Contribution of Linguistic Typology to Universal Dependency Parsing: An Empirical Investigation

Ali Basirat
Center for Language Technology
University of Copenhagen
alib@hum.ku.dk

Navid Baradaran Hemmati
Certified Translation Agency No. 1141
Mashhad, Khorasan, Iran
navidbh@gmail.com

February 21, 2026

Abstract

Universal Dependencies (UD) is a global initiative to create a standard annotation for the dependency syntax of human languages. Addressing its deviation from typological principles, this study presents an empirical investigation of a typologically motivated transformation of UD proposed by William Croft. Our findings underscore the significance of the transformations across diverse languages and highlight their advantages and limitations.

1 Introduction

Universal Dependencies (UD) [Nivre et al.2016, de Marneffe et al.2021] is widely used as a standard for morphosyntactic annotations. Ever since its initial release in October 2014, however, the scheme has been criticized with respect to its adherence to typological principles [Choi et al.2021, Kanayama and Iwamoto2020]. Croft et al. [**?**] cite Nivre [Nivre2015]'s argument that the NLP community has traditionally had little concern for language typology and linguistic universals. They maintain that the UD initiative, akin to prior parsing and tagging scheme proposals aimed at a universal description of the world's languages, fails to refer explicitly to the extensive typological literature on universals, which accounts for the language-specific annotations that it provides besides those that are actually universal in typological terms. Therefore, they continue to propose their own dependency annotation scheme, claiming to represent cross-linguistic variations more comprehensively based on the following four design principles.

The first principle distinguishes universal constructions from language-specific strategies and favors classification based on the former. For example, a copula strategy, used in English to realize a predicate nominal construction, may be represented by a different strategy in another language, so the separate relation in UD for copulas is absent in Croft et al. [Croft et al.2017]'s revision. The second principle emphasizes the use of the same labels for the same functions realized syntactically and morphologically.[1] The third principle prioritizes information packaging over lexical semantics and contributes significantly to the provision of a more economic tag set, as in the substitution of the UD relations for different nominal modifiers with a single label, detailed in Section 3. The fourth principle emphasizes consideration of dependency structure ranks, including predicates, arguments, modifiers, and adverbs qualifying modifiers, representing a range of dependency levels. This can be instantiated by Croft et al. [Croft et al.2017]'s different treatments of complex sentences, complex predicates, and arguments, although they are all dependent on the predicate.

Croft et al. [Croft et al.2017] emphasize that the advantages brought about by their scheme may sacrifice the practical purposes pursued by UD, including achieving high parsing accuracy. This concern has restricted the scheme's application to instructional purposes despite its theoretical potential to address UD's typological gaps. This paper investigates the empirical impact of the scheme on parsing accuracy, aiming to enable its future use in UD revisions.

Our results on a typologically diverse set of languages confirm that it is more straightforward to parse treebanks with typologically informed UD annotation (referred to as TUD henceforth) than to parse ones with standard UD annotation. The results show significant but not necessarily fundamental improvement, as Croft et al. [Croft et al.2017]'s proposals address only the classification of dependency relations without affecting the overall tree structure.

[1] In UD, the `case` label replaces earlier dependency relations for marking prepositional phrases, indicating a syntactic strategy, similar to how it represents a morphological strategy.

2 Related Work

Incorporating knowledge of language diversity into NLP systems is widely regarded as a valuable strategy for enhancing language independence [Bender2009]. In the context of Universal Dependencies, the literature addresses typological limitations through parsing architecture and annotation scheme considerations. Basirat and Nivre [Basirat and Nivre2021] integrate the notion of syntactic nuclei into the UD parsing framework to cope with the typological differences of languages. Their experimentation demonstrates that nucleus composition consistently improves parsing accuracy. This idea is further explored by Nivre et al. [Nivre et al.2022], who find that the observed parsing improvement results from the greater capability of the enriched models of analyzing main predicates, nominal dependents, clausal dependents, and coordination structures.

Other proposals present alternative annotation schemes or revisions to UD. Gerdes et al. [Gerdes et al.2018] propose the Surface-Syntactic Universal Dependencies (SUD), claimed to be a richer and easier variant of UD. They argue that SUD treebanks enable cross-linguistic typological measures thanks to their distributional and functional criteria. Gerdes et al. [Gerdes et al.2019] recall the SUD's general principles, update its relation set, address annotation issues, and present an orthogonal layer of syntactic features. Gerdes et al. [Gerdes et al.2021] further suggest that a new treebank should initially be developed in SUD, even if a UD treebank is intended. The 2021 International Conference on Parsing Technologies [Oepen et al.2021] was dedicated to the additional structural layer of UD, known as Enhanced Universal Dependencies (EUD), to encode grammatical relations that can be represented more adequately using graphical rather than purely rooted trees.

This paper examines a typologically revised annotation scheme for UD, called TUD, based on Croft et al. [Croft et al.2017]'s proposal. Unlike SUD and EUD, which modify dependencies structurally, TUD affects only the dependency labels while preserving the dependency tree topology. Furthermore, it involves less radical dependency relation mappings and retains the majority of original UD labels regardless of the corresponding POS tags.

3 Transformation

We devise a set of transformation rules in the form $r \rightarrow y$ to map a UD relation r to a TUD relation y.

Croft et al. [Croft et al.2017] distinguish the subject relation from object and oblique. They label this relation `sbj` regardless of its categorization as a noun phrase or a clause, in line with their fourth principle. This is realized in our script via the consolidation rules `nsubj` → `sbj` and `csubj` → `sbj`. Furthermore, they find it redundant under the third principle to tag direct and indirect objects differently, so we consider consolidation rules `iobj` → `obj⋆` and `obj` → `obj⋆` to exclude `iobj`. The asterisk indicates that `obj` is already a UD relation, with the latter rule assumed to retain it throughout the conversion.

Croft et al. [Croft et al.2017] challenge the distinction made in UD between complements in terms of grammatical role, including obligatory and nonobligatory control. Our consolidation rules `ccomp` → `comp` and `xcomp` → `comp` serve to neutralize the distinction, conforming to the third principle. Moreover, they point out that UD treats resultatives as controlled complements, which it labels `xcomp`. They suggest that these complex predicate elements be labeled similarly to other secondary predicates and adverbs of manner, which are tagged `sec`. The rule `xcomp` → `sec` is included to realize this, complying with the fourth principle. Thus, the fragmentation rules `xcomp` → `comp` and `xcomp` → `sec` have the same UD relation on their left-hand sides. `xcomp` → `comp` is set to apply where the POS tag of the token with the `xcomp` dependency relation is VERB, which is assumed not to be the case for resultatives, where `xcomp` → `sec` is to apply instead.

UD treebanks optionally set the morphological feature `AdvType` with different values for adverbs of manner, location, time, quantity or degree, cause, and modal nature. On the other hand, Croft et al. [Croft et al.2017] propose in line with their fourth principle that the diversity of adverbs in semantics, syntactic distribution, and morphological form needs to be captured and suggest that adverbs of manner should be labeled `sec`, and ones expressing degree or hedging, aspect or modality, and location or time should be tagged `qlfy`, `aux`, and `obl`, respectively. Therefore, the fragmentation rules `advmod` → `sec` | `qlfy` | `aux⋆` | `obl⋆` are there to convert `advmod` to each of the above relations if `AdvType` is set to the corresponding value. According to the UD documentation, the major values include `Man`, for adverb of manner, `Loc`, for adverb of location, `Tim`, for adverb of time, `Deg`, for adverb of quantity or degree, `Cau`, for adverb of cause, and `Mod`, for adverb of modal nature. Where a different or no setting exists, `advmod` → `obl⋆` will apply by default, as Croft et al. [Croft et al.2017] assert that the UD `advmod` relation should be excluded altogether.

Croft et al. [Croft et al.2017] analyze light verbs as complex predicates, tagged `cxp`, unlike in UD, where they are treated similarly to nominal compounds. Therefore, the rule `compound` → `cxp` is included in our script, in accordance with the fourth principle, to transform the UD `compound` relation to `cxp` where the token's parent is POS-tagged VERB, assumed to signal a light verb construction alongside the token's own `compound` dependency relation label. They also suggest that copulas should be treated as light verbs, hence the consolidation rule `cop` → `cxp` in our script, which conforms to the first principle. Furthermore, they suggest that `nummod`, `amod`, and `det` should all be tagged `mod`, as they involve the same type of information in general, conforming to the third principle. The consolidation rules `nummod` → `mod`, `amod` → `mod`, and `det` → `mod` are there to realize this simplification. Figure 1 summarizes the transformations.

Transformation rules summary:
`nsubj` → `sbj`, `csubj` → `sbj`
`iobj` → `obj*`, `obj` → `obj*`
`ccomp` → `comp`, `xcomp` → `comp` | `sec`
`advmod` → `sec` | `qlfy` | `aux*` | `obl*`
`compound` → `cxp` (when parent is VERB)
`cop` → `cxp`
`nummod` → `mod`, `amod` → `mod`, `det` → `mod`

Figure 1: A summary of the transformation rules.

It should be noted that the eventual aim of this paper is to pave the way for the creation of a totally typologically-based version of UD. The intended scheme will be applicable as a basis for the annotation of text from scratch, involving all the considerations made in Croft et al. [Croft et al.2017]. Since that would be a costly transformation, we need to ensure beforehand that it merits the cost. Therefore, we attempt a preliminary transformation phase, where we apply changes to the available UD treebanks under the limitations imposed by the UD guidelines. In other words, the treebanks resulting from the conversion procedure are intermediary means that enable empirical investigation rather than finalized corpora prepared for use by a corpus linguist. We provide a manual evaluation of the proposed transformation in the next section.

4 Experiments and Results

We evaluate the impact of the typological transformations based on their contribution to parsing performance. Our test benchmark consists of 20 treebanks from UD 2.12 belonging to diverse language families, inspired by Nivre et al. [Nivre et al.2022]. In addition to language diversity, we consider the presence of labels needed for the maximal application of the transformation rules. For this purpose, we incorporate treebanks that include the annotations required for the transformation. As stated in Section 3, for instance, the morphological feature annotation on adverb types, required for our transformation of the `advmod` relation, is optional according to the UD guidelines. Therefore, we add some of the few languages that have included this information in order to cover that specific transformation. Table 1 outlines the selected treebanks with statistics about their sizes and transformed token ratios (Col. IR).

Before proceeding with the parsing analysis, we first present our manual evaluation of the conversion rules in the following section.

4.1 Manual Evaluation

To inspect the performance of the conversion script, we attempt a manual annotation of sample sentences from two of the UD treebanks, where we have mastery over the languages. For that purpose, we randomly select 25 and 50 sentences from the development sets of Persian Seraji and English EWT, containing totals of 599 and 2001 sentences, at intervals of 24 and 40 sentences, respectively. Due to the wider variety of text types on the English side, leading to smaller-sized sentences on average, the two samples end up containing almost as many tokens: 689 and 687, respectively. Then, we manually annotate all the sentences in the two samples based on Croft et al. [Croft et al.2017]'s guidelines and compare the results to the corresponding outputs of the conversion script to spot the mismatches. For each mismatch, it is examined whether the conversion process is responsible. A summary of the manual annotation is provided in Appendix A.

In the case of Persian, a total of 71 tokens are identified, 64 of which represented annotation differences that can be traced back to disagreements between our views and the original UD treebank annotators'. In other words, over 90% of the observed incompatibility would be there also if the original UD scheme were adopted as the basis, and slightly more than the examined tokens are labeled incorrectly due to failure on the part of the conversion script. The two major erroneous cases include one where the `advmod` relation could better be converted to `qlfy` than to `obl` and one where conversion from `compound` to `cxp` is blocked as the conditions set for the application of the relevant rule are not met. Furthermore, there are 5 tokens where conversions from `nmod` or `amod` to `obl` and/or from obl to sec would provide better descriptions, while the required rules are missing due to the absence of clues. These are also considered strictly as cases of script failure.

A few inter-annotator disagreements are also observed for English, which we prefer to ignore as nonnatives. However, the conversion script is responsible for a total of 14 tokens, i.e., slightly more than 2%. Except for one token where the compound relation is incorrectly converted to cxp, they all represent the conversion, by default, of advmod → obl rather than advmod → qlfy (12 instances) or advmod → sec (1 instance).

4.2 Parsing Performance

To address Croft et al. [Croft et al.2017]'s concerns about TUD's practical and theoretical advantage, we base our analysis on the Labeled Attachment Score (LAS), as the typological conversion affects only the dependency labels, and the tree structures remain unchanged. Given that LAS accounts for both dependency labels and structures, it is a more appropriate metric for this analysis. The experiments are based on two primary dependency parsing architectures: transition-based [Nivre2004] and graph-based parsing [McDonald et al.2005]. We use UUParser [de Lhoneux et al.2017] for the former and the Biaffine parser [Dozat and Manning2017] for the latter with the settings outlined in Appendix B. We apply the transformation rules on each treebank and independently train three parsing models, each with distinct random seeds, using both the original (UD) and transformed (TUD) treebanks. The average LASs on the development sets are reported in Cols. UD and TUD. Additionally, Col. Oracle represents the upper bound for parsing performance, achievable if the dependency relations of the transformed tokens are predicted correctly.

It might be argued that any improvement in accuracy resulting from the transformation lies in the simplifying nature of the proposed scheme, which involves plenty of consolidation rules. We maintain that not as much rise in parsing accuracy could be achieved through a random set of merging rules as brought about by our typologically-motivated rules. To demonstrate this, we conduct a randomization experiment, explained in Appendix D with the results reported in the Cols. RND. To assess the significance of the differences between TUD and other baselines, we utilize McNemar's test, as detailed in Appendix C, and mark the significant differences (p-value $<.05$) with an asterisk.

The IR values indicate the importance of the typological transformation, applicable to almost 28% of the tokens, and that, if predicted correctly (Col. Oracle), it can improve the performance by 2.1 and 3.0 points for the transition and graph-based parsing, respectively. However, the parsers can only harness a small but statistically significant portion of this potential improvement, with transition-based achieving 0.21 points and graph-based achieving 0.45 points. Figure **??** visualizes the absolute LAS improvement (or degradation) caused by the typological transformations. We can observe that, on most treebanks, the parsing models result in a better performance on typologically transformed treebanks and that, except for Latin, the negative results are statistically insignificant. These findings highlight the transformation's constructive role in enhancing parsing accuracy without introducing significant adverse effects.

Earlier in this section, we visualized the typological motivation behind the transformation rules, hence their preference over random merging rules. In other words, we raise parsing performance while adhering to well-established typological principles. Following the third principle, for example, we merge all the dependency relations that package the same grammatical information into a single tag, thereby gaining both theoretical and practical benefits. Empirical evidence, summarized in Figure **??**, demonstrates that the third principle is by far the most contributive to the rise in parsing accuracy, while the fourth principle, mainly corresponding to fragmentation rules, is the most detrimental. Moreover, the first principle, represented by only one rule, is rather neutral in this respect, and the second principle is not reflected in the transformations, as UD fully conforms to this principle already. For a detailed discussion of the contribution of the individual transformation rules, see Appendix E.

5 Conclusion

The typological transformation of Universal Dependencies presents an advantage in terms of parsing performance. This benefit is observable across the two primary parsing approaches, namely the transition-based and the graph-based parsing, and in many languages. The positive impact on parsing performance can be attributed to the consolidation rules, which merge the dependency relations with similar typological properties. On the contrary, the parsing performance is slightly hindered by fragmentation rules, indicating their detrimental effect in the context of Universal Dependencies.

Our empirical results demonstrate that an annotation scheme resulting from the typological transformation does not sacrifice the practical aims of UD. Therefore, we suggest establishing such a scheme as an alternative basis for treebanking. Our manual evaluation highlights the importance of typological annotation from scratch or the use of more advanced automatic conversion from the existing UD scheme. In future work, we plan to improve the conversion method and explore the practical benefits of the proposed scheme in downstream tasks.

Acknowledgement

We are grateful to the anonymous reviewers for their invaluable feedback on this work. We also extend our thanks to the Danish research center DeiC for providing computing resources via UCloud, and to the Danish National Life Science Supercomputing Center for granting access to Computerome 2.0 through Project ku-00223.

Limitations

A limitation of this study is that not all of Croft et al. [Croft et al.2017]'s suggested transformation rules are considered due to a lack of annotation in the benchmark. Besides the labels on the right-hand sides of the rules in Section 3, Croft et al. [Croft et al.2017] name two tags for independent elements indicating indexation or agreement and linkers, `idx` and `lnk`. They categorize the above relations as common strategies, implying that they are not regarded as universal constructions. We have decided to ignore the above phenomena at this stage in the absence of clear clues as to how they are marked in each of the treebanks that contain them as independent tokens. We make the same decision for cases where it would be extremely difficult to identify the conditions for applying a rule, as in the case of depictives that are closely similar in structure to adverbial clauses. While these are both marked in UD as `advcl`, Croft et al. [Croft et al.2017] suggest that the former should be labeled `sec`, similarly to resultatives and manner adverbs, transformed via the consolidation rules `xcomp` → `sec` and `advmod` → `sec`, respectively. Our script, however, leaves `advcl` tags unchanged, as one could hardly set proper conditions for an `advcl`-to-`sec` transformation to apply given the clues available on UD treebanks. In addition to these, our benchmark lacks any application for the rules `advmod` → `sec` and `advmod` → `aux*` due to the absence of the optional morphological feature settings on the relevant UD treebanks. Besides, the manual evaluation of the conversion is restricted to two languages due to our limited expertise in other languages.

References

[Basirat and Nivre2021] Ali Basirat and Joakim Nivre. 2021. Syntactic nuclei in dependency parsing—a multilingual exploration. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1376–1387, Online. Association for Computational Linguistics.

[Bender2009] Emily M. Bender. 2009. Linguistically naive != language independent: Why NLP needs linguistic typology. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, pages 26–32, Athens, Greece. Association for Computational Linguistics.

[Choi et al.2021] Hee-Soo Choi, Bruno Guillaume, and Karen Fort. 2021. Corpus-based language universals analysis using Universal Dependencies. In *Proceedings of the Second Workshop on Quantitative Syntax (Quasy, SyntaxFest 2021)*, pages 33–44, Sofia, Bulgaria. Association for Computational Linguistics.

[Croft et al.2017] William Croft, Dawn Nordquist, Michael Regan, and Katherine Looney. 2017. Linguistic typology meets Universal Dependencies. In *Proceedings of the 15th International Workshop on Treebanks and Linguistic Theories (TLT15)*, pages 63–75, Bloomington, IN. CEUR Workshop Proceedings.

[de Lhoneux et al.2017] Miryam de Lhoneux, Yan Shao, Ali Basirat, Eliyahu Kiperwasser, Sara Stymne, Yoav Goldberg, and Joakim Nivre. 2017. From raw text to universal dependencies—look, no tags! In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, Vancouver, Canada.

[de Marneffe et al.2021] Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

[Dozat and Manning2017] Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *International Conference on Learning Representations*.

[Gerdes et al.2018] Kim Gerdes, Bruno Guillaume, Sylvain Kahane, and Guy Perrier. 2018. SUD or surface-syntactic Universal Dependencies: An annotation scheme near-isomorphic to UD. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 66–74, Brussels, Belgium. Association for Computational Linguistics.

[Gerdes et al.2019] Kim Gerdes, Bruno Guillaume, Sylvain Kahane, and Guy Perrier. 2019. Improving surface-syntactic Universal Dependencies (SUD): MWEs and deep syntactic features. In *Proceedings of the 15th International Workshop on Treebanks and Linguistic Theories (TLT15, SyntaxFest 2019)*, pages 126–132, Paris, France. Association for Computational Linguistics.

[Gerdes et al.2021] Kim Gerdes, Bruno Guillaume, Sylvain Kahane, and Guy Perrier. 2021. Starting a new treebank? go SUD! In *Proceedings of the Sixth International Conference on Dependency Linguistics (Depling, SyntaxFest 2021)*, pages 35–46, Sofia, Bulgaria. Association for Computational Linguistics.

[Kanayama and Iwamoto2020] Hiroshi Kanayama and Ran Iwamoto. 2020. How universal are Universal Dependencies? exploiting syntax for multilingual clause-level sentiment detection. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4063–4073, Marseille, France. European Language Resources Association.

[McDonald et al.2005] Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

[Nivre2004] Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57, Barcelona, Spain. Association for Computational Linguistics.

[Nivre2015] Joakim Nivre. 2015. Towards a universal grammar for natural language processing. In *Computational Linguistics and Intelligent Text Processing*, pages 3–16, Cham. Springer International Publishing.

[Nivre et al.2022] Joakim Nivre, Ali Basirat, Luise Dürlich, and Adam Moss. 2022. Nucleus composition in transition-based dependency parsing. *Computational Linguistics*, 48(4):849–886.

[Nivre et al.2016] Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

[Oepen et al.2021] Stephan Oepen, Kenji Sagae, Reut Tsarfaty, Gosse Bouma, Djamé Seddah, and Daniel Zeman, editors. 2021. *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*. Association for Computational Linguistics, Online.

[Tiedemann2015] Jörg Tiedemann. 2015. Cross-lingual dependency parsing with Universal Dependencies and predicted POS labels. In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*, pages 340–349, Uppsala, Sweden. Uppsala University.

A Manual Analysis

Table 2 and Table 3 show the results of the manual analyses made for Persian and English, respectively. They list information on the tokens where the dependency relations assigned via the manual annotation process differ from those set after the conversion. Following the first two columns representing the sentence ID in the original development treebank and the token number in that sentence, respectively, the third to fifth columns indicate the dependency labels set through the original UD annotation, the automatic conversion, and the manual TUD annotation. We include a sixth column, named Our UD to represent the label that we would set based on UD rather than TUD to enable decision-making on the actual source of mismatch. The seventh column, named C, shows if the original UD relation, mentioned in the third column, is among those that (potentially) undergo conversion. The eighth column (HA) indicates whether the manual analysis changes the head as well as the dependency tag of the token. Finally, the ninth column (CR) shows if the conversion process is responsible for the mismatch observed between the contents of Columns four and five. Obvious cases of conversion failure are those where such a mismatch occurs while Columns three and six have the same contents, which means that the script fails to identify the context required for the expected conversion to take place. However, there are additional cases where the contents of the latter pair of columns also mismatch, which means that the script can still be improved to achieve more intelligent identification of more complicated such contexts, or more sophisticated techniques can be applied to further improve the script performance, particularly where this cooccurs with a negative content in Column seven, which means that the original UD label does not actually appear on the left-hand side of any of the current rules.

B Parsing Setup

Our transition-based parsing experiments utilize the implementation from Basirat and Nivre [Basirat and Nivre2021], with the nucleus composition disabled.[2] For the graph-based experiments, we rely on the Biaffine module integrated into the SuPar parser.[3] In both parsers, we refrain from employing pre-trained embeddings, including both static and contextualized models, due to their inconsistent performance across different languages, which could potentially impact the research outcomes. Instead, we opt for a BiLSTM encoder in both scenarios to mitigate external influences and maintain result consistency. Similarly, we avoid using morphosyntactic features such as part-of-speech tags or morphological features due to their varying prediction performance at the test time, which could influence the analysis across languages [Tiedemann2015].

Both parsers are trained for 30 epochs with the word embedding size of 100 and the character embedding dimensions of 100 for UUParser and 50 for SuPar. The UUParser parameters are set to their default values as suggested by Nivre et al. [Nivre et al.2022]. The arc and relation MLP projection sizes of SuPar are set to 500 and 300, respectively, and the other parameters are set to their default values. We disable the projective parsing in both parsers. The computational resource we use to train one transition-based model is a node of three CPUs and 5–10 GB memory in an HPC—however, the graph-based models, each consisting of 12M trainable parameters, are trained on NVIDIA Tesla V100 GPU.

C Hypothesis Testing

We utilize McNemar's test to evaluate the significance of the parsing difference between the two schemes. The test enables us to measure the significance of the changes in parsing performance for each token before and after the typological transformation. More specifically, McNemar's test is a paired-sample t-test for a dichotomous variable that takes two values. In our study, the dichotomous dependent variable of the test indicates whether a token is correctly classified in a scheme or not. The variable takes a value of 1 if the dependency head and label of a token are predicted accurately and a value of 0 otherwise. The categorical independent variable of the test refers to the two dependency schemes, UD and TUD. We collect the value of the dependent variable for all tokens across the two schemes, resulting in two lists of the size of the number of tokens, with the values in each list determining whether the token is classified correctly in the corresponding scheme or not. From these lists, we build a contingency table, shown in Table 4, with the following description:

- A: the number of tokens predicted correctly in both schemes
- B: the number of tokens predicted correctly in UD but incorrectly in TUD
- C: the number of tokens mispredicted in UD but predicted correctly in TUD
- D: the number of mispredicted tokens in both schemes

With this setting, we estimate the p-value to reject the null hypothesis that the typological transformation does not impact parsing accuracy ($p_{UD} = p_{TUD}$). We estimate the p-value based on the binomial distribution. To address the effect of randomness in the parsing models, we collect the statistics from the concatenation of the three runs with different random seeds.

[2]https://github.com/abasirat/uuparser
[3]https://github.com/yzhangcs/parser

D Random Transformation

To ensure that the parsing gain made by the typological transformation is not only due to the consolidation and fragmentation of the rules but also to the linguistic motivations behind them, we design a random transformation setup where the elements of a subset of dependency labels are randomly merged or expanded. To this aim, we search among all possible sets of consolidation and fragmentation rules and select one with an impact rate close to the average impact rate of the typological transformations (28%), explained in Section 4.

In a minimal setup, the number of possible rule sets is proportionate to the number of partitions of the dependency labels set. In this setup, consolidation rules are formed by merging the subsets with at least two labels, and the fragmentation rules can be over some of the singleton subsets. Therefore, the size of the search space with n dependency labels is in the scale of B_n, which is the nth element of the Bell series, and it is approximately 5.3×10^{31} for $n = 37$ UD base dependency labels.

To make the problem more tractable and comparable with the Croft et al. [Croft et al.2017]'s typological transformation rules, we restrict the partitioning to subsets with at most two elements. In this setup, the consolidation rules in each partitioning are formed by merging the elements of subsets that include two elements (i.e., each subset $\{l_i, l_j\}$ of dependency labels introduces two rules $l_i \to l_j$ and $l_j \to l_i$), and the singleton subsets like $\{l_k\}$ either form identity rules with no impact ($l_k \to l_k$) or expand into three sub-labels ($l_k \to l_k^m$, $m = 1, 2, 3$). When expanding, one of the $l_k \to l_k^m$, $m = 1, 2, 3$ rules is randomly applied with a uniform probability. The impact rate of a consolidation rule $l_i \to l_j$ is $\frac{n_i}{N}$, and the impact rate of an expansion rule $l_k \to l_k^m$ is $\frac{2n_k}{3N}$, where n_i is the frequency of occurrence of the label l_i, and N is the total number of tokens in the corpus. The total impact rate of a rule set is then the sum of the impact rates of its rules.

Even with these simplifications, the search space is fairly large, and a complete search requires significant computing resources to find a rule set with a desired impact rate. Therefore, we formulate it as a simulated annealing search that searches for a rule set with a total impact rate of 0.28, an initial temperature of 1.0, and a cooling rate of 0.99. To address the randomness effect, we perform the random transformation three times on each treebank, train a parsing model on the transformed treebanks, and report the average LAS in Table 1, Column RND.

E Rule Contribution

We present some statistics about the distribution of the transformation rules and numerical results of each rule's contribution to the tokens' dependency label prediction. For each rule, we gather all tokens that can undergo the transformation and calculate their LAS (Labeled Attachment Score) both before and after applying the rule. Table 5 shows the absolute improvement or degradation in LAS after applying the transformation rules (Column A), along with the p-values from McNemar's significance test. It also represents the relative contribution of the rules with respect to their distribution, i.e., $A \times P$, where P is the relative frequency of the tokens undergoing each rule.

In summary, the results in Table 5 (Row SUM) show that the transformation rules contribute positively to the prediction of the dependency relations with both the transition-based and the graph-based parsers. Further investigation of the results reveals the varying contribution of the rules to the performance gain. The relative contribution of the rules represented in Column $A \times P$ (and Figure **??**) illustrates the enhancement achieved by each transformation in classifying tokens that undergo the respective transformation. We can see that most rules constructively impact parsing with similar ranks for the two parsers and that untransformed tokens ($r \to x$) are not influenced.

The most significant contribution arises from the consolidation rules. A crucial factor influencing their effectiveness is the inherent difficulty in distinguishing between source relations, often being misclassified as one another in UD, which is no longer an issue once they are merged in TUD. In particular, the effectiveness of the `iobj` $\to$ `obj*` rule is highlighted by the common misclassification scenario, where indirect objects (`iobj`) are mistakenly identified as direct objects (`obj`). Therefore, the unification of `iobj` and `obj` prevents the parser from misclassifying them as each other. We found an analogous explanation for other consolidation rules that unify the clausal complements `ccomp` and `xcomp` into `comp`, combine the subject relations `nsubj` and `csubj` into `sbj`, and merge the determiner `det` with modifiers `amod` and `nummod` into `mod`.

The small improvement made by `cop` $\to$ `cxp` in the transition-based parser can also be attributed to the misclassification of copula as the `compound`, which is unified with copula in the typological scheme.

However, the fragmentation rules such as `xcomp` $\to$ `sec` and `advmod` $\to$ `qlfy` exhibit a negative influence. The detrimental impact of `advmod` $\to$ `qlfy` stems from the frequent mutual misclassification of adverbial and adjectival modifiers in UD, which persists even after typological transformation, manifested as mislabeling qualifying adverbs (`qlfy`) as modifiers (`mod`) in TUD, albeit at a higher rate, which is in turn because `mod` in TUD has a broader scope than `amod` in UD. In addition to the erroneous items present in both schemes, the rule introduces multiple frequent errors in TUD for tokens accurately classified in UD. The top four recurring errors include the misclassification of `qlfy` as `sbj` (13%), `obl*` (72%), `mod` (4%), and `aux*` (4%) for tokens correctly classified in UD as `advmod`. Similarly, the `xcomp` $\rightarrow$ `sec` rule negatively impacts parsing accuracy by misclassifying open clausal complements (`xcomp`) and objects (`obj`) in UD. This misclassification is due to their ambiguities and syntactic similarities, which persist between `sec` and `obj` in TUD, encompassing a large number of tokens, leading to increased errors. Putting it all together, we conclude that the fragmentation rules detract from parsing performance and that their degradation levels are proportional to the scales of their target relations.

Table 1: Average parsing accuracy (LAS) before (UD) and after (TUD) typological transformation.

Language	Treebank	Family	Genus	Size	IR	Transition-based			Graph-b	
						UD	RND	TUD	Oracle	U
TUD	Oracle									
Arabic	padt	Afro-Asiatic	Semitic	254K	20%	78.12	78.10	78.49*	80.22	78.5
78.49	80.22									
Armenian	armt	Indo-European	Indo-Iranian	41K	25%	72.99	74.90	74.81	71.78	66.1
66.86	71.48									
Basque	bdt	Isolate	Basque	91K	27%	75.05	69.90	74.84	77.40	61.54
69.35	71.42									
Chinese	gsd	Sino-Tibetan	Sinitic	111K	24%	70.05	75.51	75.00	77.09	66.77
67.11	69.26									
Classical Chinese	kyoto	Sino-Tibetan	Sinitic	106K	31%	75.33	82.91	81.81	83.85	74.8
74.59	77.09									
English	ewt	Indo-European	Germanic	230K	23%	82.75	78.10	81.60*	83.71	81.60
81.81	83.71									
Finnish	rdt	Uralic	Finno-Ugric	181K	33%	78.15	87.79	72.02*	74.59	72.04
72.02*	74.59									
Hindi	hdtb	Indo-European	Indo-Iranian	316K	29%	87.58*	87.43	89.06*	90.67	89.06
89.06*	90.67									
Italian	isdt	Indo-European	Romance	288K	30%	87.24*	72.88	87.15	88.39	87.1
87.15	88.39									
Korean	kaist	Koreanic	Koreanic	69K	26%	72.53	82.95	67.49	69.98	67.4
67.21	69.98									
Latin	ittb	Indo-European	Italic	421K	28%	83.26*	79.83	85.53	87.13	85.5
85.54	87.13									
Latvian	lvtb	Indo-European	Baltic	253K	34%	79.81	49.01	78.06*	80.59	78.06
78.06*	80.59									
Marathi	ufal	Indo-European	Indo-Iranian	3K	34%	48.71	81.27	48.86	57.31	48.8
49.32	57.31									
Persian	seraji	Indo-European	Indo-Iranian	137K	28%	81.26	64.75*	78.16	78.63	78.1
78.66	78.63									
Russian	taiga	Indo-European	Slavic	187K	24%	64.95*	75.83*	67.64*	67.18	67.64
63.35	67.18									
Swedish	talbanken	Indo-European	Germanic	16K	31%	76.02	54.61*	70.46*	71.05	70.46
70.46*	71.05									
Turkish	imst	Turkic	Altaic	138K	28%	54.74*	75.91*	48.52*	50.32	48.52
48.52*	50.32									
Urdu	udtb	Indo-European	Indo-Iranian	46K	17%	76.19*	48.77	75.92*	76.69	75.92
75.92*	76.69									
Vietnamese	vtb	Austroasiatic	Vietic	34K	28%	48.62*	72.12	47.34	47.12	47.3
47.12	47.10									
Wolof	wtb	Niger-Congo	Atlantic-Congo	34K	28%	72.10	72.42	75.16*	73.93	75.16
75.16*	73.93									
Average				166K	28%	73.29*	73.22*	73.47	75.58	73.4
73.47	75.58									

Table 2: The manual analysis of Persian. C: Convertible; HA: Head Affected; CR: Conversion Responsible.

Sent. ID	Tok.	Orig. UD	Auto. TUD	Man. TUD	Our UD	C	HA	CR
dev-s1	20	obl	obl	obl	obl	no	no	no
dev-s1	24	sbj	sbj	sbj	sbj	no	no	no
dev-s1	29	nmod	nmod	nmod	nmod	no	no	no
dev-s1	30	case	case	case	case	no	no	no
dev-s25	7	compound	cxp	cxp	compound	yes	no	no
dev-s25	12	compound	cxp	cxp	compound	yes	no	no
dev-s25	13	sec	sec	sec	sec	no	no	no
dev-s25	19	aux	aux	aux	aux	no	no	no
dev-s50	6	nmod	nmod	nmod	nmod	no	no	no
dev-s50	7	cxp	cxp	cxp	cxp	no	no	no
dev-s50	11	acl	acl	acl	acl	no	no	no
dev-s75	5	aux	aux	aux	aux	no	no	no
dev-s75	6	cc	cc	cc	cc	no	no	no
dev-s75	7	nmod	nmod	nmod	nmod	no	no	no
dev-s75	15	acl	acl	acl	acl	no	no	no
dev-s75	16	cxp	cxp	cxp	cxp	no	no	no
dev-s75	17	mod	mod	mod	mod	no	no	no
dev-s100	13	obl	obl	obl	obl	no	no	no
dev-s100	26	cxp	cxp	cxp	cxp	no	no	no
dev-s100	32	obl	obl	obl	obl	no	no	no
dev-s100	31	obl	obl	obl	obl	no	no	no
dev-s100	39	conj	conj	conj	conj	no	no	no
dev-s100	42	aux	aux	aux	aux	no	no	no
dev-s150	1	nmod	nmod	nmod	nmod	no	no	no
dev-s200	31	nmod	nmod	nmod	nmod	no	no	no
dev-s275	4	compound	compound	cxp	compound	yes	no	yes
dev-s350	16	obl	obl	obl	obl	no	no	no
dev-s350	21	obl	obl	obl	obl	no	no	no
dev-s350	26	cxp	cxp	cxp	cxp	no	no	no
dev-s350	31	aux	aux	aux	aux	no	no	no
dev-s350	34	obl	obl	obl	obl	no	no	no
dev-s375	7	nmod	nmod	nmod	nmod	no	no	no
dev-s375	29	appos	appos	appos	appos	no	no	no
dev-s400	8	nmod	nmod	nmod	nmod	no	no	no
dev-s400	17	compound	compound	cxp	compound	yes	no	yes
dev-s425	6	sbj	sbj	sbj	sbj	no	no	no
dev-s450	2	nmod	nmod	nmod	nmod	no	no	no
dev-s450	3	nmod	nmod	nmod	nmod	no	no	no
dev-s450	11	sec	sec	sec	sec	no	no	no
dev-s450	12	appos	appos	appos	appos	no	no	no
dev-s450	14	nmod	nmod	nmod	nmod	no	no	no
dev-s450	23	fixed	fixed	fixed	fixed	no	no	no
dev-s450	27	obl	obl	obl	obl	no	no	no
dev-s450	29	aux	aux	aux	aux	no	no	no
dev-s475	5	fixed	fixed	fixed	fixed	no	no	no
dev-s475	7	sec	sec	sec	sec	no	no	no
dev-s475	13	fixed	fixed	fixed	fixed	no	no	no
dev-s475	29	sec	sec	sec	sec	no	no	no
dev-s475	30	fixed	fixed	fixed	fixed	no	no	no
dev-s500	8	sec	sec	sec	sec	no	no	no
dev-s500	9	nmod	nmod	nmod	nmod	no	no	no
dev-s525	8	flat	flat	flat	flat	no	no	no
dev-s525	9	flat	flat	flat	flat	no	no	no
dev-s525	10	obl	obl	obl	obl	no	no	no
dev-s525	13	fixed	fixed	fixed	fixed	no	no	no
dev-s525	21	obl	obl	obl	obl	no	no	no
dev-s525	28	aux	aux	aux	aux	no	no	no
dev-s525	29	fixed	fixed	fixed	fixed	no	no	no
dev-s525	32	obl	obl	obl	obl	no	no	no
dev-s525	33	aux	aux	aux	aux	no	no	no
dev-s525	39	fixed	fixed	fixed	fixed	no	no	no

Table 3: The manual analysis of English. C: Convertible; HA: Head Affected; CR: Conversion Responsible.

Sent. ID	Tok.	Orig. UD	Auto. TUD	Man. TUD	Our UD	C	HA	CR
dev-s1	5	nsubj	sbj	sbj	nsubj	yes	no	no
dev-s1	12	dobj	obj	obj	dobj	yes	no	no
dev-s1	18	advmod	obl	qlfy	advmod	yes	no	yes
dev-s1	24	compound	cxp	cxp	compound	yes	no	no
dev-s2	7	advmod	obl	qlfy	advmod	yes	no	yes
dev-s2	15	advmod	obl	qlfy	advmod	yes	no	yes
dev-s3	3	advmod	obl	qlfy	advmod	yes	no	yes
dev-s3	9	advmod	obl	qlfy	advmod	yes	no	yes
dev-s4	2	advmod	obl	qlfy	advmod	yes	no	yes
dev-s4	11	advmod	obl	qlfy	advmod	yes	no	yes
dev-s5	4	advmod	obl	qlfy	advmod	yes	no	yes
dev-s5	8	advmod	obl	qlfy	advmod	yes	no	yes
dev-s6	6	advmod	obl	qlfy	advmod	yes	no	yes
dev-s6	14	advmod	obl	sec	advmod	yes	no	yes
dev-s7	3	compound	cxp	compound	compound	yes	no	yes
dev-s7	9	advmod	obl	qlfy	advmod	yes	no	yes

Table 4: The contingency table for McNemar's test.

	After (TUD) Correct	After (TUD) Incorrect
Before (UD) Correct	A	B
Before (UD) Incorrect	C	D

Table 5: Rules' contributions. A: Absolute improvement (degradation) to tokens' dependency label prediction undergoing each transformation rule. n: total frequency. P: relative frequency.

Rule	n	P	Transition-based		Graph-based		
			A	$A \times P$	A	$A \times P$	p-value
advmod → qlfy	32	0.01%	-14.14	-0.0016	-9.09	-0.0010	.01
xcomp → sec	1,108	0.40%	-7.70	-0.0306	-9.90	-0.0393	.00
nsubj → sbj	20,510	1.34%	-0.20	-0.0146	-0.46	-0.0335	.08
cop → cxp	3,177	1.35%	-0.02	-0.0003	-0.32	-0.0044	.94
compound → cxp	3,195	1.14%	0.25	0.0029	-0.56	-0.0064	.32
aux → aux*	8,568	3.07%	-0.18	-0.0056	0.11	0.0034	.10
x → x	181,148	64.85%	0.05	0.0319	0.08	0.0490	.17
advmod → obl	11,853	4.24%	0.49	0.0208	0.59	0.0252	.00
amod → mod	12,923	4.63%	0.51	0.0235	0.68	0.0314	.00
obj → obj*	14,427	5.11%	0.19	0.0098	1.12	0.0580	.18
det → mod	9,810	3.51%	0.16	0.0068	1.30	0.0458	.00
nummod → mod	4,485	1.61%	0.42	0.0068	2.22	0.0357	.04
xcomp → comp	2,391	0.86%	0.72	0.0061	2.29	0.0196	.08
csubj → sbj	485	0.17%	2.63	0.0046	2.79	0.0048	.00
ccomp → comp	2,965	1.06%	2.92	0.0310	3.46	0.0367	.00
iobj → obj*	1,643	0.59%	8.39	0.0493	21.40	0.1259	.00
SUM	280,220	100%	-4.93	0.1605	15.72	0.3509	.00