# Online Service for Polish Dependency Parsing and Results Visualisation

Alina Wróblewska and Piotr Sikora

Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland
alina@ipipan.waw.pl,piotr.sikora@student.uw.edu.pl

**Abstract.** The paper presents a new online service for the dependency parsing of Polish. Given raw text as input, the service processes it and visualises output dependency trees. The service applies the parsing system – MaltParser – with a parsing model for Polish trained on the Polish Dependency Bank, and some additional publicly available tools.

**Keywords:** dependency parsing, Polish Dependency Bank, visualisation, BRAT, dependency parsing service

## 1 Introduction

Several language processing tasks, such as machine translation, question answering or information extraction, may be successfully supported by dependency parsing. A dependency-based syntactic representation transparently encodes the predicate-argument structure of a sentence, which seems to be essential to generate a new sentence or extract relevant information. That is why dependency parsing has become increasingly important in recent years (e.g., CoNLL 2006 [2] and CoNLL 2007 [10]).

Except for grammar-based dependency parsers, the manual creation of which is very time-consuming and expensive, different data-driven approaches for dependency parsing have been proposed. The best parsing results are achieved with supervised techniques so far. Supervised dependency parsers trained on correctly annotated data may achieve high parsing performance, even for languages with relatively free word order, such as Czech [10], Russian [9] or Bulgarian [10].

This paper deals with the dependency parsing of Polish, which is another language with free word order and rich morphology. We present a new online service that processes raw text, annotates its sentences with dependency trees and visualises results. The service applies the parsing system – MaltParser [11]– with a parsing model for Polish trained on the Polish Dependency Bank [18], and some additional publicly available tools.

The paper is structured as follows. Section 2 introduces publicly available achievements in the Polish dependency parsing. Section 3 describes the dependency parsing module and the visualisation application. Section 4 concludes with some ideas for future research.

## 2 Dependency Parsing of Polish

The first Polish dependency parser was developed by Obrębski [12].[1] This is a rule-based parser that was tested against a small artificial test set and no wide-coverage grammar seems to accompany the work. Regarding the idea of training data-driven dependency parsers for Polish, some preliminary experiments are presented in [19]. Results of these experiments show that it is possible to train dependency parsing models for Polish with publicly available parser-generation systems: *MaltParser* [11] and *MSTParser* [6]. The presented dependency parsing models have been trained on dependency trees from the Polish Dependency Bank (Pol. *Składnica Zależnościowa* [18]).

The Polish Dependency Bank consists of 8227 syntactically annotated sentences,[2] which have been semi-automatically derived from trees available in the Polish constituency treebank (Pol. *Składnica Frazowa* [17]). Dependency structures meet properties of valid dependency trees [5] and are labelled with grammatical functions defined for Polish.[3] Any dependency structure is annotated as a tree with nodes corresponding to tokens in a sentence and arcs representing dependency relations between two tokens. One of the related tokens is the governor of a dependency relation, while the other one is its dependent. An example of a Polish dependency tree is given in Figure 1.

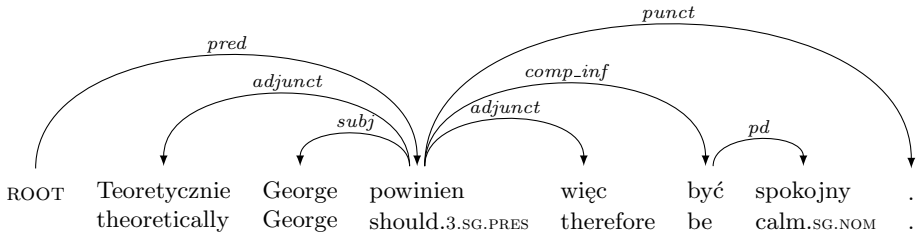


**Fig. 1.** Dependency tree of the Polish sentence *Teoretycznie George powinine więc być spokojny.* (Eng. 'So theoretically, George should not worry.').

A part of automatically converted dependency trees have been manually corrected by a linguist experienced in the Polish syntax. The first 1,000 trees were thoroughly checked for errors, and other trees were skimmed through focusing on potentially reoccurring errors.[4]

[1] This dependency parser seems to be not publicly available.

[2] In comparison to dependency treebanks for other languages, e.g., for Czech *PDT* [4], used to train dependency parsers, the size of the Polish treebank seems to be relatively small and probably not sufficient to train high-coverage parsing models. Despite this, since we do not have any larger set of training data yet, we will use the Polish Dependency Bank for the purposes of the current work.

[3] Description of Polish dependency relation types: `zil.ipipan.waw.pl/FunkcjeZaleznosciowe`.

[4] The partially corrected Polish Dependency Bank in CoNLL format (Składnica-zależnościowa-0.5.conll.gz) is available on `http://zil.ipipan.waw.pl/Składnica`.

Drawing on findings in training dependency parsing models presented in [19], we repeat one of the described experiments. A Polish dependency parser is trained on the entire partially corrected Polish Dependency Bank using *MaltParser* parsing system [11]. The transition-based dependency parser uses a deterministic parsing algorithm[5] that builds a dependency structure of an input sentence based on transitions (shift-reduce actions) predicted by a classifier. The classifier trained with the LIBLINEAR library [3] learns to predict the next transition given training data and the parse history. The feature model is defined in terms of token attributes, i.e., word form (FORM), part-of-speech tag (POS), morphological features (FEATS), and lemma (LEMMA) available in input data, or dependency types (DEPREL) extracted from partially built dependency graphs and updated during parsing.

Polish *MaltParser* trained on the entire Polish Dependency Bank is evaluated against a set of 50 manually annotated sentences (17.8 tokens/sentence) taken from Polish magazines. The performance of the Polish MaltParser is evaluated with two standard metrics: *labelled attachment score* (LAS)[6] and *unlabelled attachment score* (UAS).[7] Polish *MaltParser* tested against the set of 50 manually annotated sentences achieves 64.8% LAS/71.3% UAS.[8]

## 3 Parsing and Visualisation

Driven by the idea of making results of Polish *MaltParser* publicly available, we have prepared an online platform that allows any Internet user to input raw text that will be tagged, dependency parsed and displayed in a way convenient for perception and evaluation. To that end, a number of tools has been employed:

- *Multiservice* [13] [14],[9] a Web Service integration platform for Polish linguistic resources,
- *Pantera* [1], a morpho-syntactic rule-based Brill tagger of Polish,
- BRAT rapid annotation tool [16], an online environment for collaborative text annotation.

From the technical point of view, the dependency service is implemented as a component of *Multiservice* system, which provides a framework for different

[5] Since Polish dependency trees may be non-projective, the built-in `stackeager` parsing algorithm [8] is used in the experiment.

[6] Labelled attachment score (LAS) – the percentage of tokens that are assigned a correct head and a correct dependency type.

[7] Unlabelled attachment score (UAS) – the percentage of tokens that are assigned a correct head.

[8] The test sentences are much longer and more complex than sentences in the Polish Dependency Bank. [19] report that the Polish *MaltParser* results are significantly better – 84.7% LAS and 90.5% UAS, if the parser is evaluated against unseen sentences from the Polish Dependency Bank, which seem to be much simpler.

[9] *Multiservice* with an integrated dependency parsing module is publicly available on `glass.ipipan.waw.pl/multiservice`.

NLP-tools to work together. *Multiservice* uses Apache Thrift [15] as a basis to setup communication between various daemons[10] wrapping previously offline resources in a flexible chain of linguistic tools. As a result, the application can automatically go through all the steps from raw text to a desired output (e.g., dependency trees). Apart from providing access to linguistic resources via network, daemons have to translate the incoming Thrift data into an input format required by the wrapped tool and then do the opposite conversion for output. In order to connect dependency parsing to the *Multiservice* platform, *DependencyParserService* has been created. This service uses the *MaltParser* system with a pre-trained model to parse input sentences. Moreover, *DependencyParserService* reformats incoming and outgoing data between Thrift objects and CoNLL format [2]. The structure of Thrift object had also been modified to make it capable of containing dependency relations.

Since *MaltParser* requires sentences to be morpho-syntactically tagged before parsing, *Pantera* tagger integrated into the *Multiservice* platform fulfils this requirement. Since *Multiservice* is designed to use one communication protocol for all services, dependency parsing relies on this specific tagger only insofar as it is the only one tagger already integrated into *Multiservice*. Should another tagger become available, it can replace the current one or the user may be allowed to freely choose the source from which *DependencyParserService* would commence.

Finally, once a network based solution for the dependency parsing of Polish was ready, the only remaining task was to visualise dependency trees. At the beginning we intended to use *MaltEval*[11] [7] to visualise dependency trees. However, bringing the Java-based software to web environment wasn't straightforward, so we did not integrate it into the service. Meanwhile, we became aware of the recursively acronymised BRAT [16], which among other things has been used to visualise CoNLL-X Multilingual Dependency Parsing task data [2]. The BRAT tool seems to be flexible enough to be seamlessly embedded into the *Multiservice*'s Django-based web-page server. The final result is a simple web-service bringing visualised results of the dependency parsing of Polish in the form of user-friendly readable trees (see example in Figure 2).

## 4 Conclusions and Future Work

We have presented the online Polish dependency parsing service. The service processes raw input text, annotates sentences with dependency trees, and then visualises results. Integration of different NLP-components and the visualisation application was not a trivial task, but made it possible to present the functionality of the Polish dependency parsing to a wider audience. The dependency parsing service is freely available for research and educational purposes.

So far the online platform only enables dependency parsing of Polish sentences using *MaltParser* with one preloaded parsing model. We suppose it could

[10] Daemons are computer programs running as background processes, e.g., on a server.
[11] an evaluation tool for dependency parsers developed by the authors of *MaltParser*.

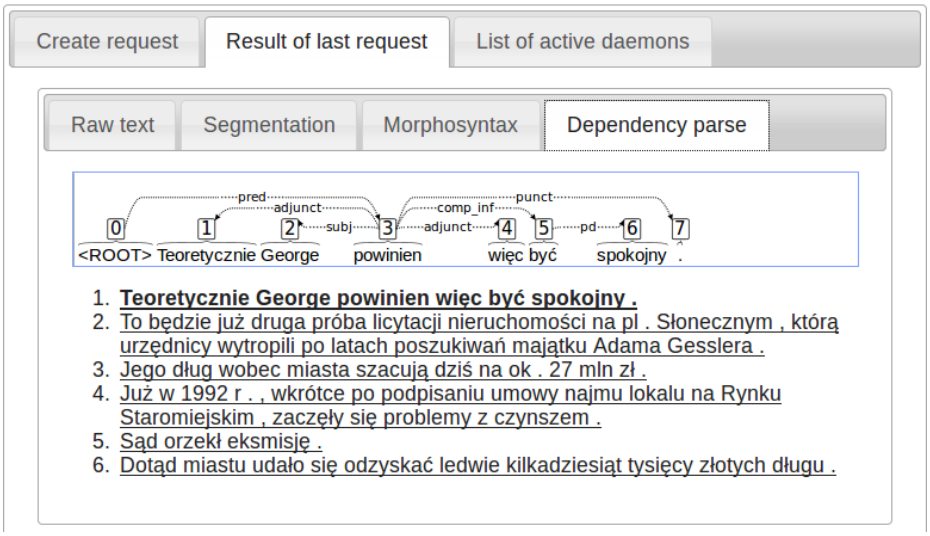


**Fig. 2.** Screenshot of the visualisation of the dependency tree previously presented in Figure 1.

be useful to be able to compare trees produced by different dependency parsing models or even different parsers. This requires expanding capabilities of the dependency parser service to allow simultaneous processing of a sentence by multiple dependency parsers. Furthermore, in order to let users see differences between multiple dependency trees at a glance, custom modificantions to BRAT visualisations need to be provided. Hence, we also plan to train Polish parsing models that will cover more linguistic facts than the current model.

Another possible path to explore is to tap BRAT's annotation capabilities and allow users to send feedback on generated results, which would lessen the workload required to train better models of Polish dependency parsing thanks to the online platform's ease of use.

**Acknowledgements.** This research is supported by the POIG.01.01.02-14-013/09 project which is co-financed by the European Union under the European Regional Development Fund.

## References

1. Acedański, S.: A Morphosyntactic Brill Tagger for Inflectional Languages. In: Advances in Natural Language Processing. LNCS, vol. 6233, pp. 3–14. Springer-Verlag, Heidelberg (2010)
2. Buchholz, S., Marsi, E.: CoNLL-X shared task on Multilingual Dependency Parsing. In: Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL-X '06). pp. 149–164 (2006)

3. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A Library for Large Linear Classification. Journal of Machine Learning Research 9, 1871–1874 (2008)
4. Hajič, J., Vidová-Hladká, B., Pajas, P.: The Prague Dependency Treebank: Annotation Structure and Support. In: Proceedings of the IRCS Workshop on Linguistic Databases. pp. 105–114 (2001)
5. Kübler, S., McDonald, R.T., Nivre, J.: Dependency Parsing. Synthesis Lectures on Human Language Technologies, Morgan & Claypool Publishers (2009)
6. McDonald, R., Pereira, F., Ribarov, K., Hajič, J.: Non-projective Dependency Parsing using Spanning Tree Algorithms. In: Proceedings of Human Language Technology Conferences and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP). pp. 523–530 (2005)
7. Nilsson, J., Nivre, J.: MaltEval: An Evaluation and Visualization Tool for Dependency Parsing. In: Proceedings of the 6th International Language Resources and Evaluation (LREC-2008). pp. 161–166 (2008)
8. Nivre, J.: Non-projective Dependency Parsing in Expected Linear Time. In: Proceedings of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. pp. 351–359 (2009)
9. Nivre, J., Boguslavsky, I.M., Iomdin, L.L.: Parsing the SynTagRus treebank of Russian. In: Proceedings of the 22nd International Conference on Computational Linguistics (COLING '08). vol. 1, pp. 641–648 (2008)
10. Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., Yuret, D.: The CoNLL 2007 Shared Task on Dependency Parsing. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007. pp. 915–932 (2007)
11. Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., Marsi, E.: MaltParser: A language-independent system for data-driven dependency parsing. Natural Language Engineering 13(2), 95–135 (2007)
12. Obrębski, T.: MTT-compatible computationally effective surface-syntactic parser. In: Proceedings of the 1st International Conference on Meaning-Text Theory. pp. 259–268 (2003)
13. Ogrodniczuk, M., Lenart, M.: Multipurpose Linguistic Web Service for Polish. In: Proceedings of the Language Technology for a Multilingual Europe workshop at the German Society for Computational Linguistics and Language Technology Conference (GSCL 2011). Hamburg, Germany (2011)
14. Ogrodniczuk, M., Lenart, M.: Web Service integration platform for Polish linguistic resources. In: Proceedings of the Eighth International Conference on Language Resources and Evaluation, (LREC-2012). pp. 1164–1168 (2012)
15. Slee, M., Agarwal, A., Kwiatkowski, M.: Thrift: Scalable Cross-Language Services Implementation. Tech. rep., Facebook, 156 University Ave, Palo Alto, USA (2007)
16. Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou, S., Tsujii, J.: BRAT: a Web-based Tool for NLP-Assisted Text Annotation. In: Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics. pp. 102–107 (2012)
17. Świdziński, M., Woliński, M.: Towards a Bank of Constituent Parse Trees for Polish. In: Proceedings of the 13th International Conference on Text, Speech and Dialogue. LNAI, vol. 6231, pp. 197–204. Springer-Verlag, Heidelberg (2010)
18. Wróblewska, A.: Polish Dependency Bank. Linguistic Issues in Language Technology 7(1) (2012)
19. Wróblewska, A., Woliński, M.: Preliminary Experiments in Polish Dependency Parsing. In: Security and Intelligent Information Systems: International Joint Conference (SIIS 2011). LNCS, vol. 7053, pp. 279–292. Springer-Verlag (2012)