A Novel Approach for Root Selection in the Dependency Parsing

Sharefah AL-Ghamdi, Hend Al-Khalifa, Abdulmalik Al-Salman
College of Computer and Information Sciences,
King Saud University, P.O. Box 145111, Riyadh 4545, Saudi Arabia
{sharefah, hendk, salman}@ksu.edu.sa

Abstract

Although syntactic analysis using the sequence labeling method is promising, it can be problematic when the labels sequence does not contain a root label. This can result in errors in the final parse tree when the postprocessing method assumes the first word as the root. In this paper, we present a novel postprocessing method for BERT-based dependency parsing as sequence labeling. Our method leverages the root's part of speech tag to select a more suitable root for the dependency tree, instead of using the default first token. We conducted experiments on nine dependency treebanks from different languages and domains, and demonstrated that our technique consistently improves the labeled attachment score (LAS) on most of them.

Keywords: Dependency Parsing, Sequence Labeling, Natural Language Processing, BERT, Transformers

1. Introduction

Dependency parsing is the task of identifying the syntactic structure of a sentence by assigning a head (parent) and a label to each word (child). Traditionally, this task has been approached using transition or graph-based methods, which rely on explicit parsing algorithms or auxiliary structures. However, it has been shown that dependency parsing can also be performed as a sequence-labeling problem (Lacroix, 2019; Strzyz et al., 2019), where each word is associated with a label that encodes its head and syntactic information. Strzyz et al. (2019) show that this approach offers a good trade-off between parsing accuracy and speed as it leverages the efficiency of deep learning frameworks running on GPUs.

One of the challenges of dependency parsing as sequence labeling is the postprocessing stage, which can introduce errors in the syntactic analysis. A common flaw in this stage is that if the parser does not assign any word a label to be the root of the parse tree, it selects the first word in the sentence as the head of the syntactic tree and updates the rest of the labels accordingly (Vilares et al. 2020). Al-Ghamdi et al. (2023) highlighted this issue and showed that it was the reason for some errors in the final syntactic results.

In this work, we aim to reduce the effect of postprocessing on dependency parsing as sequence labeling. We propose a root part-of-speech (POS) identification as postprocessing method that predicts the root POS tag for a given sentence. Instead of choosing the first token of the sentence as the root when the parser fails to label a root, we choose the first token that has the predicted POS tag as the root. This way avoids some errors in the syntactic analysis.

To apply the proposed method, a root POS identifier (RPI) was built by fine-tuning a BERT pretrained model for text classification to perform the proposed solution of identifying the root POS for a sentence. This work is an enhancement of a previous exploratory work that also attempted to build a root index identifier model, yet the model was not accurate enough to find the correct roots. The details of that exploration are beyond the scope of this paper and will be reported elsewhere.

Our method was evaluated on nine different treebanks, and we showed that it could improve the Label attachment scores (LAS) and unlabeled attachment scores (UAS) of BERT-based dependency parsing for most parsers.

The rest of this paper is organized as follows: Section 2 reviews the postprocessing for the parse trees and presents the RPI used in the proposed method. Section 3 describes the experiments setup, including the data sets and the baseline parser. Section 4 reports and analyzes the results of our method and compares it with the baseline. Finally, Section 5 concludes the paper with limitations and future work.

2. Postprocessing with root POS identification

The root's POS tag is the tag of the word that serves as the syntactic head of the parse tree for the sentence. It also can provide useful information for downstream tasks, such as parsing and semantic analysis. In this section, we propose a novel root POS identification method for postprocessing in the sequence labeling dependency parsing. The following two sub-sections explain the postprocessing steps of the parse tree and present the proposed RPI.

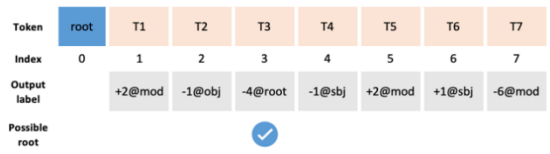
2.1 The Postprocessing for Parse Trees

The postprocessing implementation of Vilares et al. (2020) shows how to construct a parse tree from a labeled sequence, which is the output of the model. If the output labels do not include a root or a possible root, the first token in the sequence is assigned as the root node. The root is a token that has a head index of 0 and a relation label of the root. In contrast, the possible root is a token that has a head index that is not the root index but has a relation label of the root. Figure 1 shows an example of an output label sequence that does not include a root token but includes a possible root token. Token number 3 has a head index of -4, which means the fourth token to the left, but there are only three tokens to the left.

OSACT 2024 Workshop, pages 46–49

However, it has a relation label @root. Therefore, it is selected as the root node of the parse tree.

Figure 1: An example of the output of labels that



includes possible root.

Instead of selecting the first token as the root node, we apply our novel postprocessing step that identifies the root POS tag for a given sequence. Then, we select the first token that has the predicted POS tag as the root node. For instance, in the example shown in Figure 2, the third token in the sequence was selected as the root node, because it was the first verb token, and the RPI predicted the root POS tag as a verb.

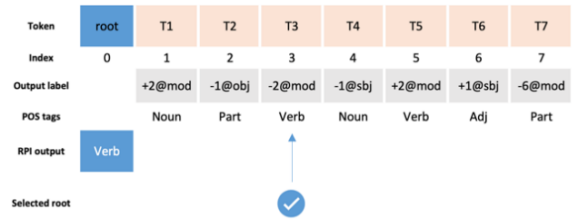


Figure 2: An example of assigning root using RPI.

To perform our proposed postprocessing step, we designed a simple but effective root POS identifier (RPI) based on BERT. It can recognize the POS tag of the syntactic root of the input sentence. We describe the details of BERT-based RPI in the next section.

2.2 The proposed RPI

We formulate the task of root POS identification as a text classification problem, which is a natural language processing task that aims to assign a label to a given sequence of words. For example, given a sentence, one can classify it as a positive or negative sentiment, or as a question or a statement.

We implemented an RPI as a simple text classification task using the pretrained BERT model (Devlin et al., 2018). We fine-tuned the pretrained language model to predict the root part-of-speech (POS) tag for an input sequence. For example, given an input sentence S with a sequence of tokens [T_1, …, T_n], the model predicts the POS tags that corresponds to the syntactic root of S.

3. Experiments

To test our proposed method, we used nine datasets as our input. Seven of them came from the UD2.12 universal dependency treebanks (Zeman et al., 2023). The other two were Arabic datasets: one was the converted version of the Penn Arabic Treebank (ATB) part2 v3.1 (Diab et al., 2013), which was part of the Columbia Arabic Treebank (CATiB), and the other was the Classical Arabic poetry dependency treebank (ArPoT) (Al-Ghamdi et al., 2021). Table 1 shows the number of different root part-of-speech (POS) tags (classes).

We measured the accuracy of our BERT-based RPI in predicting the correct POS tags of the root nodes. Then, we evaluated the effectiveness of our method with the BERT-based sequence labeling dependency parser (P-w-RPI) by comparing the UAS/LAS scores with the baseline parsers (BL) without RPI.

The baseline parser we built is based on (Vilares et al. 2020) and (Al-Ghamdi et al., 2023). The list of the fine-tuned BERT models used for each language is presented in Table 1. We also show the number of different POS tags for roots in each dataset.

We used Colab T4 GPUs to train all models on 10 epochs. All experiment's implementation codes and settings will be released on Github: https://github.com/Sharefah-Alghamdi

Table 1: Number of Root POS tags and BERT models for nine treebanks under study.

Dataset	Roots' POS	BERT Model
AR $_{ArPoT}$	5	bert-base-arabertv2
AR $_{CATiB}$	6	
AR $_{PADT}$	14	
EL $_{GDT}$	15	bert-base-greek-uncased-v1
EN $_{EWT}$	8	bert-base-uncased
FR $_{ParTUT}$	8	bert-base-french-europeana-cased
TA $_{TTB}$	6	tamil-bert
TR $_{IMST}$	13	bert-base-turkish-uncased
ZH $_{GSD}$	10	bert-base-chinese

4. Results and Analysis

The results of our RPI on the nine datasets are shown in Table 2. The highest accuracy was obtained for Tamil (96.67%), followed by Chinese (91.6%), Arabic (PADT) (91.32%), and Arabic (CATiB) (90.1%). The lowest accuracy was obtained for Turkish (76.92%), followed by Arabic (ArPoT) (79.34%), and Greek (84.21%). These results indicate that our model can effectively predict the root POS tag for most datasets, but there is still room for improvement for some languages.

Table 2: RPI accuracy for nine datasets.

Dataset	RPI
AR ArPoT	79.34
AR CATiB	91.23
AR PADT	91.32
EL GDT	84.21
EN EWT	88.88
FR ParTUT	88.18
TA TTB	96.67
TR IMST	76.92
ZH GSD	91.6
Average	**87.6**

We hypothesize that there is a relation between the difficulty of syntactic root POS identification and the difficulty of grammatical nature understanding for the language model. For example, some languages may have more complex or irregular word forms, or more syntactic variations than others. These factors may make it harder to predict the root POS tag for some languages than others. However, the average accuracy (87.6%) across all datasets shows that BERT-based text classification models can sufficiently perform our RPI.

We evaluated the impact of using RPI in the postprocessing stage on the parsing accuracy of the BERT-based dependency parser. The UAS/LAS scores of BL and BL using RPI on the nine treebanks are shown in Table 3. We calculated the average results over three runs to ensure the reliability and consistency of our models.

Table 3: UAS/LAS of the baseline (BL) parser with and without RPI.

Dataset	BL		P-w-RPI	
AR ArPoT	80.01	74.21	▲ 80.02	▲ 74.22
AR CATiB	87.54	86.47	■ 87.54	■ 86.47
AR PADT	84.49	80.55	■ 84.49	■ 80.55
EL GDT	62.24	54.88	▲ 62.37	▲ 55.31
EN EWT	83.94	80.62	▲ 84.05	▲ 80.71
FR ParTUT	89.05	87.67	▲ 89.11	▲ 87.73
TA TTB	58.50	47.16	▲ 58.64	▲ 47.24
TR IMST	63.75	51.19	■ 63.75	▼ 51.13
ZH GSD	77.10	74.02	▲ 77.21	▲ 74.12

We found that adding RPI in the postprocessing step improved UAS scores for six out of the nine treebanks, whereas the Arabic (CATiB and PADT) and Turkish had no change. The LAS scores also increased for six of treebanks. Arabic (CATiB and PADT) also had no change, and Turkish had a slight decrease (-0.06%). The highest UAS improvement was achieved by Tamil (+0.14%), followed by Greek (+0.13%). The lowest improvement was achieved by Arabic (ArPoT), which had negligible changes (0.01% and 0.02%) in UAS and LAS respectively. These results indicate that the postprocessing step using RPI can enhance the quality of parsing results by selecting more appropriate roots.

We analyzed the results of our experiments in different scenarios and found their strengths and limitations. Table 4 presents the analysis metrics of the proposed method on various datasets. The metrics are:

- **No root:** The percentage of trees without roots generated by the baseline parser.
- **Possible roots (PR):** The percentage of trees that are treated by using possible roots in the output labels.
- **Processed with RPI (w-RPI)**: The percentage of trees that processed with the use of our RPI. It is equal to the No root minus the PR columns.
- **Correct POS (c-POS)**: The percentage of correct root POS tags predicted by our RPI model.
- **First Token (FT)**: The percentage of cases where the first token with the predicted POS tag in the sentence is the correct root, as in the gold dataset. For example, the first verb is the correct root, not the second or third verb. (we counted only the roots that their POS predicted correctly).

Table 4: Average percentages for nine treebanks on metrics related to parse tree roots: No root, possible root (PR), processed by RPI (w-RPI), correct POS (c-POS), First Token roots with predicted POS (FT).

Dataset	No root	PR	w-RPI	c-POS	FT
AR ArPoT	8.36	7.63	0.74	83.33	66.67
AR CATiB	0.29	0	0.29	0	0
AR PADT	0.88	0	0.73	20.95	33.33
EL GDT	7.16	0	7.01	83.06	72.22
EN EWT	1.70	0	1.64	82.36	82.57
FR ParTUT	0.91	0	0.91	0	0
TA TTB	0.56	0	0.56	100	100
TR IMST	1.78	0	1.78	53.57	38.26
ZH GSD	3.20	0	3.13	84.32	33.98

Table 4 shows that the number of trees without roots varies across languages and treebanks, from 0.29% for Arabic (CATiB) to 8.36% for Arabic (ArPoT). Except for Arabic (ArPoT), none of the trees without roots have any possible root tokens (PR). That means our proposed postprocessing step is needed for most of the trees without root labels. The table also shows that our RPI was applied on a considerable proportion of the trees, especially for Greek (7.01%) and Chinese (3.13%).

The accuracy of RPI on predicting the root POS tag (c-POS) is also reported in Table 2. This metric explains why some datasets did not show any improvement after applying the postprocessing step. The datasets with a small number of trees that were generated without a root had lower accuracy of c-POS prediction. For instance, Arabic (CATiB and PADT) had one and six root-less trees in each respectively, and they reported low accuracy of root POS prediction by RPI. However, the postprocessing step of choosing the root instead of the first token still improved the results for these datasets, even when the POS tag was incorrectly identified for some languages such as French (improved by +0.06 in both UAS and LAS in Table 3). On the contrary, Turkish had a low accuracy of root POS prediction, which was consistent with the low performance of RPI for Turkish in Table 2.

Our method achieved better results on Greek treebank than on other treebanks, because this treebank had several factors that suited our method. First, the RPI treated a relatively large proportion of trees without roots. Second, the RPI was very accurate in predicting the POS tag of the root word in the sentence. Third, for most sentences, if there were more than one token with the predicted POS tag (three verbs, for example), usually the first one in the sequence was the correct root.

The results illustrated how our postprocessing step improves the quality and completeness of the parse trees by finding a valid root node. It also highlighted the languages' differences and challenges in predicting the correct root based on the POS tag.

5. Conclusion

The paper shows that our root POS identification as postprocessing can improve the results of the dependency parser as a sequence labeler by selecting a more proper syntactic root. The results show that our method can enhance the completeness of the dependency structure in the parse tree. We evaluated our method on nine treebanks, and demonstrated that it can enhance UAS/LAS scores over most of them.

The work also revealed that the postprocessing of the syntactic structures of sentences had different effects on different treebanks, depending on the nature of the syntactic relations in those treebanks. Therefore, we identified a limitation of our method, as the high accuracy of the RPI might not be enough to determine the correct root. For instance, if most of the sentences in a treebank have a root that is the second or third word in the sequence that has the predicted POS, our method might not be beneficial. Moreover, there might be some treebanks that we have not examined, and that might not produce sequences without roots, and in this case, there is no need for any postprocessing at all.

In the future, we might explore more treebanks and look for solutions that make our method universally applicable to all languages and treebanks.

6. References

David Vilares, Michalina Strzyz, Anders Søgaard, and Carlos Gómez-Rodríguez. 2020. *Parsing as pre-training*. In *Proceedings of the AAAI Conference on Artificial Intelligence*, (vol. 34, No. 05, pp. 9114-9121). https://doi.org/10.1609/aaai.v34i05.6446

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. *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) (pp. 4171-4186). http://dx.doi.org/10.18653/v1/N19-1423

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. "*Viable Dependency Parsing as Sequence Labeling.*" 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), pp. 717-723. 2019. http://dx.doi.org/10.18653/v1/N19-1077

Ophélie Lacroix. 2019. *Dependency parsing as sequence labeling with head-based encoding and multi-task learning.* In *Proceedings of the Fifth International Conference on Dependency Linguistics* (Depling, SyntaxFest 2019) (pp. 136-143). http://dx.doi.org/10.18653/v1/W19-7716

Sharefah Al-Ghamdi, Hend Al-Khalifa. and Abdulmalik Al-Salman 2023. *Fine-Tuning BERT-Based Pre-Trained Models for Arabic Dependency Parsing*. *Applied Sciences*, 13(7), Article # 4225. https://doi.org/10.3390/app13074225

7. Language Resource References

Daniel Zeman, Joakim Nivre; et al. 2023*. Universal dependencies 2.12,* LINDAT/CLARIAH-CZ *digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.*

Mona Diab, Nizar Habash, Owen Rambow and Ryan Roth. 2013. *LDC Arabic treebanks and associated corpora: Data divisions manual*.

Sharefah Al-Ghamdi, Hend Al-Khalifa., and Abdulmalik Al-Salman. 2021. *A Dependency Treebank for Classical Arabic Poetry*. In *Proceedings of the Sixth International Conference on Dependency Linguistics,* (Depling, SyntaxFest 2021), Sofia, Bulgaria, 21–25 March 2021; pp. 1–9