MWPToolkit: An Open-Source Framework for Deep Learning-Based Math Word Problem Solvers

Yihuai Lan[1]*, Lei Wang[2]*, Qiyuan Zhang[2]*, Yunshi Lan[3]†, Bing Tian Dai[2], Yan Wang[4], Dongxiang Zhang[5], Ee-Peng Lim[2]

[1]Xihua University
[2]Singapore Management University
[3]East China Normal University
[4]Tencent AI Lab
[5]Zhejiang University
{lei.wang.2019, yslan.2015}@phdcs.smu.edu.sg, {btdai, eplim}@smu.edu.sg
{yifan2250, qiyuanzhang97}@gmail.com , bradenwang@tencent.com , zhangdongxiang@zju.edu.cn

Abstract

While Math Word Problem (MWP) solving has emerged as a popular field of study and made great progress in recent years, most existing methods are benchmarked solely on one or two datasets and implemented with different configurations. In this paper, we introduce the first open-source library for solving MWPs called MWPToolkit, which provides a unified, comprehensive, and extensible framework for the research purpose. Specifically, we deploy 17 deep learning-based MWP solvers and 6 MWP datasets in our toolkit. These MWP solvers are advanced models for MWP solving, covering the categories of Seq2seq, Seq2Tree, Graph2Tree, and Pre-trained Language Models. And these MWP datasets are popular datasets that are commonly used as benchmarks in existing work. Our toolkit is featured with highly modularized and reusable components, which can help researchers quickly get started and develop their own models. We have released the code and documentation of MWPToolkit in https://github.com/LYH-YF/MWPToolkit.

Introduction

Math Word Problem (MWP) solving, the task of automatically answering a question by understanding the textual description of a “real-life” scenario, has attracted increasing attention in NLP community (Zhang et al. 2020a). In recent years, there has been continuous emergence of new datasets (Kushman et al. 2014; Koncel-Kedziorski et al. 2016; Upadhyay and Chang 2017; Wang, Liu, and Shi 2017; Miao, Liang, and Su 2020; Qin et al. 2020; Patel, Bhattamishra, and Goyal 2021) and deep learning-based solvers (Wang, Liu, and Shi 2017; Wang et al. 2018; Chiang and Chen 2019; Liu et al. 2019; Xie and Sun 2019; Zhang et al. 2020b,c; Shen and Jin 2020; Kim et al. 2020) pushing forward the progress of this research direction. However, most existing solvers are simply evaluated on one or two datasets with varying configurations (e.g., different train-test splitting or k-fold cross-validation), which makes it unfair to compare the different solvers directly. Moreover, it is difficult and time-consuming to reimplement these solvers as baseline methods for result comparison in future studies, because previous methods are usually developed under individual systems and have a lack of compatibility. As the above issues seriously hinder MWP solving tasks from further progressing, it is imperative to develop a unified and extensible framework.

In this paper, we present MWPToolkit, the first open-source framework for deep learning-based MWP solvers, which has the following three characteristics. (1) **Unification and Modularization**: We decouple solvers with different model architectures into highly modularized, reusable components and integrate them in a unified framework. It is convenient for researchers to study MWP solvers at a conceptual level and compare different solvers fairly. (2) **Comprehensiveness and Standardization**: MWPToolkit has deployed the popular benchmark datasets and models for MWP solving, covering Seq2Seq, Seq2Tree, Graph2Tree, and Pre-trained Language Models. Moreover, some tricks used in general NLP tasks are integrated. For example, algorithms for hyper-parameter tuning are implemented to search for the hyper-parameters efficiently and thus boost the performance. Since all solvers can be implemented with the same experimental configuration, the evaluation of different models is standardized. (3) **Extensibility and Usability**: MWPToolkit provides user-friendly interfaces for various functions or modules. The components in the framework are modeled as exchangeable modules. Researchers can try different combinations of modules by simply changing the configuration file or command lines. They can also easily develop their own models by replacing or extending corresponding modules with their proposed ones.

Framework Description

As depicted in Figure 1, MWPToolkit is a unified framework for MWP solvers that includes several core components. The config component plays a preliminary role in the framework, which aims at setting up the interaction between

[*] Equal contribution.

[†] Corresponding author.

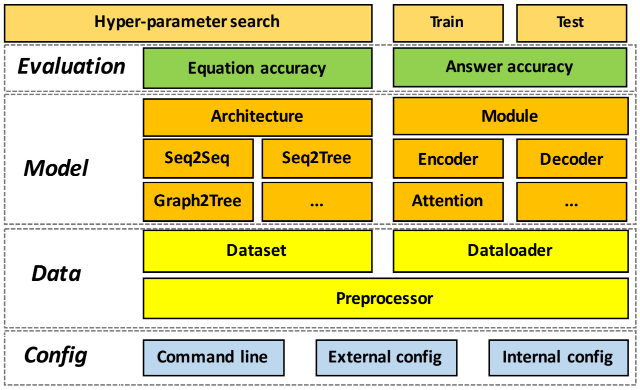

Figure 1: The overall framework of `MWPToolkit`.

the system and users. Built upon it, there are data, model and evaluation components taking charge of processing data, allocating models and evaluating the performance, respectively. These components are further integrated with hyper-parameter search, training and testing. Next, we introduce each component in detail.

Config component serves as the core human-system interaction component in which the developer can specify experimental configurations. The configuration part of our framework consists of *Command lines*, *External config*, and *Internal config*. The default configuration is defined in the internal configuration file. Users can flexibly and simply use the command lines to modify the major settings of the experiment. They can also have more customized settings with external configuration files, which is beneficial for the duplication of research results.

Data component is designed with the principle that we want to handle different MWP datasets with a unified paradigm. Therefore, we define a general data flow that gets the raw data well prepared as the input for models, that is, raw data $\mapsto$ *Preprocessor* $\mapsto$ *Dataset* $\mapsto$ *Dataloader* $\mapsto$ processed data. In *Preprocessor*, tools and functions (e.g., number detector and converter) are utilized to transform the datasets with different data structures into a unified data class. Then, *Dataset* takes charge of converting the data class into dataset class. Specifically, it splits data into train, validation and test sets, processes each question via word tokenization, transfers each solution into the required format (i.e., single equation, multiple equations), and builds a vocabulary. After that, *Dataloader* is responsible for organizing the data stream, which chunks the data into batches and converts them into tensors. Finally, these tensors can be fed into the model immediately. We have involved 6 MWP datasets in our framework, 4 of which have the single equation as the generation target and 2 of which have the multiple equations as the generation targets.

Model component aims at implementing deep learning-based models with highly modularized, reusable components, which can not only benefit the quick understanding towards the models but also facilitate the fair comparison and recombination of different deep learning modules. Models are summarized into *Seq2seq*, *Seq2Tree*, *Graph2Tree*, and *Pre-trained Models* categories based on their high-level architectures. During implementation, we decompose the models into a set of widely-used neural network modules such as *Embedder*, *Encoder*, *Decoder*, *Attention*, and *Graph*. Each of them has fine-grained subcategories. For example, the encoder can be specified as GRU, LSTM, CNN and so on. The combinations of them can form different MWP models. In total, we have implemented 28 reusable functional modules in our library, which can deploy 17 deep learning-based MWP solvers. It is worth noting that for all the deployed solvers, their results are reliable. We set the same hyper-parameters as the ones in the original papers and ensure the reimplemented result should be approximate to the reported result.

Evaluation component is a component that evaluates solvers with standard measurement criteria, namely *Equation accuracy* and *Answer accuracy*. Specifically, our evaluator generates the predicted equation sequences and obtains predicted answers by executing the predicted equation. The accuracy score is measured by conducting exact match between the predicted results and the ground-truth results for both equations and answers.

On top of the above components, we implement training and testing, where two options are provided. One is to follow the standard train-test splitting if the splitting data is given in the original dataset. Another is to conduct k-fold cross-validation. To improve the performance, we also implement a series of hyper-parameter search strategies, such as beam search, greedy search, sampling strategy.

Related Work

In NLP community, there have been a number of toolkits that managed to summarize the existing methods and establish a unified framework for a certain task. TextBox (Li et al. 2021), an open-source library for text generation, provides a comprehensive and efficient framework for reproducing and developing text generation algorithms. Liu et al. (2021) has released ExplainaBoard, which is a unified platform to evaluate interpretable, interactive and reliable capabilities of NLP systems. Photon (Zeng et al. 2020) and DIALOGPT (Zhang et al. 2020d) are two comprehensive systems for cross-domain text-to-SQL and conversational response generation tasks, respectively. To our best knowledge, there is no such a unified and comprehensive framework for MWP solving tasks. Therefore, we release `MWPToolkit`, which includes a considerable number of benchmark datasets and deep learning-based solvers.

Conclusion

This paper presented a unified, modularized, and extensible framework for deep learning-based MWP solvers, called `MWPToolkit`. We introduced the overall architecture as well as the core components. `MWPToolkit` is expected to establish a benchmark framework for MWP solving task, achieve reproducibility of existing models, and speed up the development of new methods. In the future, we will consider adding more functionalities, such as result visualization and training under weak supervision.

Acknowledgments

This work is partially supported by National Natural Science Foundation of China under Grant No. 61977026 and U1811264. Yunshi Lan is the corresponding author.

References

Chiang, T.-R.; and Chen, Y.-N. 2019. Semantically-Aligned Equation Generation for Solving and Reasoning Math Word Problems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2656–2668.

Kim, B.; Ki, K. S.; Lee, D.; and Gweon, G. 2020. Point to the Expression: Solving Algebraic Word Problems using the Expression-Pointer Transformer Model. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Koncel-Kedziorski, R.; Roy, S.; Amini, A.; Kushman, N.; and Hajishirzi, H. 2016. MAWPS: A Math Word Problem Repository. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1152–1157.

Kushman, N.; Artzi, Y.; Zettlemoyer, L.; and Barzilay, R. 2014. Learning to Automatically Solve Algebra Word Problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 271–281.

Li, J.; Tang, T.; He, G.; Jiang, J.; Hu, X.; Xie, P.; Chen, Z.; Yu, Z.; Zhao, W. X.; and Wen, J.-R. 2021. TextBox: A Unified, Modularized, and Extensible Framework for Text Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, 30–39.

Liu, P.; Fu, J.; Xiao, Y.; Yuan, W.; Chang, S.; Dai, J.; Liu, Y.; Ye, Z.; and Neubig, G. 2021. ExplainaBoard: An Explainable Leaderboard for NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, 280–289.

Liu, Q.; Guan, W.; Li, S.; and Kawahara, D. 2019. Tree-structured Decoding for Solving Math Word Problems. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 2370–2379.

Miao, S.-y.; Liang, C.-C.; and Su, K.-Y. 2020. A Diverse Corpus for Evaluating and Developing English Math Word Problem Solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 975–984.

Patel, A.; Bhattamishra, S.; and Goyal, N. 2021. Are NLP Models really able to Solve Simple Math Word Problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2080–2094.

Qin, J.; Lin, L.; Liang, X.; Zhang, R.; and Lin, L. 2020. Semantically-Aligned Universal Tree-Structured Solver for Math Word Problems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 3780–3789.

Shen, Y.; and Jin, C. 2020. Solving Math Word Problems with Multi-Encoders and Multi-Decoders. In *Proceedings of the 28th International Conference on Computational Linguistics*, 2924–2934.

Upadhyay, S.; and Chang, M.-W. 2017. Annotating Derivations: A New Evaluation Strategy and Dataset for Algebra Word Problems. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 494–504.

Wang, L.; Wang, Y.; Cai, D.; Zhang, D.; and Liu, X. 2018. Translating a Math Word Problem to a Expression Tree. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 1064–1069.

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.

Xie, Z.; and Sun, S. 2019. A Goal-Driven Tree-Structured Neural Model for Math Word Problems. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, 5299–5305.

Zeng, J.; Lin, X. V.; Hoi, S. C.; Socher, R.; Xiong, C.; Lyu, M.; and King, I. 2020. Photon: A Robust Cross-Domain Text-to-SQL System. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 204–214.

Zhang, D.; Wang, L.; Zhang, L.; Dai, B. T.; and Shen, H. T. 2020a. The Gap of Semantic Parsing: A Survey on Automatic Math Word Problem Solvers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(9): 2287–2305.

Zhang, J.; Lee, R. K.-W.; Lim, E.-P.; Qin, W.; Wang, L.; Shao, J.; and Sun, Q. 2020b. 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. Main track.

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

Zhang, Y.; Sun, S.; Galley, M.; Chen, Y.-C.; Brockett, C.; Gao, X.; Gao, J.; Liu, J.; and Dolan, B. 2020d. DIALOGPT : Large-Scale Generative Pre-training for Conversational Response Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 270–278.