System Report for CCL24-Eval Task 4: A Two-stage Generative Chinese AMR Parsing Method Based on Large Language Models

Zizhuo Shen, Yanqiu Shao✉, Wei Li
School of Information Science, Beijing Language and Culture University
Beijing 100083, China
blcushzz@gmail.com
yqshao163@163.com
liweitj47@blcu.edu.cn

Abstract

The purpose of the CAMR task is to convert natural language into a formalized semantic representation in the form of a graph structure. Due to the complexity of the AMR graph structure, traditional AMR automatic parsing methods often require the design of complex models and strategies. Thanks to the powerful generative capabilities of LLMs, adopting an autoregressive generative approach for AMR parsing has many advantages such as simple modeling and strong extensibility. To further explore the generative AMR automatic parsing technology based on LLMs, we design a two-stage AMR automatic parsing method based on LLMs in this CAMR evaluation. Specifically, we design two pipeline subtasks of alignment-aware node generation and relationship-aware node generation to reduce the difficulty of LLM understanding and generation. Additionally, to boost the system's transferability, we incorporate a retrieval-augmented strategy during both training and inference phases. The experimental results show that the method we proposed has achieved promising results in this evaluation.

1 Introduction

Semantic parsing is one of the fundamental tasks in the field of Natural Language Processing (NLP). As an important formal method of semantic parsing, Abstract Meaning Representation (AMR) (Banarescu et al., 2013) has attracted widespread attention from researchers in the NLP field. AMR represents the semantics of a sentence in the form of a single-rooted directed acyclic graph (DAG), where nodes are used to represent semantic concepts abstracted from the sentence, and edges are used to represent the semantic relations between concepts. Thanks to the development of AMR technology, researchers have also begun to try to integrate the semantic knowledge in AMR graphs into application systems such as machine translation (Song et al., 2019), text summarization (Liao et al., 2018), and event extraction (Xu et al., 2022), in order to enhance the system's understanding of semantic knowledge and improve system performance.

The data resources and automatic parsing technologies associated with AMR primarily concentrate on English. Li et al. (2021) considering the traits of Chinese, establish the annotation standards and evaluation methods fitting for Chinese AMR (CAMR), significantly advancing the progress of Chinese AMR research.

The existing CAMR automatic parsing technologies can be divided into three categories: transition-based methods (Wang et al., 2018), graph-based methods (Chen et al., 2022), and autoregressive generative methods (Bevilacqua et al., 2022). The transition-based method considers the AMR automatic parsing problem as a sequence-to-action conversion issue. This method usually relies on a complex transition system for action definition and feature extraction used for action prediction. The graph-based method treats the AMR automatic parsing problem as a graph search issue, that is, searching for the highest scoring subgraph from a directed complete graph. Due to the complexity of the AMR automatic parsing task, the graph-based method usually requires the construction of multiple models such as concept recognition and relation recognition to achieve a complete parsing process.

Proceedings of the 23rd China National Conference on Computational Linguistics, pages 154-159, Taiyuan, China, July 25 - 28, 2024.
Volume3: Evaluations

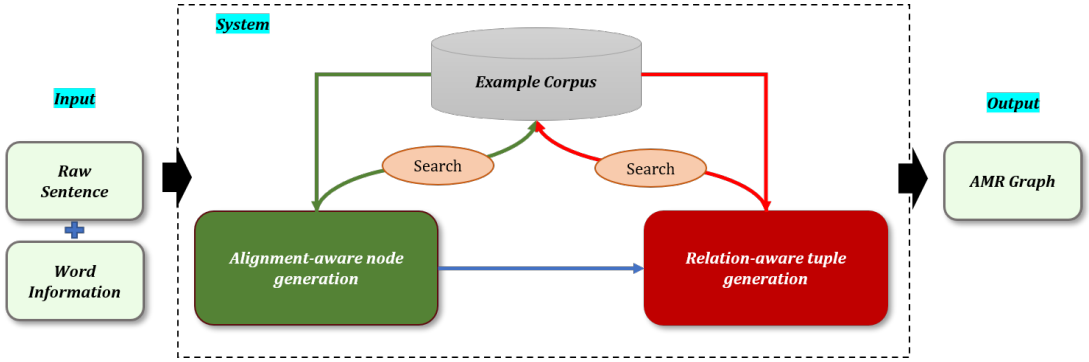

Figure 1: A Brief Flowchart of Our Method.

The autoregressive generative method considers the AMR automatic parsing problem as a sequence-to-sequence generation issue. Compared with the other two methods, the autoregressive generative method has the advantages of simple modeling and easy implementation, and has become the mainstream method for solving AMR automatic parsing problems. Recently, thanks to the powerful generation capabilities of large language models (LLMs) (Zhao et al., 2023), the autoregressive generative methods based on LLMs have also been adopted by more and more researchers and achieved competitive results (Lee et al., 2023).

The existing autoregressive generative methods based on LLMs usually adopt a strategy of generating AMR graphs in a single stage (Gao et al., 2023), such as directly generating linearized sequences of AMR graphs or generating tuples within AMR graphs. Given the complexity of the AMR graph structure, the difficulty of generating AMR graphs in a single stage is significant, which constitutes a huge challenge for the generation ability of LLMs.

In order to reduce the difficulty of generating AMR graphs by LLMs, we decompose the process of generating AMR graphs into two stages. The first stage is **alignment-aware node generation**, and the second stage is **relation-aware tuple generation**. In the first stage, we first group the nodes according to the alignment type between the nodes in the AMR graph and the words in the sentence, and then let the LLM generate different aligned type nodes in groups. In the second stage, we first group the tuples in the AMR graph according to the relation type, and then let the LLM generate tuples of different relation types in groups. Considering that this shared task requires testing the system's ability to automatically parse ancient Chinese sentences into AMR graphs in a zero-shot scenario, we adopt a **retrieval-augmented instruction tuning** strategy to fine-tune the LLM, with the aim of enhancing our system's transferability.

2 Method

Our method can be divided into three parts to descirbe: alignment-aware node generation, relation-aware tuple generation, and retrieval-augmented instruction tuning strategy. Figure 1 shows the flowchart of our method. Our system takes a sentence and the word information in the sentence as input, and after performing two pipeline tasks of alignment-aware node generation and relation-aware tuple generation, it obtains the entire AMR graph as output.

2.1 Alignment-aware Node Generation

To achieve alignment-aware node generation, we categorize the nodes into several types based on the alignment between the nodes in the AMR graph and the words in the sentence (Chen et al., 2022): **general alignment nodes**, **split alignment nodes**, **multi-word alignment nodes**, **and non-alignment nodes**. The general alignment node refers to a node in the AMR graph that can correspond to a word in the sentence. The split alignment node refers to a node in the AMR graph that can correspond to a part of a word in the sentence. The multi-word alignment node refers to a node in the AMR graph that can correspond to multiple words in the sentence. The non-alignment node refers to a node in the AMR graph that cannot correspond to any word in the sentence. Since the generation patterns of nodes with

Proceedings of the 23rd China National Conference on Computational Linguistics, pages 154-159, Taiyuan, China, July 25 - 28, 2024.
Volume3: Evaluations

Task	Input	Output

Alignment-aware Node Generation	{
	'sentence': '中国将拓宽利用外资渠道',
	'words': {'x1': '中国', 'x2': '将', 'x3': '拓宽', 'x4': '利用', 'x5': '外资', 'x6': '渠道'}
	}
		{
		'non-align': ['(x0, root, -)', '(x10, country, -)'],
		'align': ['(x1, 中国, -)', '(x2, 将, -)', '(x3, 拓宽-01, -)', '(x4, 利用-01, -)', '(x5, 外资, -)', '(x6, 渠道, -)']
		}

Relation-aware Tuple Generation	{
	'sentence': '中国将拓宽利用外资渠道',
	'nodes': {'non-align': ['(x0, root, -)', '(x10, country, -)'], 'align': ['(x1, 中国, -)', '(x2, 将, -)', '(x3, 拓宽-01, -)', '(x4, 利用-01, -)', '(x5, 外资, -)', '(x6, 渠道, -)'] }
	}
		{
		'(top, - , -)': [['(x0, root, -)', '(x3, 拓宽-01, -)']],
		'(arg0, - , -)': [['(x3, 拓宽-01, -)', '(x10, country, -)']],
		'(tense, - , -)': [['(x3, 拓宽-01, -)', '(x2, 将, -)']],
		'(arg1, - , -)': [['(x3, 拓宽-01, -)', '(x6, 渠道, -)'], ['(x4, 利用-01, -)', '(x5, 外资, -)']],
		'(name, - , -)': [['(x10, country, -)', '(x1, 中国, -)']],
		'(mod, - , -)': [['(x6, 渠道, -)', '(x4, 利用-01, -)']],
		}

Table 1: Input and Output Examples for Two Tasks.

the same alignment type are the same, the alignment-aware node generation task can help LLMs learn the generation patterns of nodes, thereby achieving better node generation effects.

The input for the node generation task is a sentence and the words in the sentence, and the output is the nodes in the AMR graph. Therefore, the node generation task can be described as follows: **Given a sentence and the words in the sentence, predict the nodes corresponding to the sentence in the AMR graph**.

For the words in the sentence, we use a dictionary to store the information of the words in the sentence, where the key of the dictionary is the index of the word and the value of the dictionary is the text of the word. For the nodes in the AMR graph, we use a ***dictionary-list*** data structure to store the grouped node information, where the key of the dictionary is the type of the node and the value of the dictionary is the list of all nodes under that type. For each node, we use a triplet to represent all the information of the node: the index of the node, the text of the node, and the index of the co-referential node. An example of the input and output for this task is given in Table 1.

2.2 Relation-aware Tuple Generation

The input for the tuple generation task is a sentence and the nodes in the AMR graph, with the output being the relational tuples in the AMR graph. Therefore, the tuple generation task can be described as follows: **Given a sentence and the nodes in the AMR graph, predict the relational tuples in the AMR graph**.

For the relational tuples in the AMR graph, we use a ***dictionary-list-list*** data structure to store the grouped tuple information. The key of the dictionary is the relational information, and we use a triplet to represent all the information of the relation: the name of the relation, the index of the aligned word of the relation, and the text of the aligned word of the relation. The value of the dictionary is a two-level nested list, where the first level list represents all the node pairs with the same relation, and the second level list represents the parent and child nodes of each node pair. An example of the input and output for this task is given in Table 1.

2.3 Retrieval-augmented Instruction Tuning Strategy

Based on our definitions of the node generation task and the tuple generation task, we can build prompts for these two tasks separately to be used for instruction tuning. The prompts for these two tasks are composed of the following three parts: task description part, example information part, and input information part. The task description part is a natural language description of the current task. The example part is an input-output pair for the current task. The input information part is the input of the sentence

to be parsed under the current task. To make the examples more representative, we use a vector retrieval model[0] to find examples for each sentence to be parsed. Specifically, we use sentences in the development set to build a vector retrieval database. For a sentence to be parsed, we vectorize it and then search for the sentence with the highest semantic similarity from the vector retrieval database by calculating the cosine similarity as the final example.

In the instruction tuning stage, we denote the entire prompt as X and the corresponding output as O. Each token in O is denoted as o_i. When calculating the loss of the language model, we only calculate the loss for the output part. The calculation formula is shown below:

$$\mathcal{L} = \sum_{i=1} -logP(o_i|X, o_{i-1}). \quad (1)$$

3 Experiments and Analysis

3.1 Experimental Setup

This shared task includes two scenarios: open testing and closed testing. We conduct experiments in the open testing scenario. The LLM we used is the Baichuan2-13B-Base [1] released by Baichuan Intelligent Technology. Due to computational resource limitations, we adopt the LoRA-based method (Hu et al., 2022) for fine-tuning the LLM. The main training parameters used during the fine-tuning process are listed in Table 2.

Parameter Name	Parameter Value
learning_rate	2e-4
lora_rank	64
lora_alpha	16
lora_dropout	0.05
num_train_epochs	30

Table 2: Main parameters used in fine-tuning.

3.2 Experimental Results

This shared task includes a total of three test sets: TestA, TestB, and TestC. Among them, TestA and TestB are modern Chinese test sets, and TestC is an ancient Chinese test set. The AlignSmatch (Xiao et al., 2022) F-scores of all teams on these test sets are shown in Table 3. BLCU is the result of our system.

According to the results on the test set, we find that the performance of our system has a small gap with other systems on TestA and TestB, but a large gap on TestC. This indicates that the transferability of our system in zero-shot scenarios is still limited.

Team	TestA	TestB	TestC
GDUFE	0.8119	0.7527	0.7156
HITSZ	0.8096	0.7485	0.6692
BLCU	0.7891	0.7416	0.5772

Table 3: The results of each team on the three test sets. GDUFE is the result of Guangdong University of Finance and Economics, HITSZ is the result of Harbin Institute of Technology (Shenzhen), and BLCU is our result.

[0]https://huggingface.co/shibing624/text2vec-base-chinese
[1]https://huggingface.co/baichuan-inc/Baichuan2-13B-Base

3.3 The Impact of the Retrieval-Augmented Strategy

To verify the impact of the retrieval augmented strategy on system results, we design a set of ablation experiments for result analysis. The experimental results listed in Table 4. Through the experimental results, we find that the retrieval augmented strategy indeed improve the system's performance to some extent. It is worth noting that the retrieval augmented strategy significantly improve the system's performance on TestC. This indicates that the retrieval augmented strategy has advantages in zero-shot scenarios.

Method	TestA	TestB	TestC
w/ retrieval-augmented	**0.7891**	**0.7416**	**0.5772**
w/o retrieval-augmented	0.7725	0.7341	0.5366

Table 4: Comparison of results using the retrieval-augmented strategy.

3.4 Error Analysis

Based on the analysis of model prediction results, we summarize several problems currently existing in our system.

- **The generated format is illegal**. Since we use relatively complex data structures to represent the output of each task, there will be some predictions in the system's prediction results that are illegal in the data structure. This inspires us that we may need to design a more reasonable representation form for CAMR tasks.

- **The generated content is illegal**. Since our system uses an autoregressive generative approach, the system will generate some content that does not match the input information or is inconsistent with the CAMR Schema (for example, the generated relation is not a relation in the CAMR Schema). This inspires us to possibly need to design more complex constrained decoding strategies.

- **Lack of relevant knowledge**. Since ancient Chinese texts are usually historical documents, the knowledge contained therein deviates significantly from the knowledge commonly used in modern Chinese. For example, the system make an error in node prediction for the sentence "命子封二百乘以伐京。". It incorrectly predicted the node "北京" based on "京". This inspires us that we may need to pre-train LLM on ancient Chinese texts.

4 Conclusion

In this paper, we propose a two-stage generative CAMR parsing method based on LLMs. We decompose the CAMR task into two pipeline subtasks: alignment-aware node generation and relation-aware tuple generation to reduce the difficulty of understanding the CAMR task for LLMs. Furthermore, we introduce retrieval-augmented strategy during training and inference to enhance the transfer learning capabilities of our system. On three test sets of the CAMR 2024 evaluation task, our system achieve promising results. Through the analysis of experimental results, we find that our system still performs poorly on the ancient Chinese test set, which inspires us to further enhance our system's CAMR auto-parsing ability in low-resource scenarios through pre-training, data augmentation, and other techniques in future works.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. *Linguistic Annotation Workshop,Linguistic Annotation Workshop*, Aug.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2022. One spring to rule them both: Symmetric amr semantic parsing and generation without a complex pipeline. *Proceedings of the AAAI Conference on Artificial Intelligence*, page 1256412573, Sep.

Liang Chen, Bofei Gao, and Baobao Chang. 2022. A two-stage method for chinese amr parsing. Sep.

Wenyang Gao, Xuefeng Bai, and Yue Zhang. 2023. System report for ccl23-eval task 2: Westlakenlp, investigating generative large language models for chinese amr parsing. In *Proceedings of the 22nd Chinese National Conference on Computational Linguistics (Volume 3: Evaluations)*, pages 64–69.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Young-Suk Lee, RamonFernandez Astudillo, Radu Florian, Tahira Naseem, and Salim Roukos. 2023. Amr parsing with instruction fine-tuned pre-trained language models. Apr.

Bin Li, Yuan Wen, Li Song, Weiguang Qu, and Nianwen Xue. 2021. Building a chinese amr bank with concept and relation alignments. *Linguistic Issues in Language Technology*, Aug.

Kexin Liao, Logan Lebanoff, and Fei Liu. 2018. Abstract meaning representation for multi-document summarization. In Emily M. Bender, Leon Derczynski, and Pierre Isabelle, editors, *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1178–1190. Association for Computational Linguistics.

Linfeng Song, Daniel Gildea, Yue Zhang, Zhiguo Wang, and Jinsong Su. 2019. Semantic neural machine translation using amr. *Transactions of the Association for Computational Linguistics*, page 1931, Mar.

Chuan Wang, Bin Li, and Nianwen Xue. 2018. Transition-based chinese amr parsing. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, Jan.

Liming Xiao, Bin Li, Zhixing Xu, Kairui Huo, Minxuan Feng, Junsheng Zhou, and Weiguang Qu. 2022. Alignsmatch: A novel evaluation method for chinese abstract meaning representation parsing based on alignment of concept and relation. In *Proceedings of the Language Resources and Evaluation Conference*, pages 5938–5945, Marseille, France, June. European Language Resources Association.

Runxin Xu, Peiyi Wang, Tianyu Liu, Shuang Zeng, Baobao Chang, and Zhifang Sui. 2022. A two-stream amr-enhanced model for document-level event argument extraction. In Marine Carpuat, Marie-Catherine de Marneffe, and Iván Vladimir Meza Ruíz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 5025–5036. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.