Proceedings of the Second Workshop on Language Technologies for Historical and Ancient Languages (LT4HALA 2022), pages 20–25
Language Resources and Evaluation Conference (LREC 2022), Marseille, 25 June 2022

Accurate Dependency Parsing and Tagging of Latin

Sebastian Nehrdich[1,2], Oliver Hellwig[1,3]
[1]Institute for Language and Information, Heinrich-Heine-Universität Düsseldorf.
[2]Khyentse Center for Tibetan Buddhist Textual Scholarship, Universität Hamburg.
[3]Department of Comparative Language Science, University of Zürich.
nehrdich@uni-duesseldorf.de, Oliver.Hellwig@uni-duesseldorf.de

Abstract

Having access to high-quality grammatical annotations is important for downstream tasks in NLP as well as for corpus-based research. In this paper, we describe experiments with the Latin BERT word embeddings that were recently be made available by Bamman and Burns (2020). We show that these embeddings produce competitive results in the low-level task of morpho-syntactic tagging. In addition, we describe a graph-based dependency parser that is trained with these embeddings and clearly outperforms various baselines.

Keywords: Latin, biaffine parser, Universal Dependencies, morpho-syntax

1. Introduction

Among the ancient languages in the Universal Dependency (UD) collection of treebanks, Latin has the largest amount of data, and its individual treebanks cover a substantial range of the language including Classical Latin (Crane et al., 2001), Christian authors (Haug and Jøhndal, 2008), a treebank dedicated to the work of Thomas Aquinas (ITTB, Cecchini et al. (2018)) and samples from Late Latin written in the Tuscany (Cecchini et al., 2020). In spite of these resources, large parts of the Latin literature have remained syntactically unanalyzed so far. Developing a reliable morpho-syntactic tagger as well as a syntactic parser for Latin is therefore a desideratum, and several publications have addressed this problem.

While the parser described in the early publication by Koch (1994) works with feature unification, most subsequent models use transition- or graph-based approaches. Bamman and Crane (2008) use the MST parser (McDonald et al., 2005) and obtain labeled attachment scores (LAS) of 54% using gold and 50% using automatically annotated morpho-syntactic information on Perseus data. The authors show that the accuracy is strongly correlated with the amount of non-projective constructions. McGillivray and Passarotti (2009) report experiments with the best parsers available at that time, reaching unlabeled attachment score (UAS) of about 79% and LAS of about 71% on the ITTB. Lee et al. (2011) propose an undirected graphical model that performs joint morpho-syntactic and dependency analysis and that improves over a pipelined approach in the UAS. The authors emphasize the importance of morpho-syntax for successfully parsing morphologically rich languages such as Latin. Ponti and Passarotti (2016) apply a neural parser with feature templates to the ITTB, achieving 90.97% UAS and 86.5% LAS. Slightly better scores are reported by Straka et al. (2019) who use a pipelined model (see Sec. 3 of this paper). Their work will serve as a baseline for model comparison in this paper. Most recently, Gamba et al. (2021) further developed the architecture proposed in Ponti and Passarotti (2016) and achieved 92.85% UAS and 89.44% LAS on the ITTB.

One problem noted by many authors is domain adaptation: Parsers trained on one Latin treebank perform suboptimally when applied to another (see e.g. Passarotti and Ruffolo (2010) and McGillivray and Passarotti (2009), Table 5), a fact that is due to the heterogeneous nature of the corpora and the marked linguistic changes in Christian and medieval Latin (on which see e.g. Dinkova-Bruun (2011) and Vincent (2016)). Another problem is the rich morpho-syntax of Latin and the resulting non-configurationality and freedom of word order, esp. for some classical authors. Andor et al. (2016) have shown that from among the two parsing architectures widely used nowadays, graph-based and transition-based, graph-based parsers are better suited for morphologically rich languages with a high degree of non-projectivity. In this paper, we therefore describe a graph-based parsing architecture that improves over previously reported results by considerable margins. Our architecture is a modified version of the biaffine parser proposed by Dozat and Manning (2017), and uses the contextualized BERT embeddings (Devlin et al., 2019) recently made available for Latin (Latin BERT; Bamman and Burns (2020)). With the help of these contextualized BERT embeddings, our parser is able to outperform the current state of the art by a clear margin. It is especially efficient when no grammatical annotation is available or the training corpus is comparatively small. In addition, we augment the space of the input features by morpho-syntactic information, which further increases the performance.

We make the code of this parser available at: `https://github.com/sebastian-nehrdich/latin-parser`

Section 2 of this paper describes the architectures of two taggers and the dependency parser, and Sec. 3

specifies the experimental settings and discusses the results of our experiments. Section 4 summarizes this paper.

2. Model specification

For morpho-syntactic tagging we use a linear transformation followed by a softmax operation on top of the pre-trained Latin BERT model (Bamman and Burns, 2020), a contextual word embedding model that uses the BERT architecture. It has 12 layers, a hidden dimensionality of 768 and was trained on a total number of 642.7M tokens taken from a large variety of digitized Latin texts ranging from 200 BCE to 1922 CE. We allow all parameters of the model to be fine-tuned during training.

For our dependency parsing experiments we use the biaffine architecture of Dozat and Manning (2017) to which a character based convolutional neural network (CharCNN) was added. This CNN uses the individual characters of each inflected form as input (Rotman and Reichart, 2019; Zhang et al., 2015). Our implementation of the parser is based on the DCST by Rotman and Reichart (2019). However, we decided not to apply the pretraining steps used in the DCST model because a series of experiments (details not reported) shows that these steps do not improve the accuracy of the parser, which is probably due to the comparatively large size of the training corpus and the expressiveness of the input features used here.

The main extension of our parser is that we integrate a contextual word embedding model and a larger number of categorical linguistic input features. In the same way as in the biaffine model, these features are represented as continuous, randomly initialized embeddings. We use embedding dimensions of 100 for all features whose values are set to the gold values provided by the UD data sets. We consider the following input features for the parser:

Morpho-syntax: Case, number and gender of each word, as provided by the UD conllu files. These features are fully specified for nouns, adjectives, non-personal pronouns and verbal nouns with a nominal inflection (e.g. participles of various tenses). Personal pronouns have case and number information. We make use of both atomic features (e.g. 'Acc', 'Sing' and 'Neut' each taken as a separate input feature) and their joint representations (e.g. 'Acc Sing Neut').

Verbal nouns: Verbal nouns convey syntactic information. We therefore evaluate a joint combination of the tense and the type of verbal nouns.

Word representations: We perform experiments with three types of word representations. First we use the character representation of each inflected word form as input for the CharCNN, following Rotman and Reichart (2019). Second we use the fastText Latin model made available by Grave et al. (2018) as static embedding model of complete words. This model has a dimensionality of 300 and has been trained on Latin Common Crawl and Wikipedia data. We decided to use fastText since it was shown in Sprugnoli et al. (2019) that its ability to model model morphology by taking subword units into account is beneficial for synonym-selection tasks. Third we evaluate how the parser performs with Latin BERT as embedding model. In this setting the representation of each inflected form is generated by taking the average of its subword embeddings produced by the Latin BERT model.

3. Experiments

We run experiments on the following tasks: POS tagging, linguistic feature tagging and dependency parsing. For POS tagging, the current state of the art is given in Bamman and Burns (2020). For linguistic feature tagging and dependency parsing it is set by UDPipe 2.0 (Straka et al., 2019). UDPipe 2.0 is a multitask model that jointly predicts POS tags, linguistic features, lemmas and dependency trees. The model is described in detail in Straka (2018). In Straka et al. (2019), UDPipe 2.0 was evaluated with two different settings: One initialized with static word embeddings and one with contextual ones. The contextual word embedding model used by these authors is BERT Multilingual Uncased (Devlin et al., 2019), a model trained on the Wikipedia dumps of the 100 languages with the largest Wikipedias, including Latin.

We use the following three treebanks from the UD framework for all our experiments: The Index Thomisticus Treebank (Cecchini et al. (2018), ITTB), containing works by Thomas Aquinas (390,785 training tokens); the PROIEL treebank (Haug and Jøhndal, 2008), containing both classical and medieval works (172,133 training tokens); and the Perseus Latin Treebank (Bamman and Crane, 2006), containing works from the Classical period (18,184 training tokens). We also create a merged dataset where the training data from all three corpora is joined and duplicates are removed from the training data. We use this merged dataset in all our experiments to evaluate how it affects the respective performance. The following abbreviations are used in the tables reporting the results of our experiments:

UDP2: UDPipe 2.0

Biaffine: the biaffine parser that we adapted for our experiments

WE: static word embeddings (see Straka (2018))

FT: Latin fastText static word embeddings

CLE: character-level embeddings

MBERT: Multilingual Bert Uncased

Feats: morpho-syntactic features; joint representation for UDPipe 2.0, jointly and atomic repr. for biaffine

Merged: Merged training corpora

Model	ITTB	PROIEL	Perseus
UDP2 WE+CLE	96.97	91.53	79.20
UDP2 WE+CLE+MBERT	97.05	91.54	80.43
Latin BERT individual	97.1	94.0	**90.8**
Latin BERT merged	**97.3**	**94.2**	86.7

Table 1: Accuracy of the morpho-syntactic tagger on the UD treebanks for Latin.

3.1. POS Tagging and Morpho-Syntax

Our experiments on POS Tagging mirror the results in Bamman and Burns (2020). We evaluated how merging the training data of the three corpora affects the performance, but could not achieve a consistent performance increase with this method. We report the results for predicting morpho-syntactic features in Tab. 1. The results show that there is a clear increase in accuracy for all three corpora when using the Latin BERT model, while the MBERT model used by UDPipe 2.0 only gives a slight increase in performance (see the first two rows of Tab. 1). With over 10% the increase is most pronounced for the Perseus corpus. We assume that this is due to the comparatively small size of this corpus, a scenario in which pretraining is especially effective. While merging the training data brings a further increase in accuracy in the case of ITTB and PROIEL, this step leads to a clear decrease for Perseus, possibly to be explained by domain effects. Another possible reason could be the different annotation guidelines of these corpora.

3.2. Dependency Parsing

We show the results of the dependency parsing task in Tab. 2. UDPipe 2.0 has been evaluated with static word embeddings (WE) as well as MBERT, adding character level embeddings (CLE) in both cases. The results in the second row of Tab. 2 show that adding the MBERT embedding to UDPipe 2.0 results in slight improvements in UAS and LAS for ITTB, an improvement in LAS for PROIEL and a clear improvement in UAS and LAS for Perseus.

The biaffine model with morpho-syntactic features (Biaffine WE+CLE+POS+Feats) shows a clear improvement over UDPipe 2.0 for all three corpora. Merging the training data of the three corpora (setting Biaffine WE+CLE+POS+Feats+Merged) leads to a lower UAS for ITTB, while for PROIEL it increases UAS and decreases LAS, and for Perseus it clearly improves both UAS and LAS.

The biaffine model based on the Latin BERT without WE/CLE/POS and linguistic features (Biaffine Latin BERT) produces a higher UAS than Biaffine WE+CLE+POS+Feats on all three corpora. For LAS, the performance only increases in the case of PROIEL. Adding WE, CLE and POS (Biaffine Latin BERT+WE+CLE+POS) to Latin BERT increases the performance of the biaffine parser for all corpora in both UAS and LAS. Finally, the combination of Latin BERT with fastText, CLE, POS and all available linguistic features (Biaffine LatinBERT+FT+CLE+POS+Feats) gives the best performance for all three corpora in terms of both UAS and LAS. Similar to the experiments with UDPipe 2.0 (see above) merging the training corpora does not produce a clear-cut outcome (setting Biaffine Latin BERT+FT+CLE+POS+Feats+Merged). While this strategy does not improve the scores for ITTB and PROIEL, it leads to a notable improvement in the case of Perseus.

These results allow for three major observations. First, Latin BERT is a powerful embedding model that significantly boosts performance when compared with non-contextual embedding models and MBERT. We hypothesize that the nature of the textual data used for pretraining is decisive for the performance of the contextual models. The New Latin material of the Wikipedia used for training the MBERT model covers only a small domain compared to the large amount of data which was used for the training of Latin BERT, and which spans a variety of domains from the classical era to the 21st century. In fact, our results show that even a biaffine parser initialized with Latin BERT without any other linguistic features (Biaffine Latin BERT) is able to outperform the UAS of a non-contextual model with full POS and linguistic feature information. This shows that BERT models, when trained on a sufficient amount of data from appropriate domains, are able to successfully capture syntactic information.

The second important observation is that adding gold annotated POS information, static word embeddings and character level embeddings on top of the Latin BERT model gives the biaffine parser another notable boost in performance. The best scores are reached when morpho-syntactic features are used as well. This leads us to the conclusion that providing the parser with linguistic features clearly improves its performance, as was already observed by Lee et al. (2011), even if these features are added on top of an already expressive contextual embedding model.

Third, merging the training data only leads to a better performance for the relatively small Perseus corpus, while the larger ITTB and PROIEL show a slight but consistent decrease in UAS and LAS. For the ITTB, one possible explanation of this contradictory behavior (more data, but worse performance) resembles the one brought forward for the case of MBERT embeddings above: The additional data mostly come from the Latin literature of the classical period and late Antiquity and may therefore differ from Thomas' Latin in terms of their vocabulary and the degree of configurationality. If this is the case, it can be seen as a warning against a simple "more is better" strategy when augmenting the training set for NLP tasks.

To better understand the differences between static and contextual embeddings, we calculate label-

Model	ITTB		PROIEL		Perseus	
	UAS	LAS	UAS	LAS	UAS	LAS
UDP2 WE+CLE	91.06	88.8	83.34	78.66	71.20	61.28
UDP2 WE+CLE+MBERT	91.25	89.10	83.34	78.70	74.39	64.68
Gamba et al. (2021)	92.85	89.44				
Biaffine WE+CLE+POS+Feats	92.74	91.52	84.66	81.58	75.43	69.48
Biaffine WE+CLE+POS+Feats+Merged	92.58	91.52	85.73	80.39	81.28	75.73
Biaffine Latin BERT	92.84	90.91	87.81	83.98	81.54	73.33
Biaffine Latin BERT+WE+CLE+POS	93.55	92.56	88.82	85,82	82.38	75.42
Biaffine Latin BERT+FT+CLE+POS+Feats	**94.04**	**92.99**	**89.21**	**86.34**	83.57	77.63
Biaffine Latin BERT+FT+CLE+POS+Feats+Merged	93.59	92.47	88.90	86.18	**85.37**	**80.16**

Table 2: Performance of the parser on the different UD treebanks for Latin.

wise accuracy scores for three models from Tab. 2, counting those cases as correct in which the label and the head of a syntactic relation are predicted correctly. We order the labels by label-wise differences between the best (`Biaffine LatinBERT+FT+CLE+POS+Feats`) and the worst of our models (`Biaffine WE+CLE+POS+Feats`). Results for the five labels with the highest and the lowest of these differences are displayed in Fig. 1. Judging from the first four labels in the upper compartment of Fig. 1, the best model performs especially well for complex syntactic structures, whose analysis needs access to sentence-level information. Somehow unexpectedly, all models have problems with coordinating conjunctions (cc) although they belong to a closed class of words; in several cases, this is due to wrong attachment. Cases with low differences between `Biaffine LatinBERT+FT+CLE+POS+Feats` and `Biaffine WE+CLE+POS+Feats` include labels which typically have short dependency lengths as well as the three labels advmod, amod and case. The poor performance that all models show for vocatives may be due to issues in the gold data, as many interjections such as *mehercules* or *heu* are labelled syntactically as vocatives on the dependency level, but as `INTJ` on the POS level.

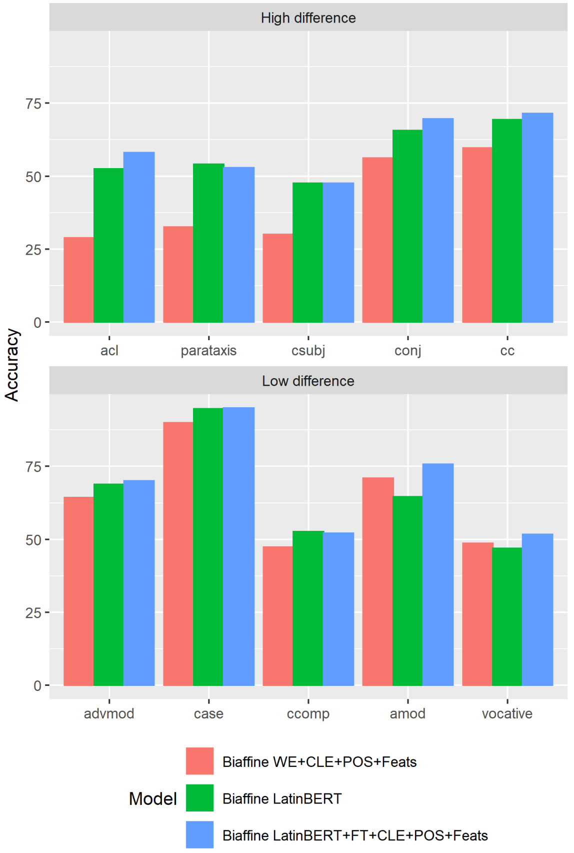

Figure 1: Comparison of label-wise accuracy scores for selected models from Tab. 2. The plot gives the labels with the highest (top) and lowest (bottom) differences between `Biaffine LatinBERT+FT+CLE+POS+Feats` and `Biaffine WE+CLE+POS+Feats`

4. Summary

This paper has shown that even a syntactically challenging language such as Latin can be analyzed with high accuracy scores when appropriate off-the-shelf components are combined in the right way. The decisive element for all three tasks discussed in this paper are contextualized word embeddings, whose application improves scores especially clearly for the small Perseus corpus. Another important result is that adding gold morpho-syntax and static word embeddings further improves the quality of a parser working with contextualized embeddings. Morpho-syntax may seem problematic when it comes to analyzing Latin texts for which this information is not yet available. As, however, the morpho-syntactic and especially the POS tagger come close to human performance for some corpora studied here, one may consider to use a pipelined approach that first runs these taggers on unannotated texts and subsequently applies the dependency parser to the enhanced representations. This is exactly the road we are planning to take when re-analyzing the LatinISE corpus (McGillivray, 2012). We hope that such an enhanced resource can yield better insights in the historical development of the Latin language.

Acknowledgments

Sebastian Nehrdich and Oliver Hellwig were funded by the German Federal Ministry of Education and Research, FKZ 01UG2121, when doing research for this paper.

5. Bibliographical References

Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov, S., and Collins, M. (2016). Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2442–2452, Berlin, Germany, August. Association for Computational Linguistics.

Bamman, D. and Burns, P. J. (2020). Latin BERT: A contextual language model for classical philology. arXiv:2009.10053 [cs.CL].

Bamman, D. and Crane, G. (2006). The design and use of a Latin dependency treebank. pages 67–78.

Bamman, D. and Crane, G. (2008). Building a dynamic lexicon from a digital library. In *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*, pages 11–20.

Cecchini, F. M., Passarotti, M., Marongiu, P., and Zeman, D. (2018). Challenges in converting the *Index Thomisticus* treebank into universal dependencies. Brussels, Belgium.

Cecchini, F. M., Korkiakangas, T., Passarotti, M., et al. (2020). A new Latin treebank for Universal Dependencies: Charters between ancient Latin and Romance languages. In *Proceedings of the LREC*, pages 933–942.

Crane, G., Chavez, R. F., Mahoney, A., Milbank, T. L., Rydberg-Cox, J. A., Smith, D. A., and Wulfman, C. E. (2001). Drudgery and deep thought. *Communications of the ACM*, 44(5):34–40.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, June.

Dinkova-Bruun, G. (2011). Medieval Latin. In James Clackson, editor, *A Companion to the Latin Language*, pages 284–302. Blackwell Publishing.

Dozat, T. and Manning, C. D. (2017). Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations*, pages 1–8.

Gamba, F., Passarotti, M., and Ruffolo, P. (2021). More data and new tools. Advances in parsing the Index Thomisticus Treebank. In *Proceedings of the CHR*, pages 108–122.

Grave, E., Bojanowski, P., Gupta, P., Joulin, A., and Mikolov, T. (2018). Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Haug, D. T. T. and Jøhndal, M. L. (2008). Creating a parallel treebank of the old indo-european bible translations. pages 27–34.

Koch, U. (1994). The enhancement of a dependency parser for Latin. Technical report, Athens, Georgia.

Lee, J. S., Naradowsky, J., and Smith, D. A. (2011). A discriminative model for joint morphological disambiguation and dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 885–894.

McDonald, R., Pereira, F., Ribarov, K., and Hajic, J. (2005). Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the EMNLP*, pages 523–530.

McGillivray, B. and Passarotti, M. (2009). The development of the "Index Thomisticus" treebank valency lexicon. In *Proceedings of the EACL 2009 Workshop on Language Technology and Resources for Cultural Heritage, Social Sciences, Humanities, and Education (LaTeCH–SHELT&R 2009)*, pages 43–50.

McGillivray, B. (2012). LatinISE corpus. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Passarotti, M. C. and Ruffolo, P. (2010). Parsing the Index Thomisticus Treebank. Some preliminary results. In *15th International Colloquium on Latin Linguistics*, pages 714–725. Innsbrucker Beiträge zur Sprachwissenschaft.

Ponti, E. M. and Passarotti, M. (2016). 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)*, pages 683–688.

Rotman, G. and Reichart, R. (2019). Deep contextualized self-training for low resource dependency parsing. *Transactions of the Association for Computational Linguistics*, 7:695–713.

Sprugnoli, R., Passarotti, M., and Moretti, G. (2019). Vir is to moderatus as mulier is to intemperans - lemma embeddings for latin. pages 1–7, 11.

Straka, M., Straková, J., and Hajič, J. (2019). Evaluating contextualized embeddings on 54 languages in POS tagging, lemmatization and dependency parsing. arXiv:1908.07448 [cs.CL].

Straka, M. (2018). UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207, Brussels, Belgium, October. Association for Computational Linguistics.

Vincent, N. (2016). Continuity and change from Latin to Romance. In James Adams et al., editors, *Early* *and Late Latin. Continuity or Change?*, pages 1–13. Cambridge University Press, Cambridge.

Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In C. Cortes, et al., editors, *Advances in Neural Information Processing Systems*, volume 28, pages 649–657. Curran Associates, Inc.