependents are obtained in a similar way. It must be noted that tokens obtained in this way to be used for evaluation may differ for various transformed data sets.

4. Result Analysis

At first, we consider the overall accuracy (see Table 1). The results confirm that the dependency representation can have a large impact on the parser accuracy—LAS varies up to 7 percent points (pp). These results have slight differences from [2] due to an upgraded Latvian Treebank version. The best achieved results for all scores are slightly better; however, for some experiments results are worse. The overall results suggest that *coordDEFAULT* is harder to learn than other coordination representation styles. This matches the observations from [2], [7]. Moreover, we continue to observe that the role count has no direct impact on LA. However, the other trends like the best parser have changed. We assume that this instability is due to a rather complicated annotation scheme and a comparably small data set.

Table 1. The overall parser results (%).

Experiment type	Role count	Parser results		
		UAS	LAS	LA
coord3_LEVEL & pmcBASELEM & xpredBASELEM	192	71.67	61.80	70.95
coord3_LEVEL & pmcBASELEM & xpredDEFAULT	188	75.65	64.71	71.96
*coord3_LEVEL & pmcDEFAULT & xpredBASELEM**	262	71.09	63.66	71.38
coord3_LEVEL & pmcDEFAULT & xpredDEFAULT	261	74.49	65.85	72.70
coordDEFAULT & pmcBASELEM & xpredBASELEM	313	67.29	59.82	69.40
coordDEFAULT & pmcBASELEM & xpredDEFAULT	306	70.86	62.28	70.22
coordDEFAULT & pmcDEFAULT & xpredBASELEM	348	69.03	62.09	68.68
coordDEFAULT & pmcDEFAULT & xpredDEFAULT	348	71.81	63.74	69.91
coordROW_NO_CONJ & pmcBASELEM &xpredBASELEM	192	73.59	63.56	72.43
coordROW_NO_CONJ & pmcBASELEM & xpredDEFAULT	188	**76.81**	65.70	**72.88**
*coordROW_NO_CONJ & pmcDEFAULT & xpredBASELEM**	262	71.81	64.49	71.73
coordROW_NO_CONJ & pmcDEFAULT & xpredDEFAULT	261	75.13	**66.82**	**73.38**
coordROW & pmcBASELEM & xpredBASELEM	192	73.52	63.76	71.69
coordROW & pmcBASELEM & xpredDEFAULT	188	**76.54**	65.87	72.60
coordROW & pmcDEFAULT & xpredBASELEM	262	73.09	65.19	71.83
coordROW & pmcDEFAULT & xpredDEFAULT	261	75.42	65.64	71.61

Table 2. Parser results on tokens in coordination constructions / punctuation mark constructions / complex predicates (%).

Experiment type	coord			pmc			xpred		
	UAS	LAS	LA	UAS	LAS	LA	UAS	LAS	LA
coord3_LEVEL & pmcBASELEM & xpredBASELEM	53.57	44.17	60.15	66.85	60.20	73.09	68.91	61.22	69.55
coord3_LEVEL & pmcBASELEM & xpredDEFAULT	57.57	45.05	61.50	73.11	66.46	**76.33**	77.88	65.38	68.27
*coord3_LEVEL & pmcDEFAULT & xpredBASELEM**	50.00	45.86	61.09	72.25	67.90	71.83	68.59	65.71	73.72
coord3_LEVEL & pmcDEFAULT & xpredDEFAULT	56.82	53.27	**67.29**	74.79	70.10	73.88	82.69	73.40	75.64
coordDEFAULT & pmcBASELEM & xpredBASELEM	43.61	36.28	45.68	64.36	59.52	72.62	64.10	58.65	68.59
coordDEFAULT & pmcBASELEM & xpredDEFAULT	47.66	41.12	47.66	71.52	66.41	75.44	74.04	63.46	65.38
coordDEFAULT & pmcDEFAULT & xpredBASELEM	43.80	37.97	44.74	69.82	64.99	69.61	66.99	62.82	71.79
coordDEFAULT & pmcDEFAULT & xpredDEFAULT	42.80	35.70	41.87	73.41	68.58	72.99	77.88	68.27	70.19
coordROW_NO_CONJ&pmcBASELEM&xpredBASELEM	55.08	44.92	60.15	67.90	61.32	72.67	70.83	62.50	69.55
coordROW_NO_CONJ&pmcBASELEM&xpredDEFAULT	58.88	45.79	59.25	72.48	65.90	76.12	76.92	66.99	69.55
*coordROW_NO_CONJ&pmcDEFAULT&xpredBASELEM**	49.62	44.92	61.09	72.88	68.11	72.25	67.95	64.42	73.40
coordROW_NO_CONJ&pmcDEFAULT&xpredDEFAULT	53.64	49.16	62.80	74.72	69.96	73.74	**81.41**	73.40	**75.96**
coordROW & pmcBASELEM & xpredBASELEM	62.78	51.69	61.28	68.26	61.04	71.69	69.55	61.86	68.59
coordROW & pmcBASELEM & xpredDEFAULT	**64.49**	51.96	62.06	73.18	66.25	75.00	79.49	67.63	68.59
coordROW & pmcDEFAULT & xpredBASELEM	58.83	51.69	61.65	74.21	68.75	72.74	70.51	66.03	73.72
coordROW & pmcDEFAULT & xpredDEFAULT	62.06	**56.26**	66.54	**75.56**	**71.36**	75.49	85.90	**75.32**	73.72

Table 3. Parser results on dependents of coordination constructions / punctuation mark constructions / complex predicates (%).

Experiment type	coord			pmc			xpred		
	UAS	LAS	LA	UAS	LAS	LA	UAS	LAS	LA
coord3_LEVEL & pmcBASELEM & xpredBASELEM	64.29	14.29	19.64	51.20	32.80	47.20	69.09	50.47	55.21
coord3_LEVEL & pmcBASELEM & xpredDEFAULT	**78.57**	19.64	21.43	**63.20**	**44.00**	49.60	75.08	51.42	53.31
*coord3_LEVEL & pmcDEFAULT & xpredBASELEM**	64.29	19.64	26.79	41.13	30.65	49.19	65.62	47.63	53.63
coord3_LEVEL & pmcDEFAULT & xpredDEFAULT	75.00	19.64	30.36	48.39	41.13	60.48	73.19	53.00	55.21
coordDEFAULT & pmcBASELEM & xpredBASELEM	21.43	17.86	30.36	53.60	33.60	47.20	64.04	48.26	54.89
coordDEFAULT & pmcBASELEM & xpredDEFAULT	16.07	14.29	25.00	60.80	36.00	42.40	75.08	49.84	51.10
coordDEFAULT & pmcDEFAULT & xpredBASELEM	23.21	19.64	33.93	40.32	33.87	57.26	57.41	42.90	48.90
coordDEFAULT & pmcDEFAULT & xpredDEFAULT	28.57	**25.00**	**35.71**	49.19	41.13	**61.29**	69.40	50.79	52.37
coordROW_NO_CONJ&pmcBASELEM&xpredBASELEM	66.07	12.50	14.29	56.00	34.40	46.40	70.03	50.16	55.52
coordROW_NO_CONJ&pmcBASELEM&xpredDEFAULT	75.00	16.07	19.64	62.40	**44.00**	50.40	76.97	52.05	53.63
*coordROW_NO_CONJ&pmcDEFAULT&xpredBASELEM**	67.86	19.64	26.79	43.55	36.29	56.45	67.51	48.26	52.37
coordROW_NO_CONJ&pmcDEFAULT&xpredDEFAULT	69.64	17.86	25.00	50.00	41.94	**61.29**	73.19	**53.94**	**56.47**
coordROW & pmcBASELEM & xpredBASELEM	71.43	17.86	19.64	55.20	36.00	48.00	67.82	49.21	54.57
coordROW & pmcBASELEM & xpredDEFAULT	75.00	17.86	19.64	61.60	41.60	45.60	**78.23**	51.42	52.68
coordROW & pmcDEFAULT & xpredBASELEM	69.64	23.21	30.36	38.71	33.06	55.65	67.19	49.53	53.63
coordROW & pmcDEFAULT & xpredDEFAULT	71.43	21.43	30.36	42.74	36.29	55.65	72.87	51.10	51.74

Further we consider the accuracy of tokens included in each construction (see Table 2). In the transformed variants of the treebank, ~15% tokens are in coordination constructions, ~32% tokens are in punctuation mark constructions and ~7% tokens are in complex predicates.

The obtained results have noticeable trends for all constructions: *pmcDEFAULT* gives on average a 6.01pp better result than a similar configuration with *pmcBASELEM*. *xpredDEFAULT* gives on average a 5.71pp better result than *xpredBASELEM*. When considering coordination, it can be observed that *coordDEFAULT* performs at least 5.5pp worse than the next better option. On a proportionally drawn test set, comparably good results give both representations where coordinated parts are arranged in a row: *coordROW* and *coodROW_NO_CONJ*. To gain a better understanding, we have run an additional test on newswire text (223 sentences). On newswire data, *coordDEFAULT*, *pmcDEFAULT* and *xpredDEFAULT* trends are similar; however, *coordROW_NO_CONJ* turns out to be better than other *coord* alternatives.

The analysis of parser results for specific constructions allows us to see that the coordination annotation is the weak spot for all parser types. Thus, any improvement of the coordination annotation performance would lead to a major performance gain.

The final aspect where dependency annotations differ is the attachment of the elements that are annotated as phrase dependents in the native Treebank model. However, these tokens are rather rare: of all tokens there are 1% coordination dependents, 2.5% punctuation mark construction dependents, and 7% complex predicate dependents. Thus, the results obtained from these measurements (see Table 3) are less reliable. However, these results show an interesting trend in UAS for coordination dependents. Considering the tree structure, if the coordinated parts are annotated as dependents of a conjunction or punctuation mark, it makes easier to make a distinction between phrase dependents and dependents of the first/last coordinated part. Meanwhile, considering the data in Table 3, we see a major UAS fall for all *coordDEFAULT* parsers, thus, suggesting that it is hard to make this distinction for the parser, at least with a training set of this size. Due to the low LA, LAS for *coordDEFAULT* parsers are comparable to others, however, the low LA most probably is a result of the role sparsity and a small training set.

Examining results for phrase dependents, it seems, that *xpredDEFAULT* is slightly better not only for phrase constituent annotation but also for phrase dependent annotation. *pmc* shows no clear distinction, but it might be due to the small amount of samples. The obtained results regarding the complex predicates are in lines with a related observation from [7]—using the semantically main word as a dependent is suitable not only for predicates with modifiers but also for predicates with auxiliary verbs despite the language differences (e.g. in Latvian, the distinction between modal verbs and other verbs is rather fuzzy compared to English).

5. Conclusions

We compared the parsing accuracy for dependency parsers trained on the same text corpus varying the dependency annotations. We compared both the overall accuracy and the accuracy on the parts whose dependency annotations vary between dependency representations. We considered different representations for three constructions annotated as phrases in Latvian Treebank: coordination constructions, punctuation mark attachment and complex predicates.

Our results confirm that decisions made in the dependency representation have a major impact on the parsing accuracy, either considering the overall accuracy, the accuracy of varied structures or the accuracy of phrase dependents. Results show that it is easier to learn complex predicates (both constituents and phrase dependents) if the semantically main word is attached to the modifier / auxiliary verb (*xpredDEFAULT*), not vice versa.

Results confirm the established idea that the so-called Prague style coordination constructions (*coordDEFAULT* where coordinated elements are dependents of the conjunction or punctuation) are harder to learn. While from the formal aspect it should be easier to distinguish dependents of the whole coordination form the dependents of the first/last dependent in this annotation style, the practical results lead to significant UAS decrease, thus the theoretical benefit does not appear in practice. However, more research on a bigger treebank should be done on this subject as Latvian Treebank has a small number of phrase dependents for coordination constructions and punctuation mark attachment phrases, making results unsuitable for fine-grade analysis.

Acknowledgements

This research was partially supported by Project No.2DP/2.1.1.1.0/13/APIA/VIAA/014 (ERDF) “Identification of relations in newswire texts and graph visualization of the extracted relation database” under contract Nr. 1/5-2013, LU MII Nr. 3-27.3-5-2013 and by Latvian 2010–2014 National Research Program No. 2 project No. 5.

References

[1] A. Søgaard, An empirical study of differences between conversion schemes and annotation guidelines. *Proc. of the 2nd International Conference on Dependency Linguistics, DepLing* (2013), 298–307

[2] L. Pretkalniņa, A. Znotiņš, L. Rituma, D. Goško, Dependency Parsing Representation Effects on the Accuracy of Semantic Applications—an Example of an Inflective Language. *Proc. of the 9th International Conference on Language Resources and Evaluation, LREC* (2014), 4074–4081

[3] L. Pretkalniņa, L. Rituma, Syntactic Issues Identified Developing the Latvian Treebank. *Proc. of the 5th International Conference on Human Language Technologies—the Baltic Perspective, HLT,* Frontiers in Artificial Intelligence and Applications **247**, (2012), pp. 185–192.

[4] G. Bārzdiņš, N. Grūzītis, G. Nešpore, B. Saulīte,. Dependency-Based Hybrid Model of Syntactic Analysis for the Languages with a Rather Free Word Order. *Proc. of the 16th Nordic Conference of Computational Linguistics* (2007), 13–20.

[5] J. Nivre, M. Kuhlmann, J. Hall, An Improved Oracle for Dependency Parsing with Online Reordering. *Proceedings of the 11th International Conference on Parsing Technologies, IWPT* (2009), 73–76.

[6] M. Ballesteros, J. Nivre, MaltOptimizer: An Optimization Tool for MaltParser. *Proc. of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics, EACL* (2012), 58–62.

[7] R. Schwartz, O. Abend, A. Rappoport, Learnability-based syntactic annotation design. *Proc. of the International Conference on Computational Linguistics, COLING* (2012), 2405–2422

[8] R. Schwartz, O. Abend, R. Reichart, A. Rappoport, Neutralizing Linguistically Problematic Annotations in Unsupervised Dependency Parsing Evaluation. *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL-HLT* **1** (2011), 663–672.

[9] M. Popel, D. Mareček, J. Štěpánek, D. Zeman, Z. Žabokrtský, Coordination Structures in Dependency Treebanks. *Proc. of the 51th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL-HLT* **1** (2013), 517–527.

[10] J. Hajič, A. Böhmová, E. Hajičová, B. Vidová Hladká, The Prague Dependency Treebank: A Three-Level Annotation Scenario. A. Abeillé (ed.): *Treebanks: Building and Using Parsed Corpora* (2000), 103–127.