Assessing the Annotation Consistency of the Universal Dependencies Corpora

Marie-Catherine de Marneffe
Linguistics Department
The Ohio State University
Columbus, OH, USA
mcdm@ling.ohio-state.edu

Matias Grioni
Computer Science Department
The Ohio State University
Columbus, OH, USA
grioni.2@osu.edu

Jenna Kanerva and **Filip Ginter**
Turku NLP group
University of Turku
Finland
{jmnybl,figint}@utu.fi

Abstract

A fundamental issue in annotation efforts is to ensure that the same phenomena within and across corpora are annotated consistently. To date, there has not been a clear and obvious way to ensure annotation consistency of dependency corpora. Here, we revisit the method of Boyd et al. (2008) to flag inconsistencies in dependency corpora, and evaluate it on three languages with varying degrees of morphology (English, French, and Finnish UD v2). We show that the method is very efficient in finding errors in the annotations. We also build an annotation tool, which we will make available, that helps to streamline the manual annotation required by the method.

1 Introduction

In every annotation effort, it is necessary to make sure that the annotation guidelines are followed, and crucially that similar phenomena do receive a consistent analysis within and across corpora. Given the recent success of the Universal Dependencies (UD) project[1] which aims at building cross-linguistically consistent treebanks for many languages and the rapid creation of 74 corpora for 51 languages supposedly following the UD scheme, investigating the quality of the dependency annotations and improving their consistency is, more than ever, of crucial importance.

While there has been a fair amount of work to automatically detect part-of-speech inconsistent annotations (i.a., Eskin (2000), van Halteren (2000), Dickinson & Meurers (2003a)), most approaches to assess the consistency of dependency annotations are based on heuristic patterns (i.a., De Smedt et al. (2016) who focus on multi-word expressions in the UD v1 corpora (Nivre et al., 2016)). There exists a variety of querying tools allowing to search dependency treebanks, given such heuristic patterns (i.a., SETS (Luotolahti et al., 2015); Grew (Bonfante et al., 2011); PML TreeQuery (Štěpánek and Pajas, 2010); ICARUS (Gärtner et al., 2013)). Statistical methods, such as the one of Ambati et al. (2011), are supplemented with hand-written rules. While approaches based on heuristic patterns work extremely well to look for given constructions (e.g., clefts) or check that specific guidelines are taken into account (e.g., auxiliary dependencies should not form a chain in UD), such approaches are limited to finding what has been defined a priori.

In this paper, we adapt the method proposed by Boyd et al. (2008) to flag potential dependency annotation inconsistencies, and evaluate it on three of the UD v2 corpora (English, French and Finnish). The original Boyd et al. method finds pairs of words in identical context that vary in their dependency relation. We show that this method works fairly well in finding annotation errors, within a given corpus. We further hypothesize that using lemmas instead of word forms would improve recall in finding annotation errors, without a detrimental effect on precision. We show that our intuition is valid for languages that are not too morphologically-rich, like English and French, but not for Finnish.

We also examine whether we can extend the method by leveraging the availability of large corpora which are automatically dependency-annotated to identify more inconsistencies than when restricting ourselves only to the given manually annotated corpus. We find that when based on automatic rather than manual annotation, the precision drops but not excessively so, but the gain in recall is rather moderate.

Finally, the Boyd et al. approach is semi-automatic, flagging potential inconsistencies

[1] http://universaldependencies.org

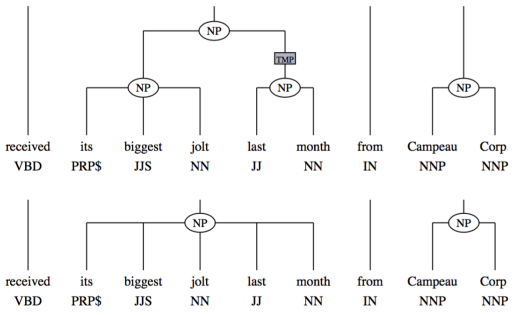


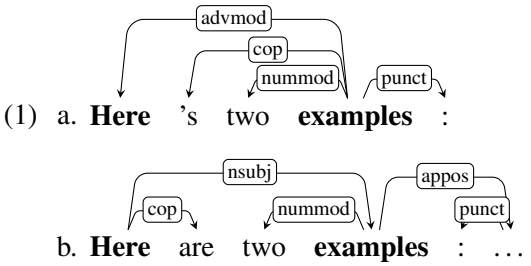
Figure 1: Example of variation nuclei for phrase-structure tree from (Boyd et al., 2008).

which require manual validation. To help streamline this manual validation process, we develop a visualization and annotation tool for the task, available to the UD community, with data for all UD treebanks.[2] Rather than a standalone tool such as ICARUS (Thiele et al., 2014), we provide an accessible browser-based interface.

2 Boyd et al. 2008: Variation nuclei

Boyd et al. (2008) extend, to dependency representation, the concept of *variation nuclei* developed by Dickinson and Meurers (2003b; 2005) for identifying inconsistent annotations in phrase-structure trees. Variation nuclei are elements which occur multiple times in a corpus with varying annotation. For phrase-structure trees, a variation nucleus is any n-gram for which bracketing or labeling varies, with one shared word of context on each side of the n-gram. Figure 1, from Boyd et al. (2008), shows an example of a 5-gram, *its biggest jolt last month*, which receives two different analyses in the Penn TreeBank.

For dependency representation, the basic elements are dependencies, i.e. pairs of words linked by a labeled dependency. Here variation nuclei are then pairs of words which are linked by different relations. However flagging any pairs of words linked by different relations would generate too many potential inconsistencies, most of which might be genuine ambiguities and not annotation errors. To restrict the number of potential inconsistencies, Boyd et al. add context restrictions. Their "non-fringe heuristic" requires the words in the nucleus to share the same context (one word to the left and one word to the right of the nucleus). Example (1) shows a variation nucleus in a dependency representation, extracted from the UD English corpus, where the pairs of words *Here* and *examples* are linked differently. Boyd et al. also experimented with a "dependency context heuristic" requiring the governors of the dependency pairs to have the same incoming dependency relation. They also considered the case of pairs of words which are linked by a dependency relation in some instances and not linked by any relation in other instances, but required for those cases that the internal context between the two words be exactly the same.



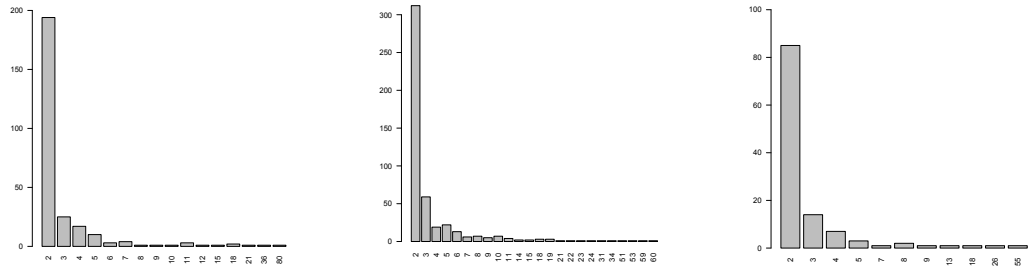
3 Extending to lemmas

Our goal in this paper is two-fold: evaluate the Boyd et al. method on the UD data, and increase recall of finding annotation errors without sacrificing precision. So far we have restricted our evaluation to words that are linked by different existing dependency relations, evaluating the "non-fringe" and "dependency context" heuristics. Boyd et al. applied their method to words (tokens). We hypothesized that to reduce data sparsity and thus find more errors, we could use lemmas instead of words, and contrary to Boyd et al., we do not require that the part-of-speech of the lemmas match. Note that the Boyd et al. method is independent of the dependency representation chosen.

4 Data

We evaluate our reimplementation and extension of the Boyd et al. method on three different languages: English, French and Finnish. We chose these three languages because they vary in their degree of morphology, and are therefore good candidates to properly evaluate the impact of using lemmas instead of words. We used the UD v2 corpora of English, French and Finnish. Table 1 gives the size of these corpora in terms of number of sentences and tokens. For the purpose of finding inconsistencies in the annotations, we collapse all the data sets (train, development, and test) available into one corpus for each language.

[2] http://www.universaldependencies.org/fixud



(a) English (266 lemma pairs) (b) French (474 lemma pairs) (c) Finnish (117 lemma pairs)

Figure 2: Number of lemma pairs (y-axis) displaying different numbers of potentially erroneous trees (x-axis).

UD v2	# sentences	# tokens
English	14,545	229,753
French	16,031	392,230
Finnish	13,581	181,138

Table 1: Size of the UD v2 English, French and Finnish corpora.

5 Evaluation

The method retrieves 266 pairs of lemmas displaying inconsistencies for English, 474 for French and 117 for Finnish, using the "non-fringe" heuristic (i.e., the pairs need to share context: same lemma to the left and same lemma to the right of the lemmas in the dependency pair). Each pair varies in the number of inconsistent trees they are associated with. But most pairs contain two trees, as can be seen in Figure 2 which shows the counts of pairs (y-axis) for the different numbers of trees they contain (x-axis).

For each language, to evaluate how many of the inconsistencies flagged are indeed annotation errors, we randomly sampled 100 of the pairs retrieved and annotated all the trees associated with these pairs, nevertheless limiting to 10 trees per dependency type.

5.1 Lemma-based approach

Table 2 gives the results. In the "non-fringe" column, we computed how many of the 100 pairs do contain erroneous trees. Thus these results indicate how precise the method is. Boyd et al. propose an additional, more stringent heuristic of "dependency context". This heuristic requires the word/lemma pairs to not only share the left/right context, but also the incoming relation type. As we did not implement this heuristic when selecting the trees for annotation, we are able to evaluate its precision as well as its recall relative to the pairs retrieved when using only the "non-fringe" heuristic. Using the 100 pairs annotated in each language as a gold-standard, we calculated the precision and recall of the "dependency context" heuristic by examining which pairs are left when adding the further requirement of shared incoming relation to the governor.

For the method used on lemmas, the results are satisfying for both English and French, with a precision of 62% and 65%, respectively. However the method is not precise enough for Finnish, with only 19% of the pairs containing annotation errors. The use of lemmas for Finnish loses too much information: different inflections in Finnish can have completely different roles in many cases, and this leads to many false positives being retrieved. A good example of this is relative clauses, where the Finnish relativizer lemmas *joka* and *mikä* get different syntactic functions depending on the case inflection. For example, in the relative clauses "joka (Case=Nom) tarvitsee" *who needs*, "jota (Case=Par) tarvitsee" *what is needed* and "jossa (Case=Ine) tarvitsee" *where something is needed*, three different syntactic functions, "nsubj", "obj" and "obl" respectively, are correctly assigned for the same lemma pair.

The more stringent heuristic of "dependency context" leads to a loss in recall (especially for French with only 47%) without a clear boost in precision. These results are in line with the results from Boyd et al. who evaluated their method on Czech (one portion of the Prague Dependency Treebank, (Böhmová et al., 2003)), Swedish (Talbanken05, (Nivre et al., 2006)) and German (Tiger Dependency Bank, (Forst et al., 2004)). For the Czech data (38,482 sentences – 670,544 tokens),

	LEMMAS			WORDS	
	"Non-fringe"	"Dependency context"		"Non-fringe"	
	Precision (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
English	62	76	66	72	79
French	65	64	47	76	73
Finnish	19	21	81	72	75

Table 2: Results of the Boyd et al. method on 100 pairs in each corpus for the "non-fringe" and "dependency context" heuristics when using lemmas as well as for the "non-fringe" heuristic when using wordforms. Recall is always reported relative to the "non-fringe" lemma-based method.

Boyd et al. obtained 58% precision on 354 pairs retrieved, increasing precision slightly to 61% when adding the more stringent heuristic, but with a recall of 66%. For the Swedish data (11,431 sentences – 197,123 tokens), 210 pairs were retrieved, with a high precision of 92%. The more stringent heuristic yielded a slight increase in precision (95%) but an important drop in recall (48%). For German (1,567 sentences – 29,373 tokens) however, due to the small corpus size, only 3 pairs were retrieved, all containing annotation errors.

5.2 Wordform-based approach

Capitalizing on the fact that every identified pair of words is also among the pairs of lemmas, we can subset the manually annotated lemma pairs and compute the precision of the method using wordforms as well as its recall relative to the lemma-based method. The results of the method based on words (instead of lemmas) are shown in the last columns of Table 2. For English and French, we see a moderate gain in precision whereas for Finnish we see a dramatic gain in precision, from 19% to 72%. The recall of the wordform-based method is in the 70–80% range for all languages, meaning that the gain in precision is offset by a loss of 20–30% of identified annotation errors. As the task is to find as many annotation errors as possible, the loss of 20–30% of identified annotation errors might not be justified, especially for English and French where it is not accompanied by a major gain in precision.

5.3 Delexicalized approach

Seeing that for Finnish, new strategies need to be explored, we also test a delexicalized version of the method, whereby only pairs of morphological features are considered, rather than wordforms or lemmas, but constrained on the context lemmas. For instance, in Figure 3, instead of using the wordform or lemma, we work at the level of the morphological features: the elements in the pairs share the same features, and the left and right contexts have identical lemmas. For English and French, initial inspection of the results revealed a hopeless over-generation, but for Finnish this method outperforms the lemma-based approach both in precision and recall. While the lemma-based method identifies 117 pairs with precision of 19%, the delexicalized version identifies 353 pairs with precision of 25%. This shows that when applying the method to Finnish, the morphology is of primary consideration, even above the lemmas themselves. Nevertheless for Finnish, the more useful method is the original Boyd et al., which considers wordforms, given that it reaches a high enough precision.

5.4 Analysis of the errors retrieved

We give here a few examples of the pairs retrieved which accurately pointed to errors in the annotations. In all examples, we bold the words that constitute the word/lemma pairs. Examples in (2), (3), (4), (5) and (6) display trees in which two very different analyses have been given to the same construction. Such trees indicate that some specific constructions in the corpus need to be systematically checked: for instance, (3) shows that comparatives in the UD French corpus need to be checked for consistency in their analysis, and (4) shows that Fr. "ce qui" *that which* needs to be checked across the board. Similarly (5) shows that number constructions in the Finnish corpus are not consistent in the choice of the head. Thus the examples flagged are useful to write patterns to check the annotations of some constructions that we may not have been thinking of a priori. (6) shows a case where there is a disagreement in the

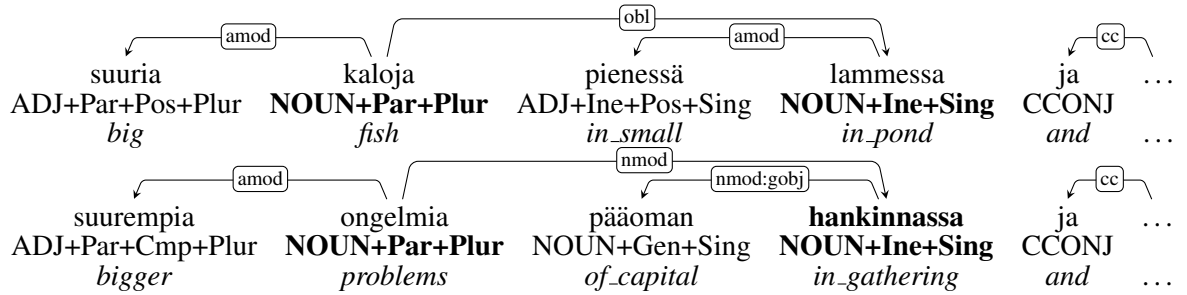
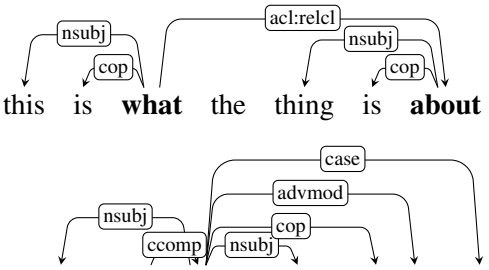


Figure 3: An example of an annotation error identified by the delexicalized method in the Finnish corpus. Here a pair of words is identified sharing a lemma-based context (*big, and*) such that the first word is a noun in plural partitive and the second word is a noun in singular inessive.

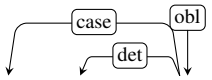
dependency type in identical phrase constructions. As the "obl" relation type has only been introduced in the recent version of the UD guidelines, it may be more error prone at this point.

(2) a. this is **what** the thing is **about**

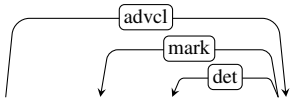


b. This store is **what** Colorado is all **about**

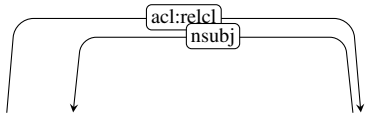
(3) a. … meilleur **que** le **précédent** .
better than the former



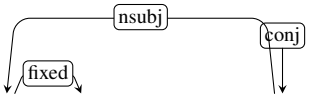
b. … meilleur **que** la **précédente** .
better than the former



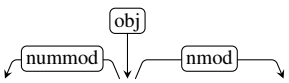
(4) a. … **ce** **qui** n' est guère **élevé** .
that which NEG is not high

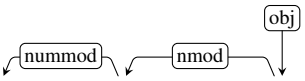


b. … **ce** **qui** est peu **élevé** .
that which is little high



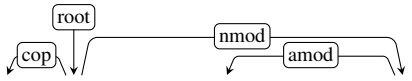
(5) a. … tuhansia **euroja** **jäsenmaksuja**
thousands of euros of subscriptions

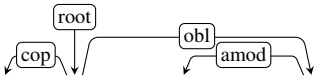




b. … tuhansia **euroja** **jäsenmaksuja**
thousands of euros of subscriptions

(6) a. on **yksi** katsotuimpia **tv-sarjoja**
is one of the most watched tv series





b. … on **yksi** pahimpia **ongelmia**
is one of the worst problems

Some errors are due to wrong attachments, such as (7) in which *able* is wrongly attached to *had* with a "ccomp" relation instead of being attached to *idea*.

(7) We **had** a pretty good idea when we signed the contract that ECS would not be **able** to complete that by the contract start date, …

The total number of annotation errors identified during the annotation of the 100 lemma pairs for each of the three corpora is summarized in Table 3. The annotation took a maximum of two hours per language and was carried out by annotators well versed in the task.

6 Extending with parsebank data

The Boyd et al. method is very useful to find annotation errors when there are similar contexts within the corpus. We examine whether we can take advantage of existing large parsebank data to find more contexts in which analyses differ, and thus hopefully catch more annotation errors in the UD data. We used the CoNLL'17 Shared Task supporting data (Ginter et al., 2017), comprising of up to several billions of words of web-crawled data

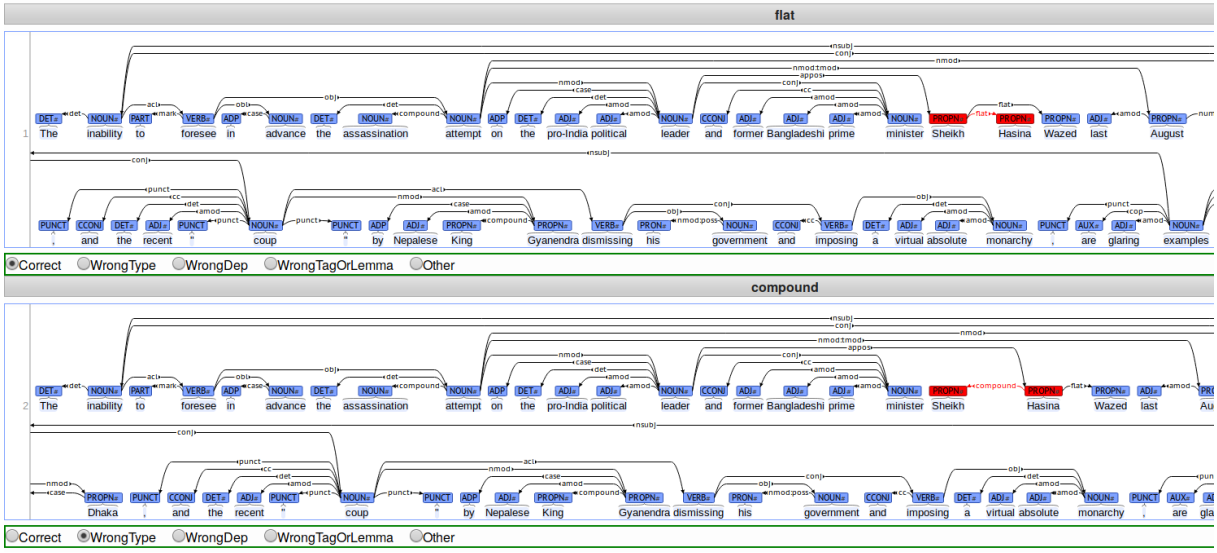


Figure 4: Example of the annotation tool.

	Erroneous		Correct
	Type	Structure	
English	63	13	223
French	56	45	241
Finnish	7	12	259

Table 3: The number of trees assessed as erroneous (incorrect relation type or incorrect structure), and the number of trees verified to be correct.

per UD language, parsed with the UDPipe 1.1 dependency parser (Straka et al., 2016). For each of the three UD corpora we analyze, we flag pairs if they appear in the corresponding parsebank data in the same context at least 5 times, but are a variation nucleus. Table 4 gives the number of trees which were manually assessed as annotation errors, as well as the percentage of trees which contain annotation errors (out of 100 pairs randomly sampled for French, all of them for English and Finnish). It also indicates how many of the erroneous trees are already found based on the treebank itself. The proportion of such erroneous trees ranges from 30% to 40% depending on the language, but this means that 60–70% of annotation errors found based on the parsebank data are not flagged by the Boyd et al. method, when operating only within the same corpus.

7 Annotation tool

The method retrieves pairs that display different analyses. However the pairs retrieved need to be checked manually: are they annotation errors or genuine ambiguities? To facilitate the annotation, we implemented a web-based tool which allows the annotation of the flagged inconsistencies to be carried out entirely in the browser in an intuitive manner. The tool is illustrated in Figure 4. First, the annotators are presented with a list of lemma pairs, sortable by various criteria. For each pair, a link is provided leading to visualizations of the trees involving the pair, which is highlighted in every tree. The trees are grouped by dependency relation, which very often results in consistent groups where every tree is correct or every tree is incorrect, thus streamlining the annotation. For each tree, the annotator can mark the

	# tree errors	"Non-fringe" Precision	% in Boyd
English	54	41%	38%
French	74	57%	36%
Finnish	10	16%	30%

Table 4: Results using parsebank data and lemmas: the number of trees that were manually assessed as annotation errors, the precision of the method, and the percentage of the erroneous trees which would be also found based on the treebank itself.

tree as *correct* or *incorrect* for three separate reasons (relation type, governor/dependent, or part-of-speech), or the catch-all category *other*. The choice is saved automatically, and retrieved in case the page with the trees is reloaded or reopened. A visual cue in the form of a green border is given to assure the annotator that the choice was successfully saved.

8 Conclusion

We evaluated the Boyd et al. (2008) method for finding annotation errors in dependency corpora on three of the UD v2 datasets (English, French and Finnish), and showed that this method performs fairly well.

We tried to adapt the Boyd et al. method to retrieve more errors, by working at the level of lemmas instead of wordforms. While results seem to indicate that this can work for languages with no case marking, it is clearly failing for a morphologically-rich language such as Finnish.

The parsebank-based method did not at present result in a large increase in recall, likely in part due to a too strict cut-off on the minimal number of parsebank instances needed in order to flag a treebank relation as inconsistent, and in part due to the noise in the automated parses of the web data. The winning system of the CoNLL'17 Shared Task[3] gains 8 percents in Labeled Attachment Score (LAS) over the baseline system which produced the parsebank analyses that we used, giving hope that this winning parser will lead to better results for the parsebank-based method.

We developed an easy and intuitive web interface for manual verification of the identified inconsistencies. Given our encouraging results on the three UD treebanks, we make both the interface and the automatically identified inconsistencies available to the UD community for all of the 70+ UD treebanks. This will allow us to expand the effort to the larger UD community and cover a number of languages and treebanks. For this, we will implement a light-weight user management so that multiple annotations for a single tree can be aggregated if necessary.

Our work is restricted to assessing the annotation consistency within a given corpus. However, moving forward, ensuring that similar constructions across corpora and languages are given the same analysis will also need to be addressed.

[3] http://universaldependencies.org/conll17/results.html

Acknowledgments

We are grateful to Alane Suhr for her help on this project early on. We thank our anonymous reviewers for their useful feedback. The work was supported by a Google Faculty Award to the first author and by the KONE Foundation and the Finnish Academy.

References

Bharat Ram Ambati, Rahul Agarwal, Mridul Gupta, Samar Husain, and Dipti Misra Sharma. 2011. Error detection for treebank validation. *Proceedings of the 9th Workshop on Asian Language Resources*, pages 23–30.

Alena Böhmová, Jan Hajič, Eva Hajičová, and Barbora Hladká. 2003. The Prague dependency treebank. In *Treebanks*, pages 103–127. Springer.

Guillaume Bonfante, Bruno Guillaume, Mathieu Morey, and Guy Perrier. 2011. Modular graph rewriting to compute semantics. In *Proceedings of the Ninth International Conference on Computational Semantics*, pages 65–74.

Adriane Boyd, Markus Dickinson, and W Detmar Meurers. 2008. On detecting errors in dependency treebanks. *Research on Language & Computation*, 6(2):113–137.

Koenraad De Smedt, Victoria Rosén, and Paul Meurer. 2016. Studying consistency in UD treebanks with INESS-Search. In *Fourteenth Workshop on Treebanks and Linguistic Theories (TLT14)*.

Markus Dickinson and W. Detmar Meurers. 2003a. Detecting errors in part-of-speech annotation. In *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics*, pages 107–114.

Markus Dickinson and W. Detmar Meurers. 2003b. Detecting inconsistencies in treebanks. In *Proceedings of the Second Workshop on Treebanks and Linguistic Theories (TLT-03)*.

Markus Dickinson and W. Detmar Meurers. 2005. Detecting errors in discontinuous structural annotation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 322–329.

Eleazar Eskin. 2000. Detecting errors within a corpus using anomaly detection. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, pages 148–153.

Martin Forst, Núria Bertomeu, Berthold Crysmann, Frederik Fouvry, Silvia Hansen-Schirra, and Valia Kordoni. 2004. Towards a dependency-based gold standard for German parsers – the TiGer Dependency Bank. In *Proceedings of the COLING Workshop on Linguistically Interpreted Corpora (LINC04), Geneva, Switzerland*.

Markus Gärtner, Gregor Thiele, Wolfgang Seeker, Anders Björkelund, and Jonas Kuhn. 2013. ICARUS – An extensible graphical search tool for dependency treebanks. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Filip Ginter, Jan Hajič, Juhani Luotolahti, Milan Straka, and Daniel Zeman. 2017. CoNLL 2017 shared task - automatically annotated raw texts and word embeddings. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University.

Juhani Luotolahti, Jenna Kanerva, Sampo Pyysalo, and Filip Ginter. 2015. SETS: Scalable and efficient tree search in dependency graphs. In *NAACL Demo*, pages 51–55.

Joakim Nivre, Jens Nilsson, and Johan Hall. 2006. Talbanken05: A Swedish treebank with phrase structure and dependency annotation. In *Proceedings of the fifth international conference on Language Resources and Evaluation (LREC-06)*.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*, pages 1659–1666.

Jan Štěpánek and Petr Pajas. 2010. Querying diverse treebanks in a uniform way. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*.

Milan Straka, Jan Hajič, and Jana Straková. 2016. UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, pos tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, Paris, France, May. European Language Resources Association (ELRA).

Gregor Thiele, Wolfgang Seeker, Markus Gärtner, Anders Björkelund, and Jonas Kuhn. 2014. A graphical interface for automatic error mining in corpora. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 57–60. Association for Computational Linguistics.

Hans van Halteren. 2000. The detection of inconsistency in manually tagged text. In *Proceedings of the 2nd Workshop on Linguistically Interpreted Corpora*.