i-Parser: Interactive Parser Development Kit for Natural Language Processing

Gaku Morio,[1*] **Hiroaki Ozaki,**[1*] **Yuta Koreeda,**[2] **Terufumi Morishita,** [1] **Toshinori Miyoshi** [1]
[1]Research and Development Group, Hitachi, Ltd., Kokubunji, Tokyo, Japan
[2]Research and Development Group, Hitachi America, Ltd., Santa Clara, CA, USA
gaku.morio.vn@hitachi.com, hiroaki.ozaki.yu@hitachi.com, yuta.koreeda@hal.hitachi.com,
terufumi.morishita.wp@hitachi.com, toshinori.miyoshi.pd@hitachi.com

Abstract

This demonstration paper presents i-Parser, a novel development kit that produces high-performance semantic parsers. i-Parser converts training graphs into sequences written in a context-free language, then our proposed model learns to generate the sequences. With interactive configuration and visualization, users can easily build their own parsers. Benchmark results of i-Parser showed high performances of various parsing tasks in natural language processing.

Introduction

Parsing text into a graph is key technology for dialog systems, document analysis, and so on. In fact, there exist abundant graph representations to deal with different needs. For example, Abstract Meaning Representation (AMR; Banarescu et al. (2013)) is designed to represent high-level sentence meaning as a graph, and it is useful for dialog systems. Elementary Dependency Structures (EDS; Oepen and Lønning (2006)) based on English head-driven phrase structure grammar requires node-to-input anchoring. Math word problems (Wang, Liu, and Shi 2017) are aimed at generating a computation tree from a natural language math problem. However, most of these graphs have been parsed by a task-specific system due to differences in graph forms.

In this paper, we present a novel development kit, i-Parser, that allows developers to easily implement desirable parsers. Figure 1 shows an overview of i-Parser. Users only need to provide training data and graph specifications. The system interactively generates a setup file by asking users graph outline questions such as *Tree or DAG?* With the setup file, i-Parser Converter universally transforms a given graph into our proposed Plain Graph Notation (PGN) sequence. Finally, i-Parser Model that leverages Transformers (Vaswani et al. 2017) is trained to generate PGN sequences.

i-Parser supports many graph variants without any task-specific implementations (e.g., transition design). i-Parser also provides out-of-the-box visualization and hyperparameter tuning tools. This allows developers to benefit from faster development cycles.

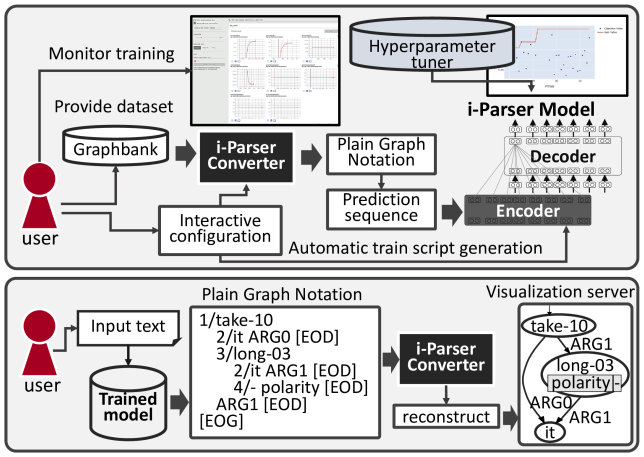

Figure 1: An overview of i-Parser. Top (training): We convert a graph into a PGN sequence, then we train the Transformer encoder-decoder. Bottom (inference): We predict the PGN for the input text to obtain an output graph.

Results of semantic graph parsing and math word problems show that i-Parser achieves high performances. [1]

System

i-Parser Converter This tool converts a graph into a PGN sequence to represent graph variants in a unified text-based notation. For example, an AMR graph (Figure 1; bottom right) can be represented as PGN, as shown in Figure 1 (bottom middle). The i-Parser Converter can also reconstruct a graph from a predicted PGN sequence.

The input graph can be in CoNLL-U or MRP (i.e., JSON) formats (Oepen et al. 2019). We also allow command options as follows:

```
converter.py --read mrp --write lark \
--write_rule {RULE} --head_is_source \
--encode_processor embed input > output
```

where `--read` specifies an input codec and `--encode_processor` specifies the PGN decora-

[*]Contributed equally.

[1]Detailed model description is presented at CoNLL 2020 shared task (Ozaki et al. 2020). This paper mainly presents the user interface, training system and visualization of i-Parser.

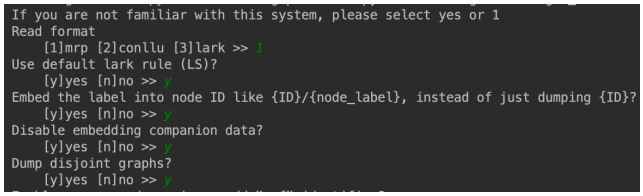

Figure 2: An example of an interactive configuration.

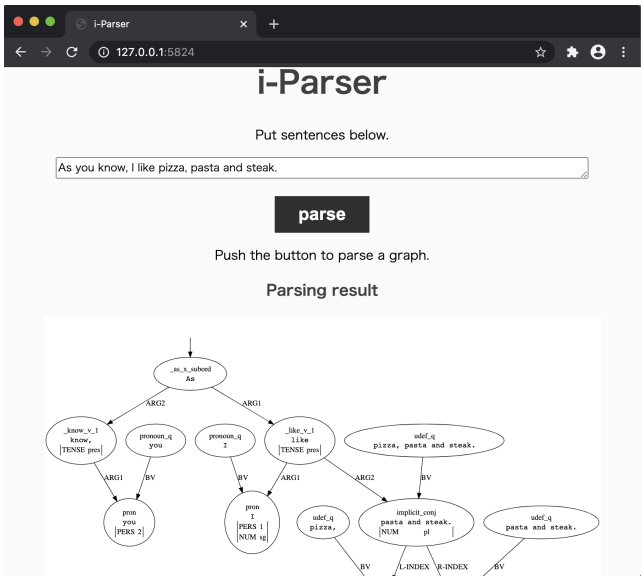

Figure 3: Visualization system

tors. We implemented this module with Lark parser (https://github.com/lark-parser/lark).

Training Procedure To train a model, we only need to edit a HOCON (https://github.com/chimpler/pyhocon) configuration file that specifies the number of hidden layers, learning rates, dropout ratio, number of epochs and so on. The i-Parser Model is based on encoder-decoder architecture in which a pre-trained model such as BERT (Devlin et al. 2019) or RoBERTa (Liu et al. 2019) is employed as an encoder and a Transformer is applied as a decoder. The i-Parser Model also utilizes biaffine attentions (Dozat and Manning 2018) to solve anchors and reentrancy edges. The i-Parser Model is implemented with PyTorch (Paszke et al. 2019) and Hugging Face Transformers (Wolf et al. 2020). i-Parser has other useful training tools:

- Monitoring training with Tensorboard (https://www.tensorflow.org/tensorboard)
- Automatic model selection based on validation scores
- Multi-GPU and multi-node training based on Horovod (Sergeev and Balso 2018) that enables fast training

Interactive Configuration i-Parser provides an interactive configuration tool (Figure 2) that enables beginners to implement their own parsers easily. After the users answer 10–20 simple questions, i-Parser constructs a new parser

Figure 4: An example of hyperparameter tuning setup.

by generating configurations of the i-Parser Converter and training scripts of the i-Parser Model. Example questions include *Use default postprocessor?*

Visualization i-Parser provides a visualization server (Figure 3) that offers developers a way to debug their models. When given input sentence(s), the system parses and draws a graph in a web browser interface.

Hyperparameter Tuner i-Parser provides a hyperparameter tuning system powered by Optuna (Akiba et al. 2019) that allows developers to build a strong parser without manual configurations. This tuning system parses a JSON file (Figure 4) in which tunable parameters are registered as environmental variables, applying Bayesian optimization.

Benchmark

We developed nine parsers using our i-Parser prototype and achieved the top performance in the CoNLL 2020 shared task (Oepen et al. 2020).We obtained F-scores of 81.54 for AMR, 93.56 for EDS, 88.73 for PTG (Hajič et al. 2012), 80.44 for Chinese AMR (Li et al. 2016), and 93.36 for German DRG (Bos et al. 2017), which are ranked the first amongst all the teams. Furthermore, we applied i-Parser to math word problems (Zhang et al. 2020b,a) where the parser had to generated an arithmetic expression tree. i-Parser achieved 77.5% solution accuracy with Math23K.These results show the effectiveness of our system.

Online Demonstration Scenario

In the upcoming AAAI conference, we will show how i-Parser works. We will present its interactive configuration and how it monitors training. The visualization system will also be a key part of our demonstration. In the visualization, we will showcase many state-of-the-art parsing results.

Conclusion

We presented a novel parser development kit, i-Parser. The interactive system allows developing new parsers with an easy configuration. Our system can contribute to extending parsing applications in the natural language processing field.

Acknowledgments

We appreciate Prof. Dr. Naoaki Okazaki at Tokyo Institute of Technology for his helpful comments. Computational resource of AI Bridging Cloud Infrastructure (ABCI) provided by the National Institute of Advanced Industrial Science and Technology (AIST) was used.

References

Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; and Koyama, M. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, 2623–2631. New York, NY, USA: ACM. ISBN 978-1-4503-6201-6.

Banarescu, L.; Bonial, C.; Cai, S.; Georgescu, M.; Griffitt, K.; Hermjakob, U.; Knight, K.; Koehn, P.; Palmer, M.; and Schneider, N. 2013. Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*.

Bos, J.; Basile, V.; Evang, K.; Venhuizen, N.; and Bjerva, J. 2017. *The Groningen Meaning Bank*, 463–496. Springer. ISBN 978-94-024-0879-9.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. 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)*, 4171–4186. Minneapolis, Minnesota: Association for Computational Linguistics.

Dozat, T.; and Manning, C. D. 2018. Simpler but More Accurate Semantic Dependency Parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Hajič, J.; Hajičová, E.; Panevová, J.; Sgall, P.; Bojar, O.; Cinková, S.; Fučíková, E.; Mikulová, M.; Pajas, P.; Popelka, J.; Semecký, J.; Šindlerová, J.; Štěpánek, J.; Toman, J.; Urešová, Z.; and Žabokrtský, Z. 2012. Announcing Prague Czech-English Dependency Treebank 2.0. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*.

Li, B.; Wen, Y.; Qu, W.; Bu, L.; and Xue, N. 2016. Annotating the Little Prince with Chinese AMRs. In *Proceedings of the 10th Linguistic Annotation Workshop held in conjunction with ACL 2016 (LAW-X 2016)*, 7–15. Berlin, Germany: Association for Computational Linguistics.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692* .

Oepen, S.; Abend, O.; Abzianidze, L.; Bos, J.; Hajič, J.; Hershcovich, D.; Li, B.; O'Gorman, T.; Xue, N.; and Zeman, D. 2020. MRP 2020: The Second Shared Task on Cross-Framework and Cross-Lingual Meaning Representation Parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2020 Conference on Natural Language Learning*.

Oepen, S.; Abend, O.; Hajic, J.; Hershcovich, D.; Kuhlmann, M.; O'Gorman, T.; and Xue, N., eds. 2019. *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*. Hong Kong: Association for Computational Linguistics.

Oepen, S.; and Lønning, J. T. 2006. Discriminant-Based MRS Banking. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*.

Ozaki, H.; Morio, G.; Koreeda, Y.; Morishita, T.; and Miyoshi, T. 2020. Hitachi at MRP 2020: Text-to-Graph-Notation Transducer. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2020 Conference on Natural Language Learning*.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Sergeev, A.; and Balso, M. D. 2018. Horovod: fast and easy distributed deep learning in TensorFlow. *arXiv preprint arXiv:1802.05799* .

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. u.; and Polosukhin, I. 2017. Attention is All you Need. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30*, 5998–6008. Curran Associates, Inc.

Wang, Y.; Liu, X.; and Shi, S. 2017. Deep Neural Solver for Math Word Problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 845–854. Copenhagen, Denmark: Association for Computational Linguistics.

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; Davison, J.; Shleifer, S.; von Platen, P.; Ma, C.; Jernite, Y.; Plu, J.; Xu, C.; Le Scao, T.; Gugger, S.; Drame, M.; Lhoest, Q.; and Rush, A. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 38–45. Online: Association for Computational Linguistics.

Zhang, J.; Lee, R. K.-W.; Lim, E.-P.; Qin, W.; Wang, L.; Shao, J.; and Sun, Q. 2020a. Teacher-Student Networks with Multiple Decoders for Solving Math Word Problem. In Bessiere, C., ed., *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, 4011–4017. International Joint Conferences on Artificial Intelligence Organization. Main track.

Zhang, J.; Wang, L.; Lee, R. K.-W.; Bin, Y.; Wang, Y.; Shao, J.; and Lim, E.-P. 2020b. Graph-to-Tree Learning for Solving Math Word Problems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 3928–3937. Online: Association for Computational Linguistics.