More Data and New Tools. Advances in Parsing the Index Thomisticus Treebank

Federica Gamba[1], Marco Passarotti[2] and Paolo Ruffolo[2]

[1] *Istituto Universitario di Studi Superiori (IUSS), Palazzo del Broletto, Piazza della Vittoria 15, 27100 Pavia - Italy*
[2] *CIRCSE Research Centre – Università Cattolica del Sacro Cuore, Largo A. Gemelli 1 – 20123 Milan – Italy*

Abstract
This paper investigates the recent advances in parsing the *Index Thomisticus* Treebank, which encompasses Medieval Latin texts by Thomas Aquinas. The research focuses on two types of variables. On the one hand, it examines the impact that a larger dataset has on the results of parsing; on the other hand, performances of new parsers are analysed with respect to less recent tools. Term of comparison to determine the effective parsing advances are the results in parsing the *Index Thomisticus* Treebank described in a previous work. First, the best performing parser among those concerned in that study is tested on a larger dataset than the one originally used. Then, some parser combinations that were developed in the same study are evaluated as well, assessing that more training data result in more accurate performances. Finally, to examine the impact that newly available tools have on parsing results, we train, test, and evaluate two neural parsers chosen among those best performing in the CoNLL 2018 Shared Task. Our experiments reach the highest accuracy rates achieved so far in automatic syntactic parsing of the *Index Thomisticus* Treebank and of Latin overall.

Keywords
dependency parsing, Latin

1. Introduction

Built upon the *Index Thomisticus* corpus, which collects the *opera omnia* of Thomas Aquinas [8], the *Index Thomisticus* Treebank (IT-TB) [20] represents a prime linguistic resource among those currently available for Latin. Developed at the CIRCSE research center in Milan, the IT-TB is the largest Latin treebank among the five currently available for this language (see Section 2). However, despite the good availability of syntactically annotated corpora for Latin, a number of setbacks do emerge when it comes to parsing Latin. First of all, the richly inflected nature of Latin results in a quite high rate of non-projectivity in dependency trees, which arises due to long distance dependencies in languages with flexible word order and tends to impact negatively the accuracy rates of automatic parsing. Nevertheless, the Medieval Latin of Thomas Aquinas' texts appears to be less non-projective than Classical Latin, as outlined by Passarotti and Ruffolo [22], who report that a rate of 3.24% non-projective dependencies in the IT-TB contrasts the 6.65% in the Latin Dependency Treebank, which includes Classical Latin texts. Moreover, the wide diachronic, diatopic and diaphasic variability of Latin affects

CHR 2021: Computational Humanities Research Conference, November 17–19, 2021, Amsterdam, The Netherlands
federica.gamba@iusspavia.it (F. Gamba); marco.passarotti@unicatt.it (M. Passarotti); paolo.ruffolo@posteo.net (P. Ruffolo)
0000-0002-9806-7187 (M. Passarotti); 0000-0002-9120-0846 (P. Ruffolo)

the overall accuracy rates of different parsers, as they tend to provide higher accuracy rates on those texts that resemble the specific textual variety they were trained on (cf. [22]).

This paper describes a study aimed to improve the performances of automatic dependency parsing for the IT-TB in its native annotation scheme, taking as a benchmark the research described by Ponti and Passarotti [23], who, after testing different parsers, individuated DeSR [1] as the best performing one. After building a new feature model for DeSR specifically suited for the IT-TB, Ponti and Passarotti [23] applied a post-processing combination technique and showed that combining parsers using different types of algorithms returned better parsing results than plain DeSR.

Recent years have seen many steps forward for what concerns both the size and the type of the available linguistic resources for Latin, as well as the performances of probabilistic tools for natural language processing (NLP) purposes. As for the IT-TB, the size of the treebank has grown remarkably since the study of Ponti and Passarotti [23], thus making it possible to evaluate what impact a larger training set can have on the performances of probabilistic tools in parsing the IT-TB. As for the NLP tools, across the very last years new techniques and tools have been developed that exploit the ever growing amount of available training data, thus making it possible to prove whether the most recent tools turn out to be more efficient than less recent ones. This paper presents the results obtained by investigating the impact that these two variables, namely a larger set of training data and new NLP tools, have on parsing the IT-TB.

The paper is organised as follows. Section 2 presents an overview of relevant related studies. In Section 3 the data are presented. Section 4 focuses on the re-evaluation of DeSR performances on the new (larger) training set of the IT-TB. Section 5 explores the impact of using more training data on the accuracy rates of two combinations of parsers. Section 6 reports the performances of two neural parsers (namely, TurkuNLP and ICS-PAS). In Section 7, we present the results provided by three combinations of DeSR with the two neural parsers. In Section 8, an in-depth evaluation of the results is performed and discussed. Finally, Section 9 concludes the paper.

2. Related Work

As mentioned, five treebanks are currently available for Latin. Beside the IT-TB, the other Latin treebanks (all dependency-based) are the following: the PROIEL treebank [15], the Latin Dependency Treebank (part of the Ancient Greek and Latin Treebank) [2], the Late Latin Charter Treebank [10] and the UDante treebank [9]. All the Latin treebanks are annotated both according to their native scheme and to the Universal Dependencies one (UD) [19], except for the UDante treebank, which is available only in the UD scheme.

With respect to parsing the IT-TB, the above-mentioned study by Ponti and Passarotti [23], which we here take as a benchmark, is preceded by other relevant works in the field. In 2010 Passarotti and Ruffolo [22] trained and tested a number of probabilistic dependency parsers, by using data from both the IT-TB and the Latin Dependency Treebank (LDT). In the same year, Passarotti and Dell'Orletta [21] employed DeSR to parse the IT-TB. They delineated an ad-hoc configuration of DeSR features so as to adapt the parser to the specific processing of Medieval Latin and improve accuracy rates. They also defined a revision parsing method and combined the outputs of different algorithms.

However, the most recent study on parsing the IT-TB is the one carried out by Ponti and

Table 1
Size of the T2 training and test sets

	Nodes	Sentences
Training	402,554	24,187
Test	44,752	2,644
Total	447,306	26,831

Passarotti [23]. In particular, for what concerns DeSR, the best results are achieved when the tool exploits a multilayer perceptron (MLP) algorithm, a reversed direction of the parsing transition (right-to-left) and a specifically-tuned settings: the best Labeled Attachment Score reported (LAS) is 83.14 and the highest Unlabeled Attachment Score (UAS) is 88.46 [7]. Regarding the best performing combination of parsers, referred to as C4 in [23], the best results are 86.5 in LAS and 90.97 in UAS. The results obtained through combination already represent an improvement with respect to [21], which were the state of the art in parsing Medieval Latin before [23].

Thanks to the availability of the UD treebanks for Latin, the CoNLL Shared Task 2018 on Multilingual Parsing from Raw Text to Universal Dependencies [28] provided results also for Latin. The tool that proved to perform best on Latin is HIT-SCIR [11], which ranked highest among all the participants, both in terms of LAS and UAS, for all the Latin treebanks. In particular, it obtained a 87.08 LAS and a 89.31 UAS on the IT-TB, a 73.61 LAS and a 77.62 UAS on the PROIEL treebank, and a 72.63 LAS and a 80.47 UAS on the Latin Dependency Treebank.[1]

3. Data

The data used in the experiments consist in the latest release of the IT-TB in its native annotation style [3], which resembles that of the analytical layer of the Prague Dependency Treebank for Czech.[2] This version of the treebank features the entire *Summa contra Gentiles* (four books) and some excerpts from *Summa theologiae* and *Scriptum super Sententiis Petri Lombardi* selected as part of the concordances of lemma *forma* 'form'. Such release of the IT-TB makes available more data than the versions used as data sources for previous experiments in dependency parsing for Latin. In particular, with respect to [23] the missing part of the third book and the entire fourth book of *Summa contra Gentiles* are now included in the dataset, corresponding to 11,881 additional sentences and 193,422 additional tokens. For practical reasons, we define T2 the enlarged dataset and T1 the dataset used in [23].

For evaluation purposes, the treebank is split in a training set and a test set with a ratio of about 9:1. Table 1 illustrates the size of the T2 training and test sets resulting from such partition. When required for the training phase, a development set with the same size of the test set is excerpted from the training data.

[1]The LLCT and the UDante treebanks were not used in the the CoNLL Shared Task 2018, since they have been made available in the UD repository since release v2.6 (May 15th 2020) and v2.8 (May 15th 2021) respectively.

[2]https://ufal.mff.cuni.cz/pdt2.0/doc/manuals/en/a-layer/html/index.html.

Table 2
Results on T1 and T2 by different settings of DeSR

	LAS		UAS	
	T1	T2	T1	T2
MLP, l	**82.01**	81.29	**87.35**	86.60
MLP, r	83.14	**83.87**	88.46	**88.72**
SVM, r	83.35	**83.92**	87.25	**88.41**

4. DeSR Evaluation

After subsetting the dataset in training set and test set, we replicate the first part of the experiments performed by Ponti and Passarotti [23]. We evaluate the accuracy rates of the dependency parser DeSR when trained on the new dataset, yet preserving the algorithms and the feature model defined in [23]. In this way, the only variable to be evaluated is the extended size of the training and test sets, as all the others remain the same, thus allowing to assess the impact of a larger amount of training data on the accuracy rates of the parser.

DeSR [1] is a shift-reduce parser which in its basic settings exploits an MLP algorithm and performs a left-to-right transition while parsing. We make use of the same version of DeSR used in [23] (v. 1.4.3).

First, the performance of DeSR with its basic settings is evaluated. Secondly, we reverse the direction of transition from standard left-to-right to right-to-left, keeping the same MLP algorithm. Thirdly, the MLP algorithm is replaced by a support vector machine (SVM) algorithm, while the transition direction is maintained right-to-left, like in [23].

To sum up, three different settings of DeSR are trained and tested:

- MLP algorithm, left-to-right (MLP, l);
- MLP algorithm, right-to-left (MLP, r);
- SVM algorithm, right-to-left (SVM, r).

Table 2 shows the accuracy rates of the different settings of DeSR in term of LAS and UAS. Results obtained by using the enlarged dataset (T2) are placed side by side to those reported by [23] (T1).

What emerges from Table 2 is that using a larger training set tends to improve the accuracy rates obtained by the parser. However, this does not hold true in the case of the left-to-right MLP algorithm, which proves to be the parser that performs worst in [23]. When it comes to the two most efficient and accurate parsers (namely, those with reversed direction of transition), the enlarged dataset improves the accuracy rates of the parser. As far as the right-to-left MLP algorithm is concerned, we observe a +0.73 improvement in terms of LAS (83.14 with T1 vs 83.87 with T2) and a +0.26 in terms of UAS. As for the right-to-left SVM algorithm, when run on T2 it outperforms the same algorithm run on T1 by +0.57 (LAS) and +0.16 (UAS). However, when the deviation consists in few tenths of percent (as in the UAS rates just mentioned) the improvements cannot be considered meaningful.

The results thus obtained will constitute a baseline for the experiments run and presented later on.

Table 3
Results on T1 and T2 by different combinations of parsers

	LAS		UAS	
	T1	T2	T1	T2
C3	86.37	**86.86**	90.91	**91.14**
C4	86.50	**87.37**	90.97	**91.56**

5. DeSR Combination

The following step in replicating the experiment of [23] concerns the combination of different parsers.

After examining the outputs of single parsers, Ponti and Passarotti [23] employed a post-processing technique that combines the outputs of different (types of) parsers. Such technique exploits the benefits of combination, following the assumption that the mutual difference between parsers promises to improve the final accuracy rates. For the purposes of combination, an algorithm based on unweighted voting was used [27]. In our work, we replicate the experiments run on the combinations named respectively C3 and C4 in [23], chosen as those that reach the highest accuracy rates among the ones concerned. Both C3 and C4 combine outputs produced by different settings of DeSR with outputs from other types of parsers, namely:

- MTGB: a graph-based parser from the MATE-tools collection [4], in its latest version (anna-3.61);
- Joint: a shift-reduce parser, part of the MATE-tools collection and developed by Bohnet et al. [5].

The structure of the combinations C3 and C4 is the following:

- C3: DeSR (MLP, r) + DeSR (MLP, l) + Joint + MTGB;
- C4: DeSR (MLP, r) + DeSR (SVM, r) + DeSR (MLP, l) + Joint + MTGB.

The results of the application of the post-processing combination technique on both T1 and T2 are shown in Table 3. Both for C3 and for C4, the larger dataset proves to lead to higher accuracy rates. In particular, the C4 combination results in the highest gap between T1 and T2 both for LAS (T1: 86.50, T2: 87.37, improvement: +0.87) and UAS (T1: 90.97, T2: 91.56, improvement: +0.59). These results will serve as a baseline for the next combination experiments.

6. New Tools

After examining the impact that a larger training/test set has on the accuracy rates of parsing the IT-TB, we focus on the second variable taken into account in our work, namely training and testing NLP tools of different (and more recent) type than those used by Ponti and Passarotti [23]. To determine which parsers to consider, we refer to the CoNLL Shared Task 2018 on Multilingual Parsing from Raw Text to Universal Dependencies [28]. Among the systems that took part in the Shared Task, we select the two (both neural) parsers that ranked highest with respect to Latin, and in particular to the IT-TB:

- TurkuNLP: end-to-end full neural parsing pipeline, developed by Kanerva et al. [16];
- ICS-PAS: a semi-supervised neural system developed in Warsaw by Rybak and Wróblewska [24].

The two neural parsers are run on the same dataset on which DeSR was trained and tested in Sections 3 and 4, in order to evaluate the specific impact that neural parsing has on the accuracy rates.

6.1. TurkuNLP

TurkuNLP [16] is a neural pipeline that performs four tasks: segmentation, morphological tagging, parsing and lemmatisation.

Lemmatisation is carried out thanks to a novel approach that exploits the OpenNMT neural machine toolkit [17]. As for parsing, the tool is based on Stanford's parser by Dozat, Qi, and Manning [13], which ranked highest in the CoNLL Shared Task 2017 on Multilingual Parsing from Raw Text to Universal Dependencies [29]. First, a word encoder embeds tokens by summing together a set of learned token embeddings, pretrained token embeddings, and token embeddings encoded from the sequence of its characters by using unidirectional LSTM. Then, token embeddings are embedded with Part-of-Speech embeddings as well. Afterwards, representations of tokens in context are created, building relations and attachments in dependency trees. See [12] for further details.

We begin by training a new model of TurkuNLP on the IT-TB extended dataset (T2). To this end, we first employ the pre-trained word embeddings for Latin, published by Facebook and developed with the fastText tool [6]. We both test the embeddings trained on Wikipedia[3] [6] and their newer version [14], trained on Wikipedia and Common Crawl.[4] Afterwards, we develop another model on T2 by using our own embeddings for the *Index Thomisticus* (IT), which we create with fastText (default settings) [6] from the *opera omnia* of Thomas Aquinas provided by the IT corpus [8]. We build two kinds of IT embeddings: (1) token-based embeddings (stored in a one-token-per-line format); (2) sentence-based embeddings (stored in a one-sentence-per-line format).

All the trained models are then evaluated with respect to the test set described in Section 3. Table 4 shows the results (LAS, UAS) obtained by TurkuNLP in comparison to the best performing settings of DeSR and to the best performing combination pipeline (C4).

The models that exploit the Facebook embeddings and the IT sentence-based embeddings prove to perform best, with the IT sentence-based embeddings (LAS: 82.7, UAS: 85.9) outperforming their token-based counterpart (LAS: 82.1, UAS: 85.5). However, as it clearly emerges from Table 4, TurkuNLP proves to obtain significantly lower accuracy rates than both DeSR (MLP, r and SVM, r) and C4, especially in terms of UAS.

6.2. ICS-PAS

ICS-PAS [24] is a neural system consisting of a jointly trained tagger, lemmatiser, and dependency parser. A cross-entropy loss function predicts the output dependency tree. To avoid

[3]https://github.com/facebookresearch/fastText/blob/master/docs/pretrained-vectors.md.

[4]https://fasttext.cc/docs/en/crawl-vectors.html.

Table 4
Results on T2 by different trained models of TurkuNLP compared to DeSR and C4

	LAS	UAS
Facebook embeddings	82.6	85.8
Newer embeddings	81.8	85.2
IT token embeddings	82.1	85.5
IT sentence embeddings	82.7	85.9
DeSR (MLP, r)	83.87	88.72
DeSR (SVM, r)	83.92	88.41
C4	87.37	91.56

Table 5
Results on T2 by different trained models of ICS-PAS compared to DeSR and C4

	LAS	UAS
Facebook embeddings	82.0	85.5
Newer embeddings	82.1	85.6
IT token embeddings	81.5	82.2
IT sentence embeddings	82.2	85.7
DeSR (MLP, r)	83.87	88.72
DeSR (SVM, r)	83.92	88.41
C4	87.37	91.56

cycles in predictions, a ‘cycle-penalty’ loss function is used. During both phases of arc prediction and label prediction, head and dependent are represented as vectors. See [24] for further details.

We evaluate ICS-PAS in the same manner as TurkuNLP. We thus begin by training a model on the extended IT-TB dataset (T2), employing the pre-trained fastText word embeddings for Latin [6] and their newer version [14]. Two further models are then trained on T2, by using respectively the token- and sentence-based embeddings of the IT-TB described in Section 6.1.

Table 5 shows the results (LAS, UAS) obtained by testing ICS-PAS with our trained models. ICS-PAS accuracy rates are displayed together with the accuracy rates provided by the best performing settings of DeSR and the best performing combination pipeline.

As illustrated in Table 5, ICS-PAS and TurkuNLP obtain extremely similar accuracy rates, with TurkuNLP slightly outperforming ICS-PAS by some tenths of percent. The best settings of DeSR and the C4 combination still provide better parsing performances.

7. A New Combination

As mentioned in Section 5, mutual difference between parsers can represent a concrete way to improve their performances. The two parsers we selected from the CoNLL Shared Task 2018 [28] differ substantially from DeSR, particularly in their choice to employ embeddings and implement neural systems. Such a sizeable difference between the parsers raises a question about the performances they could reach if combined together. To answer such question, we evaluate three different combinations of DeSR together with TurkuNLP and ICS-PAS, by

Table 6
Results on T2 by new combinations of parsers

	LAS	UAS
CombA	87.65	91.66
CombB	87.72	91.72
CombC	**89.44**	**92.85**
C4	87.37	91.56

applying the same algorithm for unweighted voting used in the experiment described in Section 5:

- CombA: DeSR (MLP, r) + DeSR (SVM, r) + DeSR (MLP, l) + ICS-PAS;
- CombB: DeSR (MLP, r) + DeSR (SVM, r) + DeSR (MLP, l) + Turku NLP;
- CombC: DeSR (MLP, r) + DeSR (SVM, r) + DeSR (MLP, l) + ICS-PAS + TurkuNLP.

As for TurkuNLP and ICS-PAS, we include in the combinations the outputs obtained with the IT sentence-based embeddings, as they proved to achieve better performances than the token-based ones.

Table 6 reports the accuracy rates, in terms of LAS and UAS, provided by the three combinations. Results obtained by C4, the best performing combination among the ones proposed in [23], are displayed as well for comparison purposes.

The results in Table 6 show how the combinations that include DeSR and, respectively, ICS-PAS (CombA) and TurkuNLP (CombB) outperform the performances provided by the two tools alone (see Subsections 6.1 and 6.2). In particular, CombA reaches 87.65 of LAS and 91.66 of UAS, while plain ICS-PAS exploiting IT sentence-based embeddings obtains 82.2 of LAS and 85.7 of UAS. As for CombB, it obtains a LAS of 87.72 and a UAS of 91.72, while plain TurkuNLP, using IT sentence-based embeddings, reaches a LAS of 82.7 and a UAS of 85.9. The gap is very remarkable, being around +5 in terms of LAS and around +6 in terms of UAS. Yet, a further, even more remarkable improvement is provided by CombC, that combines the three different DeSR settings with both ICS-PAS and TurkuNLP. While CombA and CombB achieve accuracy rates similar, although slightly higher, to the ones of C4, CombC outperforms CombA and CombB by almost 2 points in LAS (89.44) and by more than 1 in UAS (92.85). These results outperform of more than 2 points also those provided by the HIT-SCIR parser in the 2018 CoNLL Shared Task, reported here in Section 2 (LAS: 87.08, UAS: 89.31).

8. In-depth Evaluation

Given the remarkable improvement of the quality of parsing obtained by combining different parsers, we examine the specific contribution that they provide in the combination. We present here an in-depth evaluation of the results achieved on the T2 test set, by focusing on a number of relevant dependency relations and examining the parser-specific performances on them.[5]

[5]The in-depth evaluation is performed by using the MaltEval evaluation tool for dependency parsers [18], available at http://www.maltparser.org/malteval.html.

Table 7
LAS evaluation of CombC and its members for a selected set of dependency relations

Deprel	CombC	DeSR (MLP,l)	DeSR (MLP,r)	DeSR (SVM,r)	ICS-PAS	TurkuNLP
Adv	90.0	82.1	85.0	85.2	89.2	**89.8**
Atr	92.0	85.1	86.7	87.8	**91.1**	91.0
Atr_Co	65.0	50.5	52.1	54.4	68.9	**70.9**
Atv	48.0	29.3	39.7	38.9	**52.9**	48.5
AtvV	22.2	11.8	12.5	0.00	**18.2**	15.4
AuxC	90.5	78.5	84.6	82.8	86.1	**87.7**
AuxP	90.1	81.5	85.6	86.5	88.3	**88.8**
AuxZ	87.2	80.6	81.5	83.2	85.9	**86.7**
Coord	85.0	75.0	77.6	73.4	82.5	**82.8**
Obj	89.7	80.2	83.0	84.0	88.1	**88.8**
Obj_Co	67.9	47.0	48.7	53.2	68.1	**71.9**
Pnom	87.6	78.3	81.4	81.3	86.2	**86.6**
Pred	98.9	96.4	**97.2**	96.6	95.1	95.3
Pred_Co	93.2	85.6	85.4	85.4	**89.9**	88.5
Sb	90.0	81.5	83.7	83.7	89.1	**89.9**
Sb_Co	71.4	53.2	57.6	58.7	73.6	**75.1**

As highlighted by the results reported in Table 7, TurkuNLP turns out to perform best on most of the selected relations. Only in few cases (attributes: Atr; verbal attributes: Atv and AtvV; main predicates: Pred; coordinated main predicates: Pred_Co), it is outperformed by other parsers - mostly by ICS-PAS. Such remark does not match with what observed in Subsection 6.1, where TurkuNLP did not obtain high results with respect to the other parsers. A deeper analysis of its performances, though, shows the main reason of such potential discrepancy. In fact, TurkuNLP fails to handle the terminal punctuation of sentences (dependency relation: AuxK). While the parser assigns the correct relation to terminal punctuation, it always fails to select the right head. Specifically, TurkuNLP scores a 0.00 LAS with respect to terminal punctuation, whereas DeSR performs excellently (LAS between 98.6 and 100), regardless of the adopted configuration. ICS-PAS behaves similarly to TurkuNLP, obtaining a 0.8 LAS with respect to terminal punctuation.

Not surprisingly, the main predicates of sentences (Pred) are the most easily recognised relation (also when they appear in coordinated constructions: Pred_Co), as they concern nodes that do not depend on another node, but on the root of the tree (represented by a technical node assigned relation AuxS). Conversely, the treatment of coordinated constructions represents an issue, as usual in dependency parsing. All parsers included in CombC provide quite low accuracy rates for what concerns coordinated dependency relations, namely coordinated attributes (Atr_Co), objects (Obj_Co) and subjects (Sb_Co). Another tricky relation is represented by verbal attributes not participating in verb government (Atv and AtvV), which prove to be difficult for all parsers.

After the main predicates, attributes (Atr) are the second best-handled relation. The LAS for adverbials (Adv), subjects (Sb) and objects (Obj) are still high and very similar to each other (around 90.0 in CombC). Predicate nominals (Pnom), instead, seem to be more difficult to recognise (CombC: 87.6) and their LAS shows remarkable differences between parsers, ranging from 78.3 (DeSR (MLP, l)) to 86.6 (TurkuNLP).

With respect to subordinating conjunctions (AuxC), prepositions (AuxP) and coordinating nodes (Coord), no parser obtains high results, although all words that are assigned these relations belong to closed lexical classes, which should make them easier to spot and parse. The syntactic ambiguity of some of these words could play a role in such trend. Consider, for instance, the following: (a) *cum*, which can be both a subordinating conjunction (AuxC) and a preposition with meaning 'with' (AuxP); (b) *nec*, which can syntactically behave like a coordination meaning 'and not' (Coord) or an emphasising word meaning 'not even' (AuxZ); (c) *et*, which can have the syntactic function of a coordination meaning 'and' (Coord) or of an emphasising word meaning 'also' (AuxZ).

The parser-based in-depth evaluation above shows the added value of combining different tools, as a means to efficiently exploit the parser-specific contribution to achieve a substantial improvement of the accuracy rates. To give an example of the specific contribution provided by the single parsers to their combination, Figure 1 shows the dependency trees produced by five parsers and by their combination for the following sentence taken from the IT-TB: *Ergo licet aliquid de forma subtrahere* 'Therefore, it is permitted to leave something out of the form'.[6]

From the top to the bottom, Figure 1 lists the Gold Standard and the outputs predicted respectively by th CombC combination (as the best performing one: see Table 6) and by the following parsers: DeSR (MLP, l), DeSR (MLP, r), DeSR (SVM, r), ICS-PAS and TurkuNLP.

In Figure 1, correct dependency relations and labels are displayed in green, while in red are the incorrect ones. Given an ordered set of parsing outputs, the combined output is built by selecting the value proposed by the majority of parsers. For instance, ICS-PAS erroneously attaches the token *licet* to *ergo* instead of the root, and assigns the AuxC relation to their dependency. On its turn, TurkuNLP correctly individuates the root of the tree as the head node for *licet*, but fails in labelling the relation (assigning AuxC instead of Pred). However, CombC succeeds in predicting both the arc and the label for the dependency in question, by choosing the output proposed by the majority of the parsers (namely, DeSR in all its three configurations). The same can be observed with respect to the full stop at the end of the sentence. Even though TurkuNLP and ICS-PAS fail to attach it to the correct head (the root of the tree), in the prediction made by CombC the terminal full stop is made dependent on the correct head node and is assigned the right relation (AuxK), thanks to the correct prediction of DeSR.

Moreover, from Figure 1 we can observe how the different types of parsers here concerned (two neural ones vs DeSR) tend to make the same (or similar) mistakes. For instance, terminal punctuation is attached to the wrong head by both the neural parsers (ICS-PAS and TurkuNLP), which also fail in considering *subtrahere* an ellipsis (ExD) and *licet* a subordinating conjunction (AuxC). The same errors are not made by any of the three configurations of DeSR, which correctly analyse both *licet* and the terminal punctuation mark.

9. Conclusion and Future Work

In this paper we presented various experiments on automatic dependency parsing of the *Index Thomisticus* Treebank. We began by replicating some of the experiments described in [23], in order to evaluate how a larger dataset impacts parsing results. To this end, we first tested different algorithms and settings of the parser DeSR (MLP left-to-right, MLP right-to-left, SVM right-to-left). Then, we evaluated two combinations between the outputs of DeSR and

[6] *Scriptum super Sententiis Petri Lombardi*, *Liber* IV, *Distinctio* 3, *Quaestio* 1, *Articulus* 2, *Quaestiuncula* 3. Translation from https://isidore.co/aquinas.

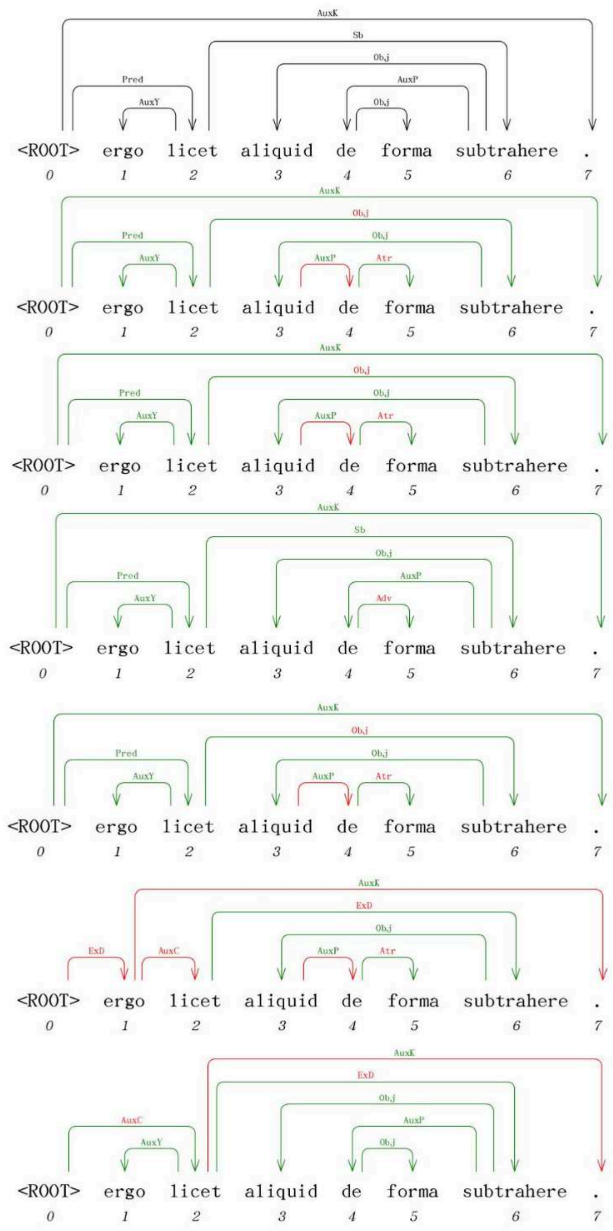
AuxK
Sb
Obj
AuxP
Obj
Pred
AuxY
<ROOT> ergo licet aliquid de forma subtrahere .
0 1 2 3 4 5 6 7
AuxK
Obj
Obj
Pred
AuxY
AuxP
Atr
<ROOT> ergo licet aliquid de forma subtrahere .
0 1 2 3 4 5 6 7
AuxK
Obj
Obj
Pred
AuxY
AuxP
Atr
<ROOT> ergo licet aliquid de forma subtrahere .
0 1 2 3 4 5 6 7
AuxK
Sb
Obj
Pred
AuxP
AuxY
Adv
<ROOT> ergo licet aliquid de forma subtrahere .
0 1 2 3 4 5 6 7
AuxK
Obj
Obj
Pred
AuxY
AuxP
Atr
<ROOT> ergo licet aliquid de forma subtrahere .
0 1 2 3 4 5 6 7
AuxK
ExD
Obj
ExD
AuxC
AuxP
Atr
<ROOT> ergo licet aliquid de forma subtrahere .
0 1 2 3 4 5 6 7
AuxK
ExD
Obj
AuxC
AuxP
AuxY
Obj
<ROOT> ergo licet aliquid de forma subtrahere .
0 1 2 3 4 5 6 7

Figure 1: Gold standard and parsing predictions by five different parsers and their combination for the same sentence from the IT-TB

other parsers (C3, C4). Results show that the larger dataset improves the accuracy rates of parsing with respect to those reported in [23].

We then trained and tested two recently available neural parsers, so as to assess how and if such approach affects the accuracy rates. Although the two selected parsers (ICS-PAS and TurkuNLP) had ranked highest in parsing the IT-TB at the CoNLL Shared Task 2018 on Multilingual Parsing, in our experiment they provided lower accuracy rates than the most accurate DeSR settings (MLP, right-to-left), and substantially lower rates than the C4 combination.

Lastly, we applied a post-processing technique of combination to verify if and to what extent combining together different types of parsers would result in higher accuracy results. The combination that joins the outputs of three DeSR settings, ICS-PAS and TurkuNLP (CombC) resulted in a substantial enhancement of parsing performances, in terms of both LAS (+2.07) and UAS (+1.29).[7]

In the near future, we plan to build and test a new set of sentence-/token-based embeddings for the IT-TB by using specifically defined parameters, instead of the default ones. The experiments on parsing the IT-TB described in this paper are just one piece of the much larger picture of dependency parsing of the Latin language. Such picture features two main, important variables.

First, the high level of diversity of Latin texts, which are spread all over (what today is called) Europe across a period of more than two millennia, heavily affects the diatopic and diachronic portability of trained model of probabilistic NLP tools.

Second, like most of the Latin treebanks, also the IT-TB is available both in its native annotation style and in the UD one, which is nowadays a standard de facto in syntactic (dependency) annotation.

As for the former, although the results of the work presented in this paper are very promising for the specific needs of the IT-TB project, they must be taken carefully when talking about Latin parsing in general terms. Indeed, in the near future it will be necessary to test techniques of domain-adaptation of the available trained models, in order to restrain the decrease of the accuracy rates when models are applied to texts of a different era and/or genre than those of their training set.

As for the latter, there are several initiatives in support of parsing the UD treebanks, like the UDPipe tool[8] [26] and the various shared tasks on UD parsing at international conferences like CoNLL and IWPT.[9]

Finally, in the coming years, one edition of the EvaLatin evaluation campaign of the NLP tools for Latin will include a task specifically devoted to syntactic dependency parsing.[10]

Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme - Grant Agreement No 769994.

References

[1] G. Attardi. "Experiments with a Multilanguage Non-Projective Dependency Parser". In: *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*. 2006, pp. 166–170.

[7]All models, datasets, outputs and scripts that either we used to perform the experiments described in this paper, or that result from them, are openly available at https://github.com/CIRCSE/IT-TB_Parsing.

[8]https://ufal.mff.cuni.cz/udpipe.

[9]See the webpage of the UD-related events at https://universaldependencies.org/events.html. The best performing system on the IT-TB at the CoNLL 2018 Shared Task (HIT-SCIR [11]) provided a LAS of 87.08. The results of the competition are available at http://universaldependencies.org/conll18/results-las.html.

[10]Information on the first edition of EvaLatin, dedicated to lemmatisation and Part-of-Speech tagging, can be found at https://circse.github.io/LT4HALA/EvaLatin. An overview of the results of the evaluation campaign is provided by [25].

[2] D. Bamman and G. Crane. "The Latin Dependency Treebank in a Cultural Heritage Digital Library". In: *Proceedings of the Workshop on Language Technology for Cultural Heritage Data (LaTeCH 2007)*. 2007, pp. 33–40.

[3] D. Bamman, M. Passarotti, R. Busa, and G. Crane. "The Annotation Guidelines of the Latin Dependency Treebank and Index Thomisticus Treebank: the Treatment of some specific Syntactic Constructions in Latin". In: *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*. Marrakech, Morocco: European Language Resources Association (ELRA), 2008.

[4] B. Bohnet. "Very High Accuracy and Fast Dependency Parsing is not a Contradiction". In: *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*. Beijing, China, 2010, pp. 89–97.

[5] B. Bohnet, J. Nivre, I. Boguslavsky, R. Farkas, F. Ginter, and J. Hajič. "Joint Morphological and Syntactic Analysis for Richly Inflected Languages". In: *Transactions of the Association for Computational Linguistics* 1 (2013), pp. 415–428. DOI: 10.1162/tacl_a_00238.

[6] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. "Enriching Word Vectors with Subword Information". In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146.

[7] S. Buchholz and E. Marsi. "CoNLL-X Shared Task on Multilingual Dependency Parsing". In: *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*. New York City, 2006, pp. 149–164.

[8] R. Busa. "Index Thomisticus Sancti Thomae Aquinatis Operum Omnium Indices Et Concordantiae in Quibus Verborum Omnium Et Singulorum Formae Et Lemmata Cum Suis Frequentiis Et Contextibus Variis Modis Referuntur". In: (1974).

[9] F. M. Cecchini, R. Sprugnoli, G. Moretti, and M. Passarotti. "UDante: First Steps Towards the Universal Dependencies Treebank of Dante's Latin Works". In: *Seventh Italian Conference on Computational Linguistics*. CEUR Workshop Proceedings. 2020, pp. 1–7.

[10] F. M. Cecchini, T. Korkiakangas, and M. Passarotti. "A New Latin Treebank for Universal Dependencies: Charters between Ancient Latin and Romance Languages". In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, 2020.

[11] W. Che, Y. Liu, Y. Wang, B. Zheng, and T. Liu. "Towards Better UD Parsing: Deep Contextualized Word Embeddings, Ensemble, and Treebank Concatenation". In: *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Brussels, Belgium, 2018, pp. 55–64.

[12] T. Dozat and C. D. Manning. "Deep Biaffine Attention for Neural Dependency Parsing". In: *arXiv preprint arXiv:1611.01734* (2016).

[13] T. Dozat, P. Qi, and C. D. Manning. "Stanford's Graph-based Neural Dependency Parser at the CoNLL 2017 Shared Task". In: *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Vancouver, Canada, 2017. DOI: 10.18653/v1/K17-3002.

[14] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov. “Learning Word Vectors for 157 Languages”. In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018).* Miyazaki, Japan: European Language Resources Association (ELRA), 2018.

[15] D. T. Haug and M. Jøhndal. “Creating a Parallel Treebank of the Old Indo-European Bible Translations”. In: *Proceedings of the Second Workshop on Language Technology for Cultural Heritage Data (LaTeCH 2008).* 2008, pp. 27–34.

[16] J. Kanerva, F. Ginter, N. Miekka, A. Leino, and T. Salakoski. “Turku Neural Parser Pipeline: An End-to-End System for the CoNLL 2018 Shared Task”. In: *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies.* Brussels, Belgium, 2018, pp. 133–142.

[17] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. Rush. “OpenNMT: Open-Source Toolkit for Neural Machine Translation”. In: *Proceedings of ACL 2017, System Demonstrations.* Vancouver, Canada: Association for Computational Linguistics, 2017, pp. 67–72.

[18] J. Nilsson and J. Nivre. “MaltEval: an Evaluation and Visualization Tool for Dependency Parsing”. In: *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC’08).* Marrakech, Morocco: European Language Resources Association (ELRA), 2008.

[19] J. Nivre, M.-C. de Marneffe, F. Ginter, Y. Goldberg, J. Hajič, C. D. Manning, R. McDonald, S. Petrov, S. Pyysalo, N. Silveira, R. Tsarfaty, and D. Zeman. “Universal Dependencies v1: A Multilingual Treebank Collection”. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16).* Portorož, Slovenia: European Language Resources Association (ELRA), 2016, pp. 1659–1666. URL: https://universaldependencies.org.

[20] M. Passarotti. “The Project of the Index Thomisticus Treebank”. In: *Digital Classical Philology* 10 (2019), pp. 299–320. DOI: 10.1515/9783110599572-017.

[21] M. Passarotti and F. Dell’Orletta. “Improvements in Parsing the Index Thomisticus Treebank. Revision, Combination and a Feature Model for Medieval Latin”. In: *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC’10).* Valletta, Malta: European Language Resources Association (ELRA), 2010.

[22] M. Passarotti and P. Ruffolo. “Parsing the Index Thomisticus Treebank. Some Preliminary Results”. In: *15th International Colloquium on Latin Linguistics.* Innsbrucker Beiträge zur Sprachwissenschaft. 2010, pp. 714–725.

[23] E. M. Ponti and M. Passarotti. “Differentia compositionem facit. A Slower-paced and Reliable Parser for Latin”. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16).* 2016, pp. 683–688.

[24] P. Rybak and A. Wróblewska. “Semi-Supervised Neural System for Tagging, Parsing and Lematization”. In: *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies.* Brussels, Belgium, 2018, pp. 45–54.

[25] R. Sprugnoli, M. Passarotti, F. M. Cecchini, and M. Pellegrini. “Overview of the EvaLatin 2020 Evaluation Campaign”. In: *Proceedings of LT4HALA 2020 - 1st Workshop on Language Technologies for Historical and Ancient Languages.* Marseille, France: European Language Resources Association (ELRA), 2020, pp. 105–110.

[26] M. Straka and J. Straková. “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.* Vancouver, Canada, 2017, pp. 88–99.

[27] M. Surdeanu and C. D. Manning. “Ensemble Models for Dependency Parsing: Cheap and Good?” In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics.* Los Angeles, California, 2010, pp. 649–652.

[28] D. Zeman, J. Hajič, M. Popel, M. Potthast, M. Straka, F. Ginter, J. Nivre, and S. Petrov. “CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies”. In: *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies.* Brussels, Belgium, 2018, pp. 1–21. DOI: 10.18653/v1/K18-2001.

[29] D. Zeman, M. Popel, M. Straka, J. Hajic, J. Nivre, F. Ginter, J. Luotolahti, S. Pyysalo, S. Petrov, M. Potthast, F. Tyers, E. Badmaeva, M. Gokirmak, A. Nedoluzhko, S. Cinkova, J. Hajic jr., J. Hlavacova, V. Kettnerová, Z. Uresova, J. Kanerva, S. Ojala, A. Missilä, C. D. Manning, S. Schuster, S. Reddy, D. Taji, N. Habash, H. Leung, M.-C. de Marneffe, M. Sanguinetti, M. Simi, H. Kanayama, V. dePaiva, K. Droganova, H. Martínez Alonso, Ç. Çöltekin, U. Sulubacak, H. Uszkoreit, V. Macketanz, A. Burchardt, K. Harris, K. Marheinecke, G. Rehm, T. Kayadelen, M. Attia, A. Elkahky, Z. Yu, E. Pitler, S. Lertpradit, M. Mandl, J. Kirchner, H. F. Alcalde, J. Strnadová, E. Banerjee, R. Manurung, A. Stella, A. Shimada, S. Kwak, G. Mendonca, T. Lando, R. Nitisaroj, and J. Li. “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.* Vancouver, Canada, 2017, pp. 1–19.