Proceedings

Sequence Tagging for Fast Dependency Parsing †

Michalina Strzyz *, David Vilares and Carlos Gómez-Rodríguez

CITIC, FASTPARSE Lab, Departamento de Computación, Campus de Elviña, Universidade da Coruña, 15071 A Coruña, Spain

* Correspondence: michalina.strzyz@udc.es

† Presented at XoveTIC Congress, A Coruña, Spain, 5–6 September 2019.

Published: 26 August 2019

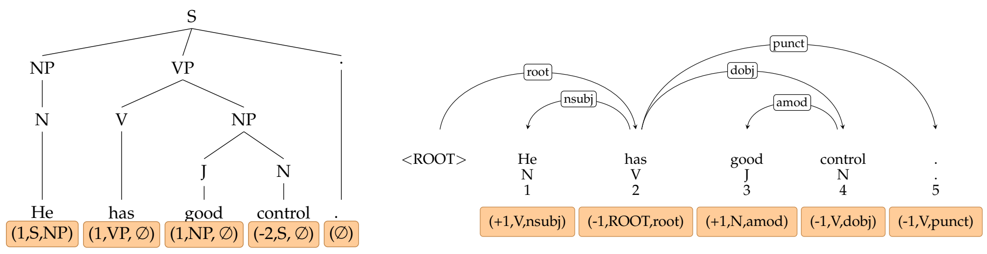
Abstract: Dependency parsing has been built upon the idea of using parsing methods based on shift-reduce or graph-based algorithms in order to identify binary dependency relations between the words in a sentence. In this study we adopt a radically different approach and cast full dependency parsing as a pure sequence tagging task. In particular, we apply a linearization function to the tree that results in an output label for each token that conveys information about the word's dependency relations. We then follow a supervised strategy and train a bidirectional long short-term memory network to learn to predict such linearized trees. Contrary to the previous studies attempting this, the results show that this approach not only leads to accurate but also fast dependency parsing. Furthermore, we obtain even faster and more accurate parsers by recasting the problem as multitask learning, with a twofold objective: to reduce the output vocabulary and also to exploit hidden patterns coming from a second parsing paradigm (constituent grammars) when used as an auxiliary task.

Keywords: Natural Language Processing; Syntax; Parsing; Sequence Tagging, Multitask Learning

1. Introduction

One of the building blocks in Natural Language Processing (NLP) is parsing, that provides syntactic analyses of a text. A structure of a sentence is commonly represented as constituency [1] or dependency tree [2]. Constituency grammar introduces the notion of *constituents* where a sentence is decomposed into sub-phrases while in dependency, words are connected according to their *dependency relation* (every word in a sentence is *dependent* on another word that is defined as its *head*). An example of each tree structure is given in Figure 1. Various parsing algorithms have been developed for constituency and dependency parsing. For the latter, transition- and graph-based approaches have been most widely used. In transition-based (or shift-reduce) dependency parsing, the best transition (create an arc between two words, do a shift, reduce a word, ...) is predicted at each timestep given the state of the current configuration of the parser [3]. In contrast, a graph-based parser explores incrementally all possible parses of a tree through graph fragments and a tree with the highest score is chosen [4].

In this context, neural architectures have gained popularity in the field of NLP, where long short-term memory (LSTM) networks have been proven to be useful in many problems, because of their ability to decide which information to remember [5]. This is especially useful in dependency parsing, where we have to identify long-distance relationships between words. In addition, it has been shown that these architectures can benefit from learning various tasks jointly, the so-called *multitask learning* (MTL) [6]. In MTL setups, it can be also helpful to add an *auxiliary task*, whose result is not relevant but can be used to improve the performance on the main task. This is due to the ability of the network to exploit hidden patterns that are present in the main task and the ability of the shared representations to prevent overfitting.



(**a**) A constituency tree (**b**) A dependency tree

Figure 1. An example of syntactic trees for the same sentence represented under the constituency and dependency formalisms. Below, labels for each token encoding the trees.

2. Method

Recent research has shown that constituency parsing can be reduced to *sequence tagging*, a structured prediction problem where for each input token a single label output is generated [7]. To do so, the syntactic trees need to be linearized through an encoding method, as shown in Figure 1a.

In a similar fashion, we propose to apply sequence tagging models for dependency parsing [8], using NCRF++ [9] as our sequence tagging framework. We propose a part-of-speech tag-based (PoS) encoding where the information of token's head and dependency relation is encapsulated in a label of the form (p_i, h_i, r_i). The first element p_i of the tuple encodes the relative distance to the token's head in terms of words with a PoS tag h_i, and where r_i is the dependency relation between those two tokens. An example of an encoded dependency tree is shown in Figure 1b. For instance, the label for the token *"control"* is (-1,V,DOBJ) which means that the head is the first token to the left (-1) among those with the PoS tag V, and that the dependency relation is DOBJ.

Furthermore, it has been shown that constituency parsing can leverage from MTL setups [10]. Hence, our model attempts to learn dependency label as a 2-task setup where: one task consists of learning (p_i, h_i) since they are the most closely related among the elements in the tuple, and the second task consists in learning the dependency relation (r_i). Additionally, we also explore whether constituency parsing as auxiliary task can improve the performance of dependency parsing as the main task.

3. Results

We evaluate models on the English Penn Treebank [11]. We use the standard metrics: Unlabeled and Labeled Attachment Score (UAS/LAS). Table 1 shows that our single-task model provides a good trade-off between speed and accuracy in comparison with existing transition- and graph-based models. In Table 2 we show that our model achieves even better performance when applying MTL, where dependency parsing as tagging is better learned when treating it as 2-task (S-MTL). Finally, the best result for dependency parsing is achieved when adding constituency parsing as auxiliary task (D-MTL-AUX). More experiments on various languages and the reported speeds when including the MTL approach are presented in [12].

Table 1. Model's speed and accuracy compared with existing dependency parsers on the PTB test set. ◇ speeds taken from the original papers.

Model	**sent/s**		**UAS**	**LAS**
	CPU	**GPU**		
KG (transition-based) [13]	$76_{\pm1}$		93.90	91.90
KG (graph-based) [13]	$80_{\pm0}$		93.10	91.00
CM [14]	$654^{\diamond}$		91.80	89.60
DM [15]		$411^{\diamond}$	95.74	94.08
Stack-Pointer [16]		$10_{\pm0}$	95.87	94.19
Our model	$101_{\pm2}$	$648_{\pm20}$	**93.67**	**91.72**

Table 2. Unlabeled Attachment Score (UAS), Labeled Attachment Score (LAS) and speed on a single core CPU for the MTL models on the PTB test sets. S-S: single model, S-MTL: 2-task, D-MTL-AUX: with constituency parsing as auxiliary task.

Model	Dependency Parsing		Speed (CPU)
	UAS	LAS	sent/sec
S-S	93.60	91.74	$117_{\pm 6}$
S-MTL	93.84	91.83	$133_{\pm 1}$
D-MTL-AUX	**94.05**	**92.01**	$133_{\pm 1}$

4. Discussion

We have obtained a fast and accurate dependency parsing method showing that the dependency parsing problem can be reduced to a conceptually simple sequence tagging task where dependency trees are encoded into labels. In this way, our research has put emphasis not only on the accuracy of dependency parsing, but also on improving the speed, to make it feasible to parse the big amounts of data available today.

Funding: This work has received funding from the European Research Council (ERC), under the European Union's Horizon 2020 research and innovation programme (FASTPARSE, grant agreement No 714150), from the ANSWER-ASAP project (TIN2017-85160-C2-1-R) from MINECO, and from Xunta de Galicia (ED431B 2017/01).

References

1. Chomsky, N. Three models for the description of language. *IRE Trans. Inf. Theory* **1956**, *2*, 113–124.
2. Mel'cuk, I.A. *Dependency Syntax: Theory and Practice*; State University of New York Press: Albany, NY, USA, 1988.
3. Nivre, J. An efficient algorithm for projective dependency parsing. In Proceedings of the Eighth International Conference on Parsing Technologies, Nancy, France, 23–25 April 2003; pp. 149–160.
4. McDonald, R.; Crammer, K.; Pereira, F. Online large-margin training of dependency parsers. In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05), Ann Arbor, MI, USA, 25–30 June 2005; pp. 91–98.
5. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780.
6. Caruana, R. Multitask learning. *Mach. Learn.* **1997**, *28*, 41–75.
7. Gómez-Rodríguez, C.; Vilares, D. Constituent Parsing as Sequence Labeling. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 1314–1324.
8. Strzyz, M.; Vilares, D.; Gómez-Rodríguez, C. 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, Minneapolis, USA, 2–7 June, 2019; pp. 717–723
9. Yang, J.; Zhang, Y. NCRF++: An Open-source Neural Sequence Labeling Toolkit. In Proceedings of ACL 2018, System Demonstrations, Melbourne, Australia, 15–20 July, 2018; pp. 74–79.
10. Vilares, D.; Abdou, M.; Søgaard, A. Better, Faster, Stronger Sequence Tagging Constituent Parsers. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, USA, 2-7 June, 2019; pp. 3372–3383.
11. Marcus, M.P.; Marcinkiewicz, M.A.; Santorini, B. Building a large annotated corpus of English: The Penn Treebank. *Comput. Linguist.* **1993**, *19*, 313–330.
12. Strzyz, M.; Vilares, D.; Gómez-Rodríguez, C. Sequence Labeling Parsing by Learning Across Representation. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019), Florence, Italy, 28 July–2 August, 2019; pp. 5350–5357.
13. Kiperwasser, E.; Goldberg, Y. Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations. *Trans. Assoc. Comput. Linguist.* **2016**, *4*, 313–327.

14. Chen, D.; Manning, C. A fast and accurate dependency parser using neural networks. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October, 2014; pp. 740–750.
15. Dozat, T.; Manning, C.D. Deep Biaffine Attention for Neural Dependency Parsing. In Proceedings of the 5th International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
16. Ma, X.; Hu, Z.; Liu, J.; Peng, N.; Neubig, G.; Hovy, E. Stack-Pointer Networks for Dependency Parsing. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July, 2018; pp. 1403–1414.