Procesamiento del Lenguaje Natural, Revista nº 74, marzo de 2025, pp. 253-262 recibido 05-12-2024 revisado 21-01-2025 aceptado 07-02-2025

Universal Dependencies annotation of Old English with spaCy and MobileBERT. Evaluation and perspectives

Anotación de Dependencias Universales de inglés antiguo con spaCy y MobileBERT. Evaluación y perspectivas

Javier Martín Arista, Ana Elvira Ojanguren López, Sara Domínguez Barragán
Universidad de La Rioja
{javier.martin, ana-elvira.ojanguren, sara.dominguez}@unirioja.es

Abstract: The aim of this article is to assess three training corpora of Old English and three configurations and training procedures as to the performance of the task of automatic annotation of Universal Dependencies (UD, Nivre et al., 2016). The method is aimed to deciding to what extent the size of the corpus improves results and which configuration turns out the best metrics. The training methods include a pipeline with default configuration, pre-training of tok2vec step and a model of language based on transformers. For all training methods, three training corpora with four different sizes are tested: 1,000, 5,000, 10,000, and 20,000 words. The training and the evaluation corpora are based on ParCorOEv2 (Martín Arista et al., 2021). The results can be summarised as follows. The larger training corpora result in improved performance in all the stages of the pipeline, especially in POS tagging and dependency parsing. Pre-training the tok2vec stage yields better results than the default pipeline. It can be concluded that the performance could improve with more training data or by fine-tuning the models. However, even with the limited training data selected for this study, satisfactory results have been obtained for the task of automatically annotating Old English with UD.

Keywords: Natural Language Processing, Universal Dependencies, syntactic annotation, NPL library, Transformers.

Resumen: El objetivo de este artículo es evaluar tres corpus de entrenamiento de inglés antiguo y tres configuraciones y procedimientos de entrenamiento en relación con el rendimiento en la tarea de anotación automática de Dependencias Universales (UD, Nivre et al., 2016). El método tiene como objetivo determinar en qué medida el tamaño del corpus mejora los resultados y qué configuración ofrece las mejores métricas. Los métodos de entrenamiento incluyen una tubería con configuración predeterminada, pre-entrenamiento del paso tok2vec y un modelo de lenguaje basado en transformadores. Para todos los métodos de entrenamiento, se han probado tres corpus de entrenamiento de cuatro tamaños diferentes: 1.000, 5.000, 10.000 y 20.000 palabras. Los corpus de entrenamiento y evaluación se basan en ParCorOEv2 (Martín Arista et al., 2021). Los resultados se pueden resumir de la siguiente manera: los corpus de entrenamiento más grandes dan lugar a un mejor rendimiento en todas las etapas de la tubería, especialmente en el etiquetado de partes de discurso y el análisis de dependencias. El pre-entrenamiento de la etapa tok2vec produce mejores resultados que la tubería predeterminada. Se puede concluir que el rendimiento podría mejorar con más datos de entrenamiento o con fine tuning de los modelos. Sin embargo, incluso con los datos de entrenamiento limitados seleccionados para este estudio, se han obtenido resultados satisfactorios para la tarea de anotar automáticamente el inglés antiguo con UD.
Palabras clave: Procesamiento de lenguaje natural, dependencias universales, anotación sintáctica, librería de PLN, transformer.

ISSN 1135-5948 DOI 10.26342/2025-74-17

1 Introduction

This article engages in the automatic morpho-syntactic annotation of a historical language with Universal Dependencies.

The Universal Dependencies (UD) framework (de Marneffe et al., 2021) is a cross-linguistic initiative designed to create a standardised, consistent, and universal representation of morpho-syntactic structures across languages. Its primary goal is to enable comparative linguistic analysis and to carry out multilingual natural language processing (NLP) tasks. UD builds on earlier syntactic annotation frameworks, such as the Stanford Dependencies (de Marneffe and Manning, 2008) and other advances in morphological annotation, like Interset (Zeman, 2008). The framework of UD has been applied to language acquisition (MacDonald et al., 2013), comparative linguistics (de Marneffe et al., 2014), NLP tasks (Nivre, 2015), and translation (Nivre, 2016). The 2021 release of the UD dataset contains 183 treebanks over 104 languages (Nivre et al., 2020).

UD annotation comprises UPOS (universal part-of-speech tags), XPOS (language-specific part-of-speech tags), Feats (universal morphological features), lemmas, and dependency heads and labels. These components enable linguists to annotate texts in a way that is both language-specific and cross-linguistically comparable. Morphological information, such as tense, number, and case, is encoded as features, while syntactic information is expressed through labeled dependency relations between words that include the head, the target and the dependency relation itself. Dependency relations are defined by domain, including (i) core arguments and predicate: nominal subject (nsubj), object (obj), indirect object (iobj), clausal subject (csubj), clausal complement (ccomp), open clausal complement (xcomp); (ii) nominal dependents: nominal modifier (nmod), adjectival modifier (amod), numeric modifier (nummod), appositional modifier (appos), determiner (det), classifier (clf); (iii) verb/predicate dependents: auxiliary (aux), copula (cop), marker (mark), adverbial modifier (advmod), adnominal clause (acl), adverbial clause modifier (advcl), compound (compound), fixed multiword expression (fixed); (iv) coordination: conjunct (conj), coordinating conjunction (cc), flat multiword expression (flat); (v) function words: case marking (case), expletive (expl), dislocated elements (dislocated), discourse element (discourse), vocative (vocative), list (list); and (vi) loose/special: parataxis (parataxis), orphan (orphan), goes with (goeswith), reparandum (reparandum), punctuation (punct), root (root), unspecified dependency (dep).

UD adheres to a set of design principles that emphasise universality, transparency, and linguistic typology (Nivre, 2015). The framework focuses on surface-level syntactic structures that can be readily observed and annotated. This makes it particularly useful for low-resource languages, where the vast amounts of training data required by more complex syntactic theories may not be available (Piotrowski, 2012), as is the case with Old English.

Old English is the diachronic stage of the English language spoken in England between approximately the 7th and the 11th centuries (CE). Old English is a language with generalised explicit inflection on nouns, pronouns, adjectives, and verbs, which display a wide range of grammatical distinctions (Campbell 1959). On the syntactic side, Old English shows a relatively free word order compared to Contemporary English, primarily due to its rich inflectional system, which allows grammatical relations to be expressed through morphology rather than positional syntax (Fischer et al. 2000). Nevertheless, the default word order in main clauses was SVO. On the lexical side, the vocabulary of Old English was predominantly Germanic, with a significant portion of its lexical stock inherited from Proto-Germanic, although extensive borrowing from Latin and Old Norse greatly enriched the language (Townend 2002). The written records of Old English comprise around 3,000 texts and a total of 3 million words. The main textual sources of Old English are *The Dictionary of Old English web corpus* (3 million words; Healey et al., 2004) and *The York-Toronto-Helsinki Parsed Corpus of Old English Prose* (1.5 million words; Taylor et al., 2003).

The remainder of the article is structured as follows. Section 2 describes the data and the

computational models selected for the task of annotating Old English text with UD. Section 3 presents the results, including loss values and precision of XPOS assignment, UPOS assignment, FEATS assignment, LEMMA assignment, Unlabeled Attachment Score, Labeled Attachment Score, F-Score and a composite score of all stages. Section 4 discusses the results of this study and compares them with those obtained by Vila and Giarda (2023). Section 5 draws the main conclusions of the article.

2 Description of the task

We have tagged Old English text for lemma and Part-of-Speech (POS) and parsed it for dependency relations. This includes the following columns: UPOS (universal part-of-speech tags), XPOS (language-specific part-of-speech tags), Feats (universal morphological features), lemmas, and dependency heads and labels. The annotation follows the CoNLL-U Plus format, which results from adding the column for word-formation and the column for gloss to the standard CoNLL-U format. Word-formation and gloss have not been annotated at this stage of the research. The tabulation for one sentence can be seen in Figure 1.

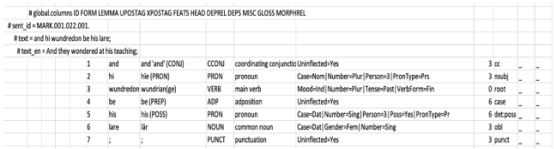

Figure 1. CoNLL-U Plus Format.

Visualisations with UD Annotatrix have been generated, illustrated in Figure 2.

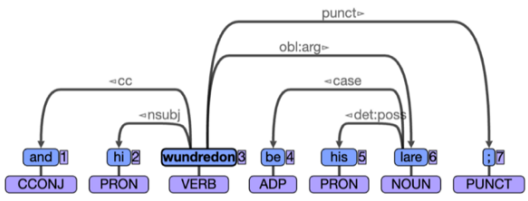

Figure 2. Visualisation with UD Annotatrix.

POS tagging, lemmatisation and dependency parsing of Old English texts have been carried out automatically using spaCy and a pipeline architecture that includes various stages of text processing. The main stages of the pipeline are the following. The tokeniser divides the text into tokens by separating the different words and punctuation marks. The tokeniser is the only rule-based and non-trainable stage. The rules are applied recursively, as presented in Figure 3. The predefined rules for English have been used.

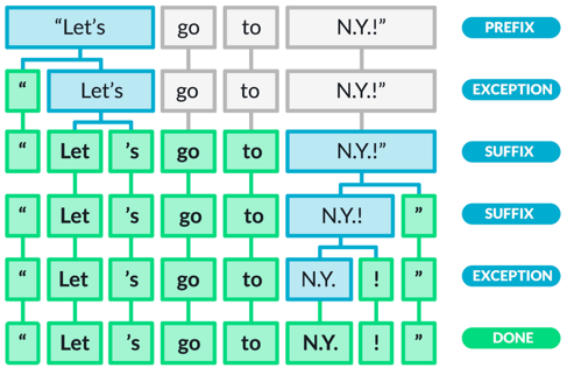

Figure 3. SpaCy tokeniser[1].

Tok2vec / Transformer transforms tokens into vectors that represent proximity between words numerically. Vectors are used in the following stages. Morpho assigns UPOS (universal lexical category) and FEATS (morphological features) to each token. Lemmatiser attributes lemma to each token, while Parser assigns HEAD (head of dependency and DEPREL (dependency relation). The input to the pipeline is plain text, although spaCy stores the data from the different stages. The tokeniser converts the plain text into that structure, while generating the relevant tokens. The second stage, tok2vec (or transformer, depending on the specific case), associates a numerical vector with each token. These vectors are inputted to the next stages. SpaCy can decide whether to share the tok2vec stage or not with the subsequent stages. However, since our training corpus is rather small, this stage has been shared, as it reduces the number of trainable parameters of the system, which ultimately should result in faster training and more accurate results. The output is text parsed with UD morpho-syntactic tagging and annotation. This is illustrated in Figure 4.

[1] https://spacy.io/usage/spacy-101#annotations-token.

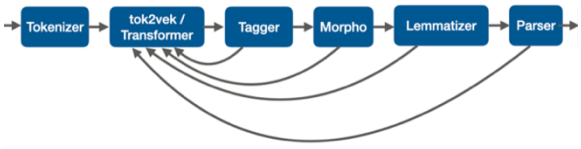

Figure 4. Pipeline architecture.

Basic training corresponds to the standard training in spaCy. It requires a dataset and a tagset. The components of spaCy, such as the tagger, are statistical models based on neural networks. Every decision that these components make, such as assigning a label, is based on predictions based on the model's weight values. These weight values are estimated during the model training process by using examples of text and the corresponding labels as predicted by the model. Training is an iterative process throughout which the model's prediction is compared with the reference labels to estimate the gradient of the loss, which is defined as the difference between the prediction and the correct value. The loss value is used to calculate the weight gradients through backpropagation. These gradients indicate how the weight values should be adjusted so that the predictions of the model converge with the reference labels over time. Training with spaCy can be described as in Figure 5.

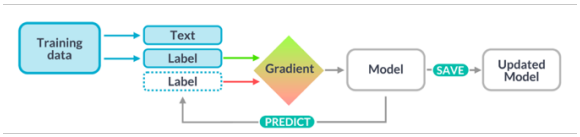

Figure 5. Training with spaCy.[2]

During the training process, spaCy displays a series of metrics calculated with the evaluation set. By default, an evaluation is performed every 200 iterations. Epoch (E) is incremented each time a complete iteration over the training set is performed. Documents (#) are grouped into batches of at least 100 words. Each processed batch increments the iteration number.

For the automatic UD annotation of Old English, a pipeline architecture that includes various stages of text processing has been used. Four different sizes of training corpora have been evaluated by using an independent evaluation subset. Three different configurations and training methods for the system have also been evaluated. The aim of the evaluation was to determine the extent to which increasing the size of the training corpus improved the performance, with a view to opting for the most precise configuration. For each corpus size, process, and processing stage, both training and evaluation metrics have been calculated. For all training methods, three training corpora with four different sizes have been tested: 1,000, 5,000, 10,000, and 20,000 words. The evaluation corpus was a random selection of sentences that has been used for all training subsets, in such a way that the sentences in the evaluation corpus have not included in any of the training subsets. To avoid repetitions, the evaluation corpus has been selected in the first place. This has guaranteed that no sentence appeared in both the training and the evaluation sets.

Loss values have been defined for each stage. Precision has been calculated as follows:

TAG_ACC: Precision of XPOS assignment, tagger stage.

POS_ACC: Precision of UPOS assignment, morphologiser stage.

MORPH_ACC: Precision of FEATS assignment, morphologiser stage.

LEMMA_ACC: Precision of LEMMA assignment, trainable_lemmatiser stage.

DEP_UAS: Unlabeled Attachment Score, percentage of tokens assigned the correct HEAD value, parser stage.

DEP_LAS: Labeled Attachment Score, percentage of tokens assigned both the correct HEAD and DEPREL values, parser stage.

SENTS_F: F-Score of sentence segmentation, parser stage.

SCORE: Composite score of all stages. It is not an average, as each stage has a weighting.

The tok2vec stage starts with randomly initialised weights, so that the vectors assigned to each token are initially random as well. Although these weights are adjusted with each iteration, starting with weights that are closer to being correct will turn out better and faster training results. This stage can be pretrained using unlabeled plain text. Pretraining trains this stage by performing an approximate language modeling task, learning to reflect the semantic distance and co-occurrence of tokens in their associated vectors. The weights of this stage after pretraining can be reused as initial weights during normal training. Since this part can be done with unlabeled text, a larger corpus

[2] https://spacy.io/usage/training.

of Old English was used (3 million words) and trained for 50 iterations. The loss function value for each iteration of pretraining can be seen in Figure 6. A new epoch (x axis) starts each time that the entire corpus has been iterated.

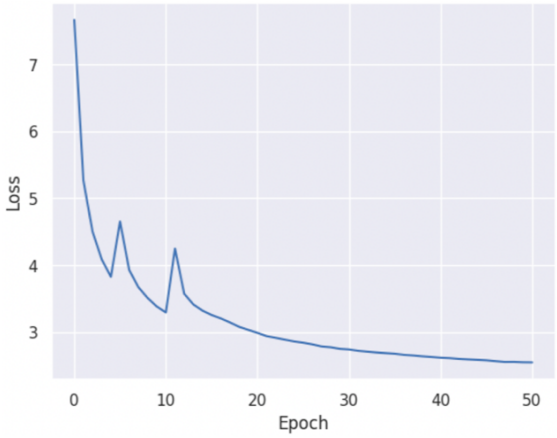

Figure 6. Loss value in pretraining.

The conversion from token to numerical vector can also be done with a language model based on transformers, which replaces tok2vec. In general, one could start with a pre-trained model and fine-tune it for the necessary tasks (tagging, dependencies, etc.) with spaCy. In this case, it is not possible to use a pretrained transformer for modern English because some graphemes Old English (æ, Æ, ð, Ð, þ, Þ) cannot be found in the training of modern English. The model considers the tokens with these graphemes as alien vocabulary and is consequently unable to distinguish them. A new tokenizer has been trained with the plain text corpus (3 million words), with which a new language model has been trained. The architecture of MobileBERT has been selected. MobileBERT is a transformer with a reduced number of parameters (25.3M), which is in keeping with the size of the corpus of written records of Old English (around 17Mb). Language modeling corpora are typically in the order of gigabytes of text, which is usually easier to train than historical languages. Moreover, since we are training models for a new language, we cannot use knowledge transfer techniques from other models to accelerate and improve training.

3 Results

As expected, with a larger training corpus, it takes longer to reach the maximum precision value, and the precision is better. However, the speed at which precision improves decreases rapidly and drastically after a few iterations for any of the tested corpus sizes, instead of gradually decreasing. This indicates that any of the corpus sizes is too small: once the training corpus has been iterated, subsequent iterations repeat the same examples, thus preventing the model from learning new information.

The results for the three models and corpus sizes for each precision metric are presented in figures 7-13.

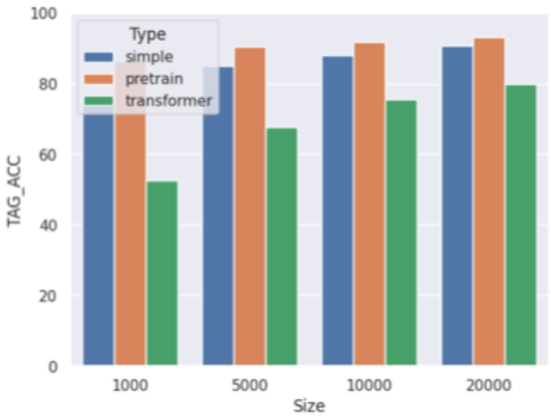

Figure 7. TAG_ACC.

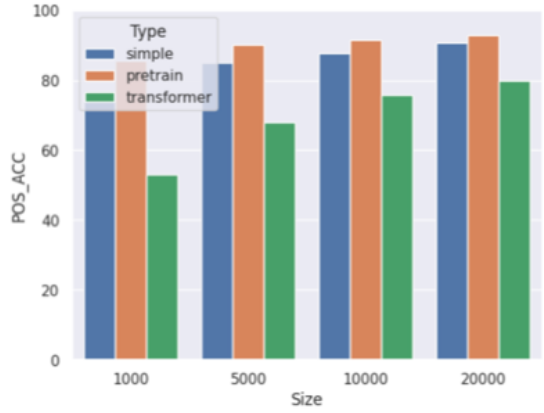

Figure 8. POS_ACC.

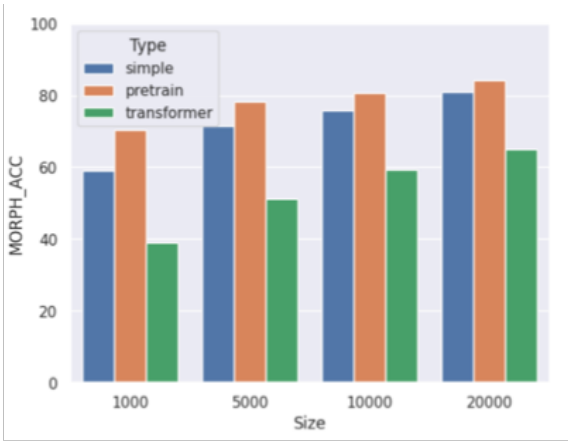

Figure 9. MORPH_ACC.

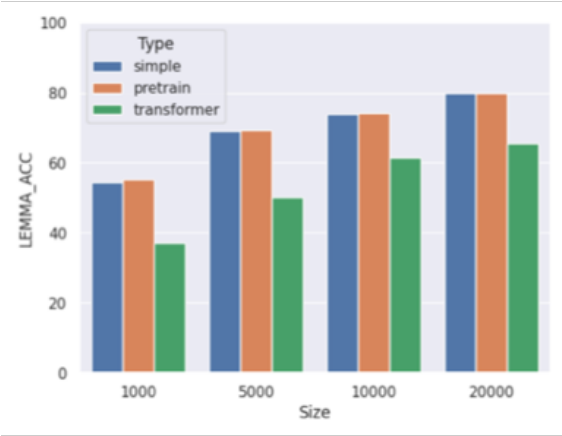

Figure 10. LEMMA_ACC.

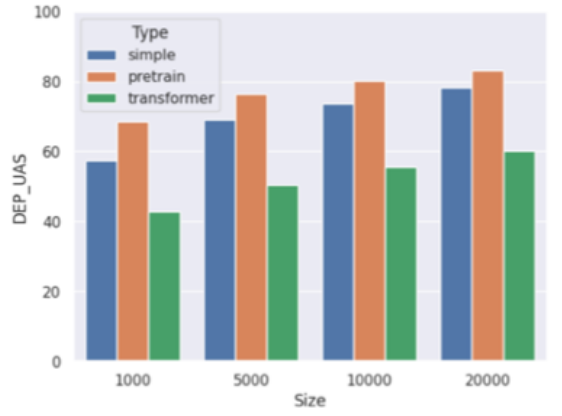

Figure 11. DEP_UAS.

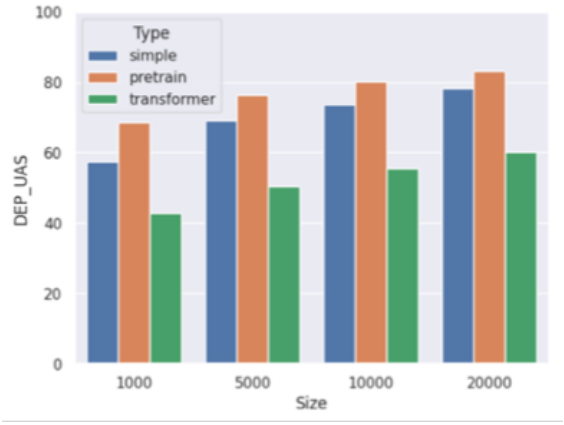

Figure 12. DEP_LAS.

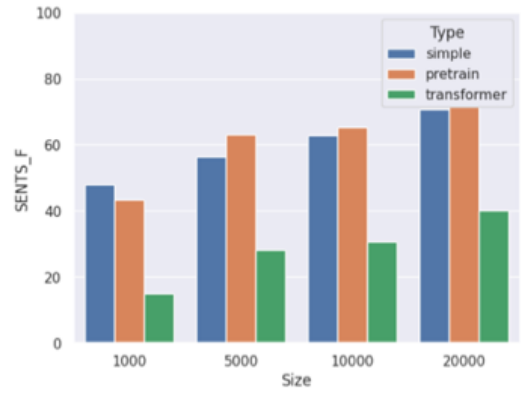

Figure 13. SENTS_F.

The results in figures 7-13 can be interpreted as follows. In these results, it can be observed that as the training size is increased, the performance of all three models improves. It is beyond a doubt that the pretrained model outperforms the simple and transformer models in all cases. The precision rate ranges between 75% and 90%, depending on the evaluation parameter. The lowest precision rates, around 75%, are obtained in tasks related to dependency analysis, although it is surprising to get such low results in the sentence segmentation task (SENTS_F). This could be due to the fact that there are sentences in the training corpus with periods in the middle, punctuation in Old English texts being very far from contemporary conventions. Additionally, we found that the configuration involving pretraining the tok2vec stage yielded better results compared to the default pipeline configuration. Specifically, the POS tagging and dependency parsing stages showed significant improvements with larger training corpora. The lemmatisation stage also benefited

from increased training data but to a lesser extent. The evaluation results confirmed the efficiency of the training process and highlighted the importance of corpus size and training configuration in achieving better performance.

4 Discussion

Villa and Giarda (2023) test the parsing performances of a multilingual parser when applied to Old English data. These authors use different sets of languages to train the models, including individual languages and combinations with the target language (Swedish, Icelandic and German, both alone and combined with Old English). The Old English data chosen by Villa and Giarda (2023) consist of two prose texts, *Adrian and Ritheus* and the first homily of *Ælfric's Supplemental Homilies*. The annotation has been converted from the YCOE morpho-syntactic parsing to the CoNLL-U format of UD, including morphological features, lexical categories and dependency relations. Villa and Giarda (2023) automatically process morphological features and lexical categories, while problematic areas such as lemmatisation and, above all, dependency relations are annotated manually. A total of 292 Old English sentences are adapted to UD requirements, with a total of 5,315 tokens, including training and test data. Overall, all the models trained with data that include the target language data turn out better accuracy rates than their counterparts trained without incorporating the target language data to the training set. The best accuracy rates are obtained by combining data of Old English with German and Icelandic, whose alphabet shares the graphemes <æ> and <ð> with Old English. Figure 14 summarises the main differences between the study conducted by Villa and Giarda (2023) and the present study. Differences arise as to various aspects, including parser type, epoch and word count, model training, maximal accuracy, Labeled Attachment Score (LAS) and Unlabeled Attachment Score (UAS).

	Villa and Giarda (2023)	This study
Parser type	Multilingual	Monolingual
Epoch count	30	50
Word count	≈5,000	25,000 (golden corpus)
Model training	Pre-trained models for Swedish, Icelandic and German, combined with Old English	Model trained specifically for Old English
Maximal accuracy	≈75% (Icelandic, German and Old English)	≈95% (categories) ≈80% (relations) Both in Old English
UAS	≈68% (Icelandic and Old English)	≈82% Old English
LAS	≈59% (Icelandic and Old English)	≈73% Old English

Figure 14: Comparison with Villa and Giarda (2023).

As can be seen in Figure 14, this study throws better precision rates than Villa and Giarda (2023), including categories and functions as well as UAS and LAS. It remains for future research to determine whether the higher precision metrics is attributable to the size of the corpus or to the learning method, as Villa and Giarda combine models pre-trained for modern Germanic languages with Old English and this study has trained the models specifically with Old English data.

Villa and Giarda (2023) discuss the linguistic reasons for the errors made by the best-performing models. These authors anticipate that, given the structural and functional features of Old English, areas of error are likely to be related to the freedom of word order, case syncretism, the co-existence of prepositional and postpositional government, as well as the various types of marking of relative clauses. That said, the most relevant remarks by dependency relation reported by these authors are the following. No relevant issues arise with respect to the dependency relation advmod (adverbial modifier), although Villa and Giarda (2023) find shortcomings related to the words ne 'not' and swa 'so', given that they can perform various functions in the sentence. The same applies to the dependency relation obl (oblique nominal). Villa and Giarda (2023), however, admit that postpositions cannot be automatically annotated, which leads to the misanalysis of obl. As regards the annotation of relative clauses (acl:relcl), Villa and Giarda (2023) identify problems related to the choice of relative pronouns and discontinuity. With respect to the choice of pronoun, all the models correctly annotate the relative clauses with þe 'that' but fail to mark the relative clause if a

different pronoun is used. As for discontinuity, the models misanalyse instances of relative clauses in which the antecedent and the relative clause are separated by constituents of the main clause. The models mark the dependency relation on the adjacent constituent in these cases, even though it is not the noun phrase modified by the relative clause. Villa and Giarda (2023) also find a series of recurrent annotation errors that result from the incorrect interpretation of the POS tagging. This affects the dependency relations cc (coordinating conjunction), advmod:neg (adverbial modifier: negation), aux (auxiliary), advmod:lmod (locative adverbial modifier) and advmod:tmod (temporal adverbial modifier).

This study highlights areas of inadequacy of the automatic parsing of Old English with UD that substantially differ from the points made by Villa and Giarda (2023). At word level, the most problematic phenomenon is negative contractions of verbs, pronouns and adverbs, which are not always identified as such by the model, probably as a result of the wrong assignment of POS tag.

At phrase level, the most challenging areas are noun phrases and prepositional phrases. More specifically, issues arise with respect to flat multiword expressions (comprising proper names, numerals, and titles and honorifics), appositive constructions, coordinating constructions and fixed multiword expressions. Appositions are one of the weak points of the model's performance. No apposition has been analysed correctly. Instead of the apposition (appos) relation, flat (flat multiword expression), dep (unspecified dependency), and list have been incorrectly assigned.

Above the phrase, the identification of the root of a clause or a sentence arises as one of the most complex tasks of annotation. The wrong assignment of the root triggers generalised error in category and function identification. Constructions with incorrect root assignment belong to the following types: (i) copulative, (ii) existential, (iii) conditional sentences in which the main clause is placed in the second part of the construction, (iv) clauses with verbs conjugated for compound tenses, and (v) passives with *weorðan* 'to become'.

At clause level, the most problematic areas include oblique nominals, double object constructions, indirect objects and foreign words. The 'obl' relation is used for nouns, pronouns or noun phrases that function as non-core (oblique) arguments or adjuncts. In an inflectional language like Old English, the oblique status can be the result of either the assignment of an oblique morphological case or of prepositional government. In general terms, the model fails to recognise oblique nominals when they are not governed by a preposition but are marked by morphological case only. As for double object constructions, all of them have been assigned the correct dependency relation. As a general rule, both objects have been considered direct objects by the model. The assignment of the indirect object is slightly more accurate than the one of the double object construction. Foreign words constitute another shortcoming in the model's performance. To begin with, the model does not effectively recognise code-switching because it analyses Latin words as Old English ones. Secondly, the model fails to recognise sequences of foreign words, as it analyses each word independently rather than using a relation specifically designed for sets such as 'list' or 'conj'.

At the level of the complex sentence, the performance of the model seems to have been poorer. This relatively low performance coincides with the presence of some of the most complex syntactic relations found within the UD framework, including orphans, adverbial clause modifiers (advcl), clausal complements (ccomp) and open clausal complements (xcomp). The orphan relation is used in cases of head ellipsis where simple promotion would result in a misleading dependency relation. The most common instance is predicate ellipsis, where one of the core arguments of the verb is promoted to clausal head. The open clausal complement of a verb or an adjective is a predicative or clausal complement without an independent subject. Instead, the reference of the subject is controlled by an argument that is external to the xcomp. Most open clausal complements have been annotated as objects. Figure 15 provides the annotation of *And hu an nunne wearð cuca bebyrged* 'And how a nun was buried alive' [OROS.0.029.002]. This is a passive construction (*wearð bebyrged* 'was buried'), in which *bebyrged* must be tagged as the main verb and, therefore, as the root of the

sentence, while *wearð* must be marked as a passive auxiliary. The adjective *cuca* 'alive' modifies the subject of the sentence (*an nunne* 'a nun') and describes the state throught the process depicted by the lexical verb. As can be seen in the visualisation in Figure 15, *an nunne* and *cuca* agree in case, gender and number (nominative feminine singular). Therefore, *cuca* is a predicative complement and must be annotated with the xcomp label as depending on *bebyrged*, given that predicative complements must always be attached to the main predicate of the sentence.

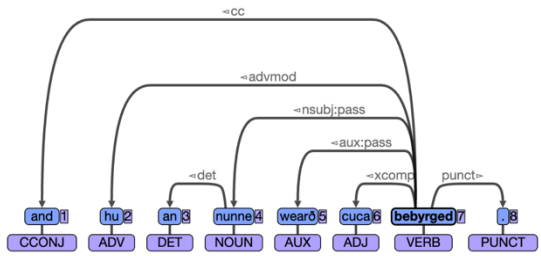

Figure 15: Complex sentence, xcomp.

The model has found the dependency relation that holds between *cuca* and *bebyrged* but has not identified it correctly, because *bebyrged* has been tagged with the relation subject, in spite of being marked as a verb, while *cuca* has been considered a noun and annotated as a passive subject. Other problematic areas at the level of the complex sentence include clausal complements that result from the expression of direct speech and the adverbial clause modifier. In this respect, the model has not been able to annotate correctly adverbial clause modifiers with a non-finite form of the verb (typically, a participle). These instances have been assigned the functions clausal complement or even object.

From a theoretical perspective, the annotation of dependencies that require non-projective arcs represents one of the most challenging areas for computational analysis. According to Decatur (2022: 7), 'an arc is projective if there is a path from the head to every word that lies between the head and the dependent in the sentence'.

5 Conclusion

The results of this task are encouraging because a historical language for which there are no pretrained models, such as Old English, has been automatically annotated for UD. On the other hand, the precision rate, between 75% and 90%, depending on the target, needs improving. Future research will determine which is more efficient: analysing more training corpora or including more unanalysed Old English text. Considering that the written records are limited, further evaluation may be required to decide if the currently available analysed corpus would be sufficient for fine-tuning a transformer model to perform these tasks.

Acknowledgements

PID2023-149762NB-100, funded by MCIN / AEI / 10.13039/501100011033

Bibliographical references

Campbell, A. 1959. *Old English grammar*. Oxford University Press.

Decatur, J. 2022. *A Survey of Non-Projective Dependencies and a Novel Approach to Projectivization for Parsing.* PhD Dissertation. Uppsala Universitet.

de Marneffe, M.-C., and Manning, C. D. 2008. Stanford typed dependencies manual. Stanford University. Revised in 2016.

de Marneffe, M., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J., and Manning, C. 2014. Universal Stanford Dependencies: a cross-linguistic typology. *Proceedings of LREC 2014*, pages 4585-4592.

de Marneffe, M., Manning, C., Nivre, J., and Zeman, D. 2021. Universal Dependencies. *Computational Linguistics,* 47(2): 255-308.

Fischer, O., van Kemenade, A., Koopman, W., and van der Wurff, W. 2000. *The syntax of early English*. Cambridge University Press.

Healey, A. (Ed.), Price Wilkin, J., and Xiang, X. 2004. *The Dictionary of Old English web corpus*. Toronto: Dictionary of Old English Project, Centre for Medieval Studies, University of Toronto.]

Martín Arista, J., Domínguez Barragán, S., García Fernández, L., Ruíz Narbona, E., Torre Alonso, R., and Vea Escarza, R. (comp.). 2021. *ParCorOEv2. An open access annotated parallel corpus Old English-English*. Nerthus Project,

Universidad de La Rioja, www.nerthusproject.com.

McDonald, R., Nivre, J., Quirmbach-Brundage, Y., Goldberg, Y., Das, D., Ganchev, K., Hall, K., Petrov, S., Zhang, H., Täckström, O., Bedini, C., Bertomeu, N. and Lee, J. 2013. Universal Dependency Annotation for Multilingual Parsing. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 92-97.

Nivre, J. 2015. Towards a universal grammar for natural language processing. In A. Gelbukh (Ed.), *Computational Linguistics and Intelligent Text Processing*. New York: Springer, pages 3-16.

Nivre, J. 2016. Universal Dependencies: A Cross-Linguistic Perspective on Grammar and Lexicon. *Proceedings of the Workshop on Grammar and Lexicon: Interactions and Interfaces*, pages 38-40.

Nivre, J., de Marneffe, M., Ginter, F., Goldberg, Y., Hajiˇc, J., Manning, C., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. 2016. Universal Dependencies v1: a multilingual treebank collection. *Proceedings of the 10th International Conference on Language Resources and Evaluation*, pages 1659-1666.

Nivre, J., de Marneffe, M., Ginter, F., Hajiˇc, J., Manning, C., Pyysalo, S., Schuster, S., Tyers, F., and Zeman, D. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. *Proceedings of the Twelfth International Conference on Language Resources and Evaluation (LREC 2020)*, pages 4027-4036.

Taylor, A., Warner, A., Pintzuk, S., and Beths, F. 2003. *The York-Toronto-Helsinki Parsed Corpus of Old English Prose* [https://www-users.york.ac.uk/~lang22/YcoeHome1.htm].

Townend, M. 2002. *Language and history in Viking age England: Linguistic relations between speakers of Old Norse and Old English*. Brepols.

Villa, L. B. and Giarda, M. 2023. Using modern languages to parse ancient ones: a test on Old English. *Proceedings of the 5th Workshop on Research in Computational Linguistic Typology and Multilingual NLP (SIGTYP 2023),* Dubrovnik, Croatia. Association for Computational Linguistics (ACL), pages 30-41.

Zeman, D. 2008. Reusable tagset conversion using tagset drivers. *Proceedings of LREC*, pages 28-30.