Assessing the Impact of Incremental Error Detection and Correction. A Case Study on the Italian Universal Dependency Treebank

Chiara Alzetta[⋆], **Felice Dell'Orletta**[◇], **Simonetta Montemagni**[◇], **Maria Simi**[•], **Giulia Venturi**[◇]
[⋆]Università degli Studi di Genova
[◇]Istituto di Linguistica Computazionale "Antonio Zampolli" (ILC-CNR), Pisa
ItaliaNLP Lab - *www.italianlp.it*
[•]Dipartimento di Informatica, Università di Pisa
chiara.alzetta@edu.unige.it,
{felice.dellorletta,simonetta.montemagni,giulia.venturi}@ilc.cnr.it
simi@di.unipi.it

Abstract

Detection and correction of errors and inconsistencies in "gold treebanks" are becoming more and more central topics of corpus annotation. The paper illustrates a new incremental method for enhancing treebanks, with particular emphasis on the extension of error patterns across different textual genres and registers. Impact and role of corrections have been assessed in a dependency parsing experiment carried out with four different parsers, whose results are promising. For both evaluation datasets, the performance of parsers increases, in terms of the standard LAS and UAS measures and of a more focused measure taking into account only relations involved in error patterns, and at the level of individual dependencies.

1 Introduction

Over the last years, many approaches to detect errors and inconsistencies in treebanks have been devised (Dickinson, 2015). They can be categorized in two main groups, depending on whether the proposed quality check procedure relies on heuristic patterns (Dickinson and Meurers, 2003, 2005; Boyd et al., 2008) or on statistical methods (Ambati et al., 2011). More recently, the Universal Dependencies (UD) initiative (Nivre, 2015) has yielded a renewed interest as shown by the methods and tools introduced by de Marneffe et al. (2017); Alzetta et al. (2018); Wisniewski (2018). A number of reasons prompted the importance of these methods: they can be useful to check the internal coherence of the newly created treebanks with respect to other treebanks created for a same language or to the annotation guidelines. The risk of inconsistencies or errors is considerable if we consider that 70% of the released UD treebanks originate from a conversion process and only 29% of them has been manually revised after automatic conversion. In this paper, we extend the method proposed by Alzetta et al. (2018) for error detection and correction in "gold treebanks" and we evaluate its impact on parsing results.

2 Incremental Approach to Error Detection

Detection of annotation errors is often depicted as a two–stage static process, which consists in finding errors in a corpus and correcting them. Dickinson and Tufis (2017) provide a broader view of the task of improving the annotation of corpora, referred to as *iterative enhancement*: "iterative enhancement encompasses techniques that can be iterated, improving the resource with every pass". Surveyed methods for iterative enhancement are applied to both corpora with (mostly) completed annotation and corpora with in–progress annotation. In our opinion, the strategy of iterative enhancement is particularly relevant in the construction of treebanks which result from the conversion of pre-existing resources, as it is more often the case, and/or whose annotation scheme is continuously evolving e.g. to accommodate new linguistic phenomena or to increase cross-lingual consistency, as it happens in the Universal Dependencies (UD) initiative[1]. In this paper, the error detection method proposed by Alzetta et al. (2018) is incrementally extended to deal with other corpus sections from other domains and registers: this can be seen as a first step of an iterative enhancement approach, which represents one of the currently explored lines of research.

Alzetta et al. (2018) proposed an original error detection and correction method which represents the starting point for the case study reported in this paper. The method, tested against the Italian Universal Dependency Treebank (henceforth IUDT)

[1]http://universaldependencies.org/

Proceedings of the Second Workshop on Universal Dependencies (UDW 2018), pages 1–7
Brussels, Belgium, November 1, 2018. ©2018 Association for Computational Linguistics

(Bosco et al., 2013), mainly targets systematic errors, which represent potentially "dangerous" relations providing systematic but misleading evidence to a parser. Note that with systematic errors we refer here to both real errors as well as annotation inconsistencies internal to the treebank, whose origin can be traced back to different annotation guidelines underlying the source treebanks, or that are connected with substantial changes in the annotation guidelines (e.g. from version 1.4 to 2.0).

This error detection methodology is based on an algorithm, LISCA (*LInguiStically–driven Selection of Correct Arcs*) (Dell'Orletta et al., 2013), originally developed to measure the reliability of automatically produced dependency relations that are ranked from correct to *anomalous* ones, with the latter potentially including incorrect ones. The process is carried out through the following steps:

- LISCA collects statistics about a wide range of linguistic features extracted from a large reference corpus of automatically parsed sentences. These features are both *local*, corresponding to the characteristics of the syntactic arc considered (e.g. the linear distance in terms of tokens between a dependent d and its syntactic head h), and *global*, locating the considered arc within the overall syntactic structure, with respect to both hierarchical structure and linear ordering of words (e.g. the number of "siblings" and "children" nodes of d, recurring respectively to its right or left in the linear order of the sentence; the distance from the root node, the closer and furthest leaf node);

- collected statistics are used to assign a *quality* score to each arc contained in a target corpus (e.g. a treebank). To avoid possible interferences in detecting anomalies which are due to the variety of language taken into account rather than erroneous annotations, both reference and target corpora should belong to the same textual genre or register. On the basis of the assigned score, arcs are ranked by decreasing quality scores;

- the resulting ranking of arcs in the target corpus is partitioned into 10 groups of equivalent size. Starting from the assumption that *anomalous* annotations (i.e. dependencies which together with their context occurrence are deviant from the "linguistic norm" computed by LISCA on the basis of the evidence acquired from the reference corpus) concentrate in the bottom groups of the ranking, the manual search of error patterns is restricted to the last groups. Detected anomalous annotations include both *systematic* and *random* errors. Systematic errors, formalized as *error patterns*, are looked for in the whole target corpus, matching contexts are manually revised and, if needed, corrected.

The methodology was tested against the newspaper section of the Italian Universal Dependency Treebank (henceforth IUDT–news), which is composed by 10,891 sentences, for a total of 154,784 tokens. In this paper, the error detection and correction method depicted above is extended to other sections of the IUDT treebank, containing texts belonging to different genres (namely, legal and encyclopedic texts).

3 Incremental Enhancement of IUDT

The incremental error detection strategy depicted in Section 2 was used to improve IUDT version 2.0 (officially released in March 2017). IUDT 2.0 is the result of an automatic conversion process from the previous version (IUDT 1.4), which was needed because of major changes in the annotation guidelines for specific constructions and new dependencies in the Universal Dependencies (UD) tagset[2]. In spite of the fact that this process was followed by a manual revision targeting specific constructions, the resulting treebank needed a quality check in order to guarantee homogeneity and coherence to the resource: it is a widely acknowledged fact that automatic conversion may cause internal inconsistencies, typically corresponding to systematic errors.

The first step of this revision process is described in Alzetta et al. (2018), which led to IUDT version 2.1, released in November 2017. At this stage, 0.51% dependency relations of IUDT–news were modified (789 arcs): among them, 286 arcs (36.01%) turned out to be random errors, while 503 (63.99%) represent systematic errors.

For the latest published version of IUDT (i.e. 2.2, released in July 2018), error patterns identified in IUDT–news were matched against the other sections of IUDT, which contain legal texts and Wikipedia pages. Although error patterns were acquired from IUDT–news, their occurrence in the

[2]http://universaldependencies.org/v2/summary.html

other two sections of the treebank turned out to be equivalent. In particular, modified arcs corresponding to systematic errors are 0.36% in IUDT–news, 0.34% in IUDT–Wikipedia and 0.35% in IUDT–legal, for a total amount of 1028 deprels, 525 of which were modified in the passage from version 2.0 to version 2.1. This result proves the effectiveness of the methodology: despite of the fact that error patterns were retrieved in a significantly limited search space of the news section of the treebank (covering about 25% of the total number of arcs in IUDT–news), they turned out to be general enough to be valid for the other language registers represented by the other IUDT sub–corpora.

Version 2.2 of IUDT has been further improved: the result is IUDT version 2.3, still unpublished. In this version, residual cases instantiating error patterns were corrected and instances of one of the six error patterns (concerned with nonfinite verbal constructions functioning as nominals) were reported to the original annotation, since we observed that the proposed annotation was no longer convincing on the basis of some of the new instances that were found.

Overall, from IUDT version 2.0 to 2.3, a total of 2,237 dependency relations was modified: 50.91% of them (corresponding to 1,139 arcs) represented systematic errors, while 49.08% (i.e. 1,098 arcs) contained non–pattern errors. Among the latter, 25.77% are random errors (286 arcs), while 74.22% are structural errors (i.e. 815 erroneous non-projective arcs).

4 Experiments

In order to test the impact of the result of our incremental treebank enhancement approach, we compared the dependency parsing results achieved using IUDT versions 2.0 vs 2.3 for training.

4.1 Experimental Setup

Data. Although the overall size of IUDT changed across the 2.0 and 2.3 versions, we used two equivalent training sets of 265,554 tokens to train the parsers, containing exactly the same texts but different annotations. For both sets of experiments, parser performances were tested against a dev(elopment) set of 10,490 tokens and a test set of 7,545 tokens, differing again at the annotation level only. **Parsers.** Four different parsers were selected for the experiments, differing at the level of the used parsing algorithm. The configurations of the parsers were kept the same across all experiments.

DeSR MLP is a transition-based parser that uses a Multi-Layer Perceptron (Attardi, 2006; Attardi et al., 2009), selected as representative of transition-based parsers. The best configuration for UD, which uses a rich set of features including third order ones and a graph score, is described in Attardi et al. (2015). We trained it on 300 hidden variables, with a learning rate of 0.01, and early stopping when validation accuracy reaches 99.5%.

TurboParser (Martins et al., 2013) is a graph-based parser that uses third-order feature models and a specialized accelerated dual decomposition algorithm for making non-projective parsing computationally feasible. It was used in configuration "full", enabling all third-order features.

Mate is a graph-based parser that uses passive aggressive perceptron and exploits a rich feature set (Bohnet, 2010). Among the configurable parameters, we set to 25 the numbers of iterations. *Mate* was used in the pure graph version.

UDPipe is a trainable pipeline for tokenization, tagging, lemmatization and dependency parsing (Straka and Straková, 2017). The transition-based parser provided with the pipeline is based on a non-recurrent neural network, with just one hidden layer, with locally normalized scores. We used the parser in the basic configuration provided for the CoNLL 2017 Shared Task on Dependency Parsing.

Evaluation Metrics. The performance of parsers was assessed in terms of the standard evaluation metrics of dependency parsing, i.e. *Labeled Attachment Score* (LAS) and *Unlabeled Attachment Score* (UAS). To assess the impact of the correction of systematic errors, we devised a new metric inspired by the *Content-word Labeled Attachment Score* (CLAS) introduced for the CoNLL 2017 Shared Task (Zeman and al., 2017). Similarly to CLAS, the new metric focuses on a selection of dependencies: whereas CLAS focuses on relations between content words only, our metric is computed by only considering those dependencies directly or indirectly involved in the pattern–based error correction process. Table 2 reports the list of UD dependencies involved in error patterns: it includes both modified and modifying dependencies occurring in the rewriting rules formalizing error patterns. Henceforth, we will refer to this metric

as *Selected Labeled Attachment Score* (SLAS).

4.2 Parsing Results

The experiments were carried out to assess the impact on parsing of the corrections in the IUDT version 2.3 with respect to version 2.0. Table 1 reports the results of the four parsers in terms of LAS, UAS and SLAS achieved against the IUDT *dev* and *test* sets of the corresponding releases (2.0 vs 2.3). It can be noticed that all parsers improve their performance when trained on version 2.3, against both the *test set* and the *dev set*. The only exception is represented by UDPipe for which a slightly LAS decrease is recorded for the *dev set*, i.e. -0.12%; note, however, that for the same *dev set* UAS increases (+0.12%). The average improvement for LAS and UAS measures is higher for the *test set* than for the *dev set*: +0.38% vs +0.17% for LAS, and +0.35% vs +0.23% for UAS. The higher improvement is obtained by UDPipe (+0.91% LAS, +0.69% UAS) on the *test set*.

Besides standard measures such as LAS and UAS, we devised an additional evaluation measure aimed at investigating the impact of the pattern–based error correction, SLAS, described in Section 4.1. As it can be seen in Table 1, for all parsers the gain in terms of SLAS is significantly higher: the average improvement for the *test set* and the *dev set* is +0.57% and +0.47% respectively. It is also interesting to note that the SLAS values for the two data sets are much closer than in the case of LAS and UAS, suggesting that the higher difference recorded for the general LAS and UAS measures possibly originates in other relations types and corrections (we are currently investigating this hypothesis). This result shows that SLAS is able to intercept the higher accuracy in the prediction of dependency types involved in the error patterns.

To better assess the impact of pattern–based error correction we focused on individual dependencies involved in the error patterns, both modified and modifying ones. This analysis is restricted to the output of the MATE parser, for which a lower average SLAS improvement is recorded (0.34). For both *dev* and *test* sets versions 2.0 and 2.3, Table 2 reports, for each relation type, the number of occurrences in the gold dataset (column "gold"), the number of correct predictions by the parser (column "correct") and the number of predicted dependencies, including erroneous ones (column "sys"). For this dependency subset, an overall reduction of the number of errors can be observed for both evaluation sets. The picture is more articulated if we consider individual dependencies. For most of them, both precision and recall increase from version 2.0 to 2.3. There are however few exceptions: e.g. in the 2.3 version, the number of errors is slightly higher for the *aux* relation in both *dev* and *test* datasets (+4 and +1 respectively), or the *acl* relation in the *dev set* (+3).

Table 3 reports, for the same set of relations, the recorded F-measure (F1), accounting for both precision and recall achieved by the MATE parser for individual dependencies: interesting differences can be noted at the level of the distribution of F1 values in column "Diff", where positive values refer to a gain. Out of the 14 selected dependencies, a F1 gain is reported for 10 relations in the *dev set*, and for 8 in the *test set*. Typically, a gain in F1 corresponds to a reduction in the number of errors. Consider, for example, the *cc* dependency involved in a head identification error pattern (`conj head`), where in specific constructions a coordinating conjunction was erroneously headed by the first conjunct (coordination head) rather than by the second one (this follows from a change in the UD guidelines from version 1.4 to 2.0): in this case, F1 increases for both evaluation datasets (+1.55 and +2.77) and errors decrease (-5 and -6). However, it is not always the case that a decrease of the F1 value is accompanied by a higher number of errors for the same relation. Consider, for example, the *acl* relation for which F1 decreases significantly in version 2.3 of both *dev* and *test* datasets (-6.97 and -4.59). The *acl* relation is involved in a labeling error pattern (`acl4amod`), where adjectival modifiers of nouns (*amod*) were originally annotated as clausal modifiers. Whereas in the *dev set 2.3* the F1 value for *acl* decreases and the number of errors increase, in the *test set 2.3* we observe a decrease in F1 (-4.59%) accompanied by a reduction of the number of errors (-1). The latter case combines apparently contrasting facts: note, however, that the loss in F1 is also influenced by the reduction of *acl* occurrences, some of which were transformed into *amod* in version 2.3.

Last but not least, we carried out the same type of evaluation on the subset of sentences in the development dataset which contain at least one instance of the error patterns: we call it *Pattern Corpus*. For this subset the values of LAS, UAS and

	DeSR MLP			MATE			TurboParser			UDPipe		
	LAS	**UAS**	**SLAS**	**LAS**	**UAS**	**SLAS**	**LAS**	**UAS**	**SLAS**	**LAS**	**UAS**	**SLAS**
Dev 2.0	87.89	91.18	81.10	90.73	92.95	85.82	89.83	92.72	84.10	**87.02**	90.14	79.11
Dev 2.3	**87.92**	**91.23**	**81.48**	**90.99**	**93.28**	**86.28**	**90.34**	**93.14**	**84.98**	86.90	**90.26**	**79.25**
Diff.	0.03	0.05	0.38	0.26	0.33	0.46	0.51	0.42	0.88	-0.12	0.12	0.14
Test 2.0	89.00	91.99	82.59	91.13	93.25	86.08	90.39	93.33	84.78	87.21	90.38	79.66
Test 2.3	**89.16**	**92.07**	**83.14**	**91.41**	**93.70**	**86.30**	**90.54**	**93.49**	**85.00**	**88.12**	**91.07**	**80.95**
Diff.	0.16	0.08	0.55	0.28	0.45	0.22	0.15	0.16	0.22	0.91	0.69	1.29

Table 1: Evaluation of the parsers against the IUDT test and development sets version 2.0 and 2.3.

deprel	**IUDT 2.0**						**IUDT 2.3**					
	Development			**Test**			**Development**			**Test**		
	gold	**correct**	**sys**	**gold**	**correct**	**sys**	**gold**	**correct**	**sys**	**gold**	**correct**	**sys**
acl	151	118	146	83	71	86	115	83	114	71	56	70
acl:relcl	137	106	131	100	77	100	137	112	138	101	80	100
amod	637	606	636	455	439	455	667	641	669	460	445	464
aux	218	208	229	172	162	167	217	206	231	172	159	165
aux:pass	78	69	84	69	64	74	79	71	85	69	63	76
cc	325	305	323	217	194	217	326	311	324	217	200	217
ccomp	62	43	61	29	19	32	62	46	63	30	19	27
conj	372	289	403	253	175	251	370	281	394	257	178	252
cop	126	100	117	85	79	87	126	101	113	85	80	89
nmod	977	828	986	710	612	723	976	827	976	705	615	725
obj	412	372	438	275	247	291	413	374	433	275	247	288
obl	678	541	640	523	427	504	681	551	648	523	425	503
obl:agent	43	39	46	39	36	38	43	40	45	39	36	39
xcomp	92	73	84	58	39	47	96	73	86	62	43	53
TOTAL	4308	3697	4324	3068	2641	3072	4308	3717	4319	3066	2646	3068

Table 2: Statistics of individual dependencies involved in an *error pattern* in the test and development sets of IUDT 2.0 and 2.3 (*gold*). *sys* refers to the number of predicted dependencies by the MATE parser and *correct* to the correct predictions.

deprel	**Development**			**Test**		
	F1 2.0	**F1 2.3**	**Diff**	**F1 2.0**	**F1 2.3**	**Diff**
acl	**79.46**	72.49	-6.97	**84.02**	79.43	-4.59
acl:relcl	79.11	**81.45**	2.35	77.00	**79.60**	2.60
amod	95.20	**95.95**	0.75	**96.48**	96.32	-0.16
aux	**93.06**	91.97	-1.10	**95.58**	94.36	-1.22
aux:pass	85.18	**86.58**	1.40	**89.51**	86.89	-2.62
cc	94.14	**95.69**	1.55	89.40	**92.17**	2.77
ccomp	69.92	**73.60**	3.68	62.30	**66.66**	4.37
conj	**74.58**	73.56	-1.02	69.44	**69.94**	0.49
cop	82.31	**84.52**	2.21	91.86	**91.96**	0.10
nmod	84.36	**84.73**	0.37	85.42	**86.01**	0.60
obj	87.53	**88.42**	0.89	87.28	**87.74**	0.46
obl	82.09	**82.92**	0.83	**83.15**	82.84	-0.31
obl:agent	87.64	**90.91**	3.27	**93.51**	92.31	-1.20
xcomp	**82.95**	80.22	-2.74	74.29	**74.78**	0.49

Table 3: F1 scores and differences for a selection of individual dependencies involved in error patterns by the MATE parser trained on IUDT 2.0 and 2.3.

SLAS for the MATE parser are much higher, ranging between 98.17 and 98.93 for the Pattern corpus 2.0, and between 98.58 and 99.38 for the Pattern corpus 2.3. The gain is in line with what reported in Table 1 for MATE, higher for what concerns LAS (+0.36) and UAS (+0.45), and slightly lower for SLAS (+0.41). Trends similar to the full evaluation datasets are reported also for the dependency-based analysis, which shows however higher F1 values.

5 Conclusion

In this paper, the treebank enhancement method proposed by Alzetta et al. (2018) was further extended and the annotation quality of the resulting treebank was assessed in a parsing experiment car-

ried out with IUDT version 2.0 vs 2.3.

Error patterns identified in the news section of the IUDT treebank were looked for in the other IUDT sections, representative of other domains and language registers. Interestingly, however, error patters acquired from IUDT-news turned out to be characterized by a similar distribution across different treebank sections, which demonstrates their generality.

The resulting treebank was used to train and test four different parsers with the final aim of assessing quality and consistency of the annotation. Achieved results are promising: for both evaluation datasets all parsers show a performance increase (with a minor exception only), in terms of the standard LAS and UAS as well as of the more focused SLAS measure. A dependency-based analysis was also carried out for the relations involved in error patterns: for most of them, a more or less significant gain in the F-measure is reported.

Current developments include: i) extension of the incremental treebank enhancement method by iterating the basic steps reported in the paper to identify new error patterns in the other treebank subsections using LISCA; ii) extension of the incremental treebank enhancement method to other UD treebanks for different languages; iii) extension of the treebank enhancement method to identify and correct random errors.

Acknowledgements

We thank the two anonymous reviewers whose comments and suggestions helped us to improve and clarify the submitted version of the paper. The work reported in the paper was partially supported by the 2–year project (2016-2018) *Smart News, Social sensing for breaking news*, funded by Regione Toscana (BANDO FAR-FAS 2014).

References

C. Alzetta, F. Dell'Orletta, S. Montemagni, and G. Venturi. 2018. Dangerous relations in dependency treebanks. In *Proceedings of 16th International Workshop on Treebanks and Linguistic Theories (TLT16)*, pages 201–210, Prague, Czech Republic.

B. R. Ambati, R. Agarwal, M. Gupta, S. Husain, and D. M. Sharma. 2011. Error Detection for Treebank Validation. In *Proceedings of 9th International Workshop on Asian Language Resources (ALR)*.

Giuseppe Attardi. 2006. Experiments with a multilanguage non-projective dependency parser. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 166–170, Stroudsburg, PA, USA. Association for Computational Linguistics.

Giuseppe Attardi, Felice Dell'Orletta, Maria Simi, and Joseph Turian. 2009. Accurate dependency parsing with a stacked multilayer perceptron. In *Proceeding of Evalita 2009*, LNCS. Springer.

Giuseppe Attardi, Simone Saletti, and Maria Simi. 2015. Evolution of italian treebank and dependency parsing towards universal dependencies. In *Proceedings of the Second Italian Conference on Computational Linguistics*, CLIC-it 2015, pages 23–30, Torino, Italy. Accademia University Press/Open Editions.

Bernd Bohnet. 2010. Very high accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 89–97, Stroudsburg, PA, USA. Association for Computational Linguistics.

C. Bosco, S. Montemagni, and M. Simi. 2013. Converting Italian Treebanks: Towards an Italian Stanford Dependency Treebank. In *Proceedings of the ACL Linguistic Annotation Workshop & Interoperability with Discourse*, Sofia, Bulgaria.

A. Boyd, M. Dickinson, and W. D. Meurers. 2008. On Detecting Errors in Dependency Treebanks. *Research on Language & Computation*, 6(2):113–137.

F. Dell'Orletta, G. Venturi, and S. Montemagni. 2013. Linguistically-driven Selection of Correct Arcs for Dependency Parsing. *Computaciòn y Sistemas*, 2:125–136.

M. Dickinson. 2015. Detection of Annotation Errors in Corpora. *Language and Linguistics Compass*, 9(3):119–138.

M. Dickinson and W. D. Meurers. 2003. Detecting Inconsistencies in Treebank. In *Proceedings of the Second Workshop on Treebanks and Linguistic Theories (TLT 2003)*.

M. Dickinson and W. D. Meurers. 2005. Detecting Errors in Discontinuous Structural Annotation. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 322–329.

M. Dickinson and D. Tufis. 2017. Iterative enhancement. In *Handbook of Linguistic Annotation*, pages 257–276. Springer, Berlin, Germany.

M.C. de Marneffe, M. Grioni, J. Kanerva, and F. Ginter. 2017. Assessing the Annotation Consistency of the Universal Dependencies Corpora. In *Proceedings of the 4th International Conference on Dependency Linguistics (Depling 2007)*, pages 108–115, Pisa, Italy.

A. Martins, M. Almeida, and N. A. Smith. 2013. "turning on the turbo: Fast third-order non-projective turbo parsers". In *Annual Meeting of the Association for Computational Linguistics - ACL*, volume -, pages 617–622.

J. Nivre. 2015. Towards a Universal Grammar for Natural Language Processing. In *Computational Linguistics and Intelligent Text Processing - Proceedings of the 16th International Conference, CICLing 2015, Part I*, pages 3–16, Cairo, Egypt.

Milan Straka and Jana Straková. 2017. Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics.

G. Wisniewski. 2018. Errator: a tool to help detect annotation errors in the universal dependencies project. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 4489–4493, Miyazaki, Japan.

D. Zeman and al. 2017. CoNLL 2017 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada.