Parse Me if You Can: Artificial Treebanks for Parsing Experiments on Elliptical Constructions

Kira Droganova, Daniel Zeman
Charles University, Faculty of Mathematics and Physics
Malostranské náměstí 25, Praha, Czechia
{droganova, zeman}@ufal.mff.cuni.cz

Abstract

In this work we focus on a particular linguistic phenomenon, ellipsis, and explore the latest parsers in order to learn about parsing accuracy and typical errors from the perspective of elliptical constructions. For this purpose we collected and processed outputs of several state-of-the art parsers that took part in the CoNLL 2017 Shared Task. We extended the official shared task evaluation software to obtain focused evaluation of elliptical constructions. Since the studied structures are comparatively rare, and consequently there is not enough data for experimentation, we further describe the creation of a new resource, a semi-artificially constructed treebank of ellipsis.

Keywords: Ellipsis, Syntactic parsing, Evaluation, Universal Dependencies

1. Introduction

Ellipsis, i.e. omission of linguistic content that is silently understood by both the speaker and the addressee, is a phenomenon present—in various forms—in many natural languages. Ellipsis obviously makes natural language understanding harder; but sometimes it also complicates syntactic parsing of the content that is not omitted. In dependency syntax (which is the framework within which we operate), a parent node may be missing while its dependents are present. One might either create an "empty" node for the missing word, or choose a substitute parent among the words that are not missing. Both options make parsing difficult: in the former case, the parser must learn where to generate empty nodes; in the latter, relations are drawn between nodes that would not be connected otherwise, hence they are not easily learned from data.

In any case, modern dependency parsers *are* data-driven and they can hardly account for those types of ellipsis that are not represented in training data. If the data contains empty nodes, the parser can try to learn generating them. If the data does not contain any specific annotation of ellipsis, we have to hope that the parser learns to occasionally attach dependents to strange parents, even without knowing that it is ellipsis what caused the lack of better options.

In this study we focus on elliptical constructions in the so-called *basic representation* of Universal Dependencies (UD) (Nivre et al., 2016). The annotation style of UD does not mark ellipsis explicitly when it does not have to: most types are solved by simply promoting one orphaned dependent to the position of its missing parent. Admittedly, there are treebanks that overtly annotate a wider range of elliptical structures. Our main reason for working with UD is practical: substantial data is available in this annotation style for several dozens of languages, and state-of-the art parsers have been trained and tested on UD.

The one exception where UD explicitly marks ellipsis are certain types of gapping and stripping (Droganova and Zeman, 2017), where multiple orphaned dependents of a missing predicate have to be connected using a special relation called `orphan` (Figure 1). In the present work we investigate how frequent are the `orphan` relations in data, how well can existing parsers learn to recognize them, and how can we extend the data to provide more training material and improve parsing accuracy.

2. Data

For the purpose of the experiments we use the system outputs from the CoNLL 2017 Shared Task (Zeman et al., 2017), that are now available as a corpus. We chose 12 teams whose systems surpassed baseline results (Zeman et al., 2017) on labelled attachment score (LAS): C2L2, darc, HIT-SCIR, IMS, Koç University, LATTICE, NAIST-SATO, Orange-Deskiñ, Stanford, TurkuNLP, ÚFAL-UDPipe 1.2 and UParse.

3. Experiments

The idea behind this work is to look closely at the latest parsers regarding their ability to parse non trivial linguistic constructions such as elliptical constructions, and collect the information about typical errors, how they differ from parser to parser.

For the purpose of this experiment we adapted and extended the evaluation script which had been created to evaluate system output files for the 2017 Shared Task. The main idea of such adaptation is to save evaluation techniques that were proposed and implemented by the 2017 task organizers. Since the data was selected relying on these techniques, we hope that following the same line, especially regarding word alignments and sentence segmentation, helps us to be more precise. The script is available at the Shared Task page. [1] The adapted script can be found on github [2]

The adapted script provides information of two types:

- Statistics on correctly predicted `orphan` relations;
- Statistics on erroneously predicted or missed `orphan` relations and typical errors.

[1] http://universaldependencies.org/conll17/evaluation.html
[2] https://github.com/Kira-D/conll2017/tree/deprelCalc

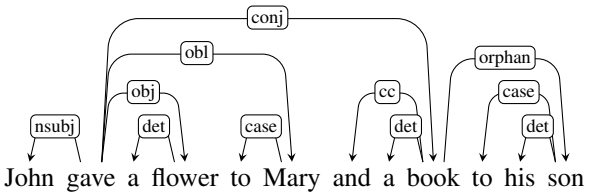

Figure 1: UD v2 uses the `orphan` relation to attach unpromoted dependents of a predicate to the promoted dependent.

4. Evaluation

Table 1 shows the statistics on correctly predicted `orphan` relations. In these calculations we use the relative number of all `orphan` nodes for every team, which is based on alignment between system output words and gold standard words. In other words, only successfully aligned `orphan` nodes from gold standard are included in this number. It is clearly seen that both Recall and F-measure are rather low. At the same time, percentage of correctly predicted dependency labels for head nodes is quite high.

Table 2 shows the statistics on erroneously predicted or missed `orphan` labels. For every parser that we selected for the experiment, we calculate error pares "relation1-relation2", where the first relation was taken from the aligned gold word and the second relation was assigned by the system. Table 2 provides top 5 error pairs. Every cell contains the following information:

- the error pair;
- the contribution of the pair to the number of all errors concerning `orphan` label (percentage);
- the number of instances of the error type (frequency);
- h.error shows erroneously predicted head nodes (percentage and absolute number).

It seems that parsers make mistakes in similar conditions: the error types and their frequencies are almost the same from parser to parser.

What is important, the number of `orphan` labels is just a tiny fraction of all labels and the contribution of their low values of Recall and F-measure to the final figures calculated on the whole amount of data goes virtually unseen. Hence, the question is if the parsers perform really poorly on elliptical constructions or it is simply the lack of data.

We would answer with the proposal of creating artificial treebanks for parsing experiments and find out.

Parser	All	Correct	Recall	F1	Parent	Parent %
C2L2	1420	217	15.28%	26.48%	192	88.48%
darc	1411	194	13.75%	19.06%	180	92.78%
HIT-SCIR	1411	341	24.17%	34.13%	292	85.63%
IMS	1421	241	16.96%	28.83%	208	86.31%
Koc-University	1420	194	13.66%	20.78%	161	82.99%
LATTICE	1420	200	14.08%	20.62%	166	83.0%
NAIST-SATO	1420	391	27.54%	41.53%	357	91.3%
Orange-Deskin	1420	369	25.99%	35.16%	280	75.88%
Stanford	1420	454	31.97%	49.11%	408	89.87%
TurkuNLP	1420	218	15.35%	23.37%	189	86.7%
UFAL-UDPipe-1-2	1423	226	15.88%	23.69%	182	80.53%
UParse	1420	326	22.96%	33.44%	288	88.34%

Table 1: Correctly predicted `orphan` relations. Parser: names of the teams in alphabetic order; All: number of `orphan` labels; Correct: number of correctly predicted `orphan` labels; Recall: number of correct `orphan` labels divided by the number of gold-standard `orphan` nodes; F1: f-measure: 2PR / (P+R); Parent: number of correctly predicted parent nodes; Parent %: percent of correctly predicted parent nodes;

5. Creating artificial treebanks

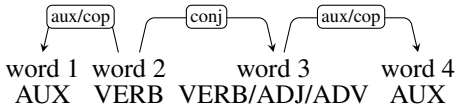

Figure 2: An example of a gapping pattern.

Recent research (Schuster et al., 2017; Droganova and Zeman, 2017) provides a detailed overview of elliptical constructions within the UD framework and presents typical patterns that can be used for detection of elliptic constructions. This information allows us to develop a script that transforms non-elliptic UD style trees to elliptic trees.

Figure 2 shows a subtree pattern that matches sentences where gapping (Johnson, 2009) could potentially occur (but it did not, or at least it was not annotated following the UD guidelines, because there is no `orphan` relation). An example of an English sentence that matches the pattern: "But not always do those three agree, and not always are their decisions equal."

Figure 3 provides the tree structure of this sentence. The sentence has a verb as a "root" node, which is linked with an auxiliary verb with "aux" relation and with an adjective with "conj" relation and this adjective linked with its dependent auxiliary with "cop" relation, therefore it would be a match. After transformation the sentence would lose an adjective and its dependent. The new structure is represented at Figure 4.

The methodology requires manual efforts. After application of the script, the data have to be checked and corrected:

- After artificial omission sentences must remain grammatically correct;
- The patterns are designed to match as many instances as possible, so the erroneous instances have to be filtered out.

Potentially, the methodology can be applied to all UD treebanks. We are currently working on Russian, Czech, and

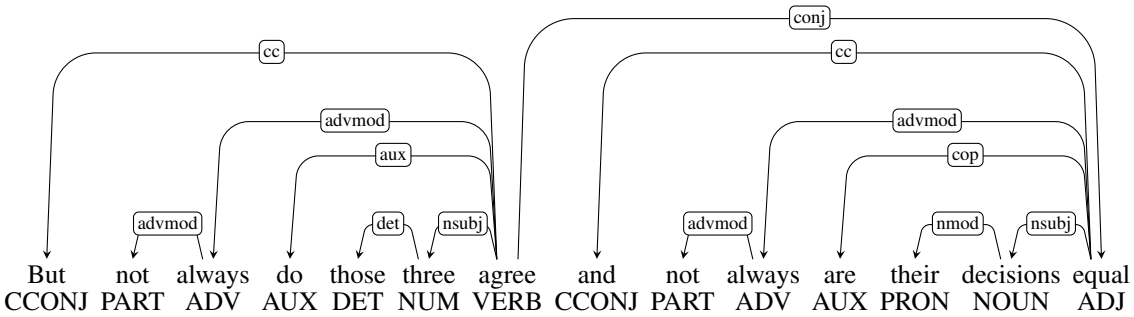

Figure 3: An example of a matched sentence.

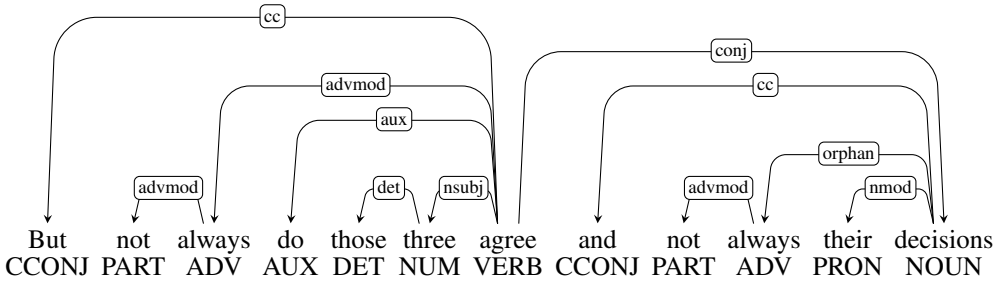

Figure 4: An example of a matched sentence.

English. We are planning to release these artificial elliptic UD treebanks after our manual checks and corrections. Artificial treebanks can facilitate testing and improving parsers performance regarding ellipsis. Hence, they would allow us to pay decent attention to this rare linguistic phenomenon.

6. Acknowledgements

The work was partially supported by the grant 15-10472S of the Czech Science Foundation, and by the GA UK grant 794417.

7. Bibliographical References

Droganova, K. and Zeman, D. (2017). Elliptic constructions: Spotting patterns in ud treebanks. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, number 135, pages 48–57.

Johnson, K. (2009). Gapping is not (VP) ellipsis. *Linguistic Inquiry*, 40(2):289–328.

Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajič, J., Manning, C., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. (2016). Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association.

Schuster, S., Lamm, M., and Manning, C. D. (2017). Gapping constructions in Universal Dependencies v2. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*.

Zeman, D., Popel, M., Straka, M., Hajic, J., Nivre, J., Ginter, F., Luotolahti, J., Pyysalo, S., Petrov, S., Potthast, M., et al. (2017). Conll 2017 shared task: Multilingual parsing from raw text to universal dependencies. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19.

8. Language Resource References

Zeman, Daniel and Potthast, Martin and Straka, Milan and Popel, Martin and Dozat, Timothy and Qi, Peng and Manning, Christopher and Shi, Tianze and Wu, Felix G. and Chen, Xilun and Cheng, Yao and Björkelund, Anders and Falenska, Agnieszka and Yu, Xiang and Kuhn, Jonas and Che, Wanxiang and Guo, Jiang and Wang, Yuxuan and Zheng, Bo and Zhao, Huaipeng and Liu, Yang and Teng, Dechuan and Liu, Ting and Lim, Kyungtae and Poibeau, Thierry and Sato, Motoki and Manabe, Hitoshi and Noji, Hiroshi and Matsumoto, Yuji and Kırnap, Ömer and Önder, Berkay Furkan and Yuret, Deniz and Straková, Jana and Vania, Clara and Zhang, Xingxing and Lopez, Adam and Heinecke, Johannes and Asadullah, Munshi and Kanerva, Jenna and Luotolahti, Juhani and Ginter, Filip and Kuan, Yu and Sofroniev, Pavel and Schill, Erik and Hinrichs, Erhard and Nguyen, Dat Quoc and Dras, Mark and Johnson, Mark and Qian, Xian and Liu, Yang and Vilares, David and Gómez-Rodríguez, Carlos and Aufrant, Lauriane and Wisniewski, Guillaume and Yvon, François and Dumitrescu, Stefan Daniel and Boroş, Tiberiu and Tufiş, Dan and Das, Ayan and Zaffar, Affan and Sarkar, Sudeshna and Wang, Hao and Zhao, Hai and Zhang, Zhisong and Hornby, Ryan and Taylor, Clark and Park, Jungyeul and de Lhoneux, Miryam and Shao, Yan and Basirat, Ali and Kiperwasser, Eliyahu and Stymne, Sara and Goldberg, Yoav and Nivre, Joakim and Akkuş, Burak Kerim and Azizoglu, Heval and Cakici, Ruket

C2L2	orphan-conj 23.0% 327 h.error: 85.63% 280 orphan-nmod 15.05% 214 h.error: 42.52% 91 orphan-obl 6.12% 87 h.error: 62.07% 54 orphan-advmod 6.05% 86 h.error: 72.09% 62 conj-orphan 5.91% 84 h.error: 61.9% 52
darc	orphan-conj 14.5% 267 h.error: 76.4% 204 orphan-nmod 12.6% 232 h.error: 47.84% 111 conj-orphan 7.98% 147 h.error: 78.23% 115 orphan-obl 5.75% 106 h.error: 55.66% 59 orphan-advmod 5.05% 93 h.error: 60.22% 56
HIT-SCIR	orphan-conj 16.05% 266 h.error: 79.32% 211 orphan-nmod 10.38% 172 h.error: 51.74% 89 conj-orphan 8.99% 149 h.error: 69.13% 103 orphan-obl 6.46% 107 h.error: 68.22% 73 orphan-advmod 4.59% 76 h.error: 80.26% 61
IMS	orphan-conj 22.92% 328 h.error: 81.4% 267 orphan-nmod 14.12% 202 h.error: 43.56% 88 orphan-obl 6.92% 99 h.error: 56.57% 56 orphan-advmod 6.5% 93 h.error: 67.74% 63 conj-orphan 5.87% 84 h.error: 69.05% 58
Koc-University	orphan-conj 20.5% 343 h.error: 79.01% 271 orphan-nmod 12.67% 212 h.error: 52.83% 112 conj-orphan 6.16% 103 h.error: 69.9% 72 orphan-obl 5.5% 92 h.error: 66.3% 61 orphan-advmod 4.72% 79 h.error: 70.89% 56
LATTICE	orphan-conj 17.24% 300 h.error: 82.0% 246 orphan-nmod 13.33% 232 h.error: 49.57% 115 conj-orphan 7.93% 138 h.error: 68.84% 95 orphan-obl 6.03% 105 h.error: 63.81% 67 orphan-advmod 5.11% 89 h.error: 67.42% 60
NAIST-SATO	orphan-conj 17.23% 257 h.error: 82.1% 211 orphan-nmod 11.73% 175 h.error: 45.71% 80 conj-orphan 9.72% 145 h.error: 56.55% 82 orphan-obl 6.7% 100 h.error: 67.0% 67 orphan-advmod 4.83% 72 h.error: 63.89% 46
Orange-Deskin	orphan-conj 15.14% 262 h.error: 71.37% 187 conj-orphan 10.34% 179 h.error: 69.27% 124 orphan-nmod 9.76% 169 h.error: 45.56% 77 orphan-obl 5.6% 97 h.error: 65.98% 64 orphan-advmod 4.68% 81 h.error: 66.67% 54
Stanford	orphan-conj 17.71% 247 h.error: 85.43% 211 orphan-nmod 12.19% 170 h.error: 45.88% 78 conj-orphan 10.9% 152 h.error: 61.84% 94 orphan-obl 5.3% 74 h.error: 64.86% 48 orphan-advmod 5.23% 73 h.error: 65.75% 48
TurkuNLP	orphan-conj 19.96% 329 h.error: 74.77% 246 orphan-nmod 12.38% 204 h.error: 47.55% 97 conj-orphan 8.56% 141 h.error: 73.05% 103 orphan-obl 6.37% 105 h.error: 67.62% 71 orphan-advmod 5.95% 98 h.error: 63.27% 62
UFAL-UDPipe-1-2	orphan-conj 17.12% 288 h.error: 81.94% 236 orphan-nmod 12.31% 207 h.error: 44.93% 93 conj-orphan 8.62% 145 h.error: 73.79% 107 orphan-obl 5.77% 97 h.error: 59.79% 58 orphan-advmod 4.88% 82 h.error: 56.1% 46
UParse	orphan-conj 14.96% 243 h.error: 81.48% 198 orphan-nmod 12.5% 203 h.error: 50.74% 103 conj-orphan 9.11% 148 h.error: 59.46% 88 orphan-obl 5.91% 96 h.error: 69.79% 67 orphan-advmod 4.37% 71 h.error: 59.15% 42

Table 2: Erroneously predicted or missed `orphan` labels and their frequencies

and Moor, Christophe and Merlo, Paola and Henderson, James and Wang, Haozhou and Ji, Tao and Wu, Yuanbin and Lan, Man and de la Clergerie, Eric and Sagot, Benoît and Seddah, Djamé and More, Amir and Tsarfaty, Reut and Kanayama, Hiroshi and Muraoka, Masayasu and Yoshikawa, Katsumasa and Garcia, Marcos and Gamallo, Pablo. (2017). *CoNLL 2017 Shared Task System Outputs*.