LLMSR@XLLM25: A Language Model-Based Pipeline for Structured Reasoning Data Construction

Hongrui Xing[12†*], Xinzhang Liu[2*], Zhuo Jiang[2], Zhihao Yang[2], Yitong Yao[2], Zihan Wang[2], Wenmin Deng [2], Chao Wang[2], Shuangyong Song[2], Wang Yang[1‡], Zhongjiang He[2‡], Yongxiang Li[2]

[1]School of Cyber Science and Engineering, Southeast University
[2]Institute of Artificial Intelligence (TeleAI), China Telecom Corp Ltd.

Correspondence: {xinghr,wang.yang}@seu.edu.cn
{liuxz2,hezj}@chinatelecom.cn

Abstract

In this paper, we present a novel pipeline for the XLLM Shared Task-III: Large Language Model for Structural Reasoning (LLM-SR). Our pipeline addresses key challenges in automatic process-reward training data construction, such as high manual annotation costs, limited accuracy of large models in structured data processing, and dependency on auxiliary information for validation. To overcome these limitations, we first decompose the construction process into extraction and validation phases. Leveraging model-generated annotations, we produce pseudo-labeled data and iteratively refine model performance. Second, by analyzing structured data patterns, we encode structural constraints into a rule-based module and fine-tune the model with Gradient Reward Policy Optimization (GRPO), significantly improving structured data extraction success rates. Finally, we train the model to generate critical responses that assess evidence-conclusion relationships, thus enhancing validation reliability. Experimental results demonstrate that our pipeline outperforms models with an order of magnitude more parameters and achieves the first position on the task[1].

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities in mathematical and logical reasoning tasks, particularly when employing Chain-of-Thought (CoT) prompting to decompose problems into multi-step reasoning processes (Guo et al., 2025)(Yang et al., 2024)(Li et al., 2024a)(Wang et al., 2024b). However, even state-of-the-art models often produce unreliable intermediate reasoning steps, leading to cascading errors that compromise final outputs (Tyen et al., 2024). To mitigate this issue, existing research has introduced step-wise verification methods (Zeng et al., 2023). For instance, process reward models (PRM) can evaluate reasoning paths during training, identify erroneous steps, and offer precise corrective feedback (Li et al., 2023). Alternatively, step-wise analysis of CoT data from both correctness and redundancy perspectives can generate high-quality reasoning traces for training (Xia et al., 2025). These approaches not only enhance reasoning reliability but also improve overall data quality through generating constructive critiques of flawed reasoning steps, thereby providing valuable optimization signals for model refinement.

However, developing such step-wise verification models faces three fundamental challenges:

- The scarcity of high-quality step-level annotated reasoning datasets.
- The limited capability of current models in both processing structural inputs and constructing regular outputs.
- The accuracy in justifying the logical validity step by step.

Acquiring high-quality training data requires labor-intensive step-by-step annotation of reasoning processes with correctness feedback. For instance, the PRM800K dataset (Lightman et al., 2023) utilizes expert annotators to provide process supervision annotations. This heavy dependence on skilled annotators significantly hinders both the development and practical application of step-wise verification models.

Moreover, existing verification approaches typically employ simplistic segmentation method (e.g. explicit "Step:" markers, double line breaks, or periods) to parse reasoning steps (Zhang et al., 2024). Such rigid segmentation fails to capture the nuanced compositional structure inherent in natural

[1]Codes: https://github.com/pipiPdesu/CoTParser.
[†]Work done during the internship at TeleAI.
[*]These authors contributed equally to this work.
[‡] Corresponding authors.

Proceedings of the 1st Joint Workshop on Large Language Models and Structure Modeling (XLLM 2025), pages 342–350

CoT reasoning. This limitation fundamentally constrains the verification fidelity in real-world applications, where reasoning complexity often exceeds template-based patterns.

Additionally, verification models struggle with complex problems, as even large-scale models frequently fail to accurately determine step correctness, revealing critical limitations in current verification paradigms. To generate high quality dataset, Marh-Shepherd (Wang et al., 2024a) uses rejection sampling to generate multiple reasoning paths starting from certain steps, approximating step correctness based on the accuracy of the final answer. rStar-Math (Guan et al., 2025) constructs positive-negative sample pairs to train PPM, guiding the model towards correct solutions. Although these methods effectively create step-level supervision data and improve PRM capabilities, their data primarily consist of synthetic samples and require substantial computational resources and additional messages for construction. More importantly, they cannot accurately parse the naturally occurring CoT processes.

To tackle these challenges, XLLM Shared Task-III: LLM for Structural Reasoning requires participants to extract all conditions, statements, and their corresponding evidence from given problems and associated CoT processes, then determine whether the evidence sufficiently supports each extracted statement-evidence pair. This approach achieves fine-grained CoT analysis to enhance the generation of more coherent and accurate reasoning processes.

In this work, we propose a fine-grained analysis pipeline for CoT reasoning processes. The method decomposes the task into two components: extraction and verification. For the extraction task, following the construct of AIFlow (Shao and Li, 2025), we identify the problem conditions, statements, and supporting evidence from the question and CoT process. To address data scarcity, we first employ prompt engineering and preliminary fine-tuning to generate high-quality pseudo-labels. These extraction patterns are then formalized as prior knowledge into rule-base reward, and the model is further trained using GRPO (Shao et al., 2024) to streamline the extraction process and improve extraction accuracy. For the verification task, inspired by positive-incentive noise (Li, 2022), we reformulate it as generating concise yet effective critiques for statement-evidence pairs to determine whether the evidence supports the statement. Our method achieved first place in this competition. While maintaining low resource consumption, our model improves extraction capability by 20% compared to baselines. Our findings demonstrate the feasibility of using prior knowledge as rule reward to enhance model performance on specific NLU tasks like text extraction and recognition, as well as transforming verification tasks into critique generation tasks to improve verification capabilities of smaller models.

2 Related work

LLMs for Information Extraction. The emergence of large language models has introduced new solutions and research directions for information extraction. (Wu et al., 2024) proposed the Multi-stage Structured Entity Extraction method, which enhances effectiveness and efficiency by breaking down the task. PIVOINE (Lu et al., 2023) focuses on the issue of Open-world IE, improving the model's instruction-following ability by constructing the INSTRUCTOPENWIKI dataset, and demonstrates excellent generalization to unseen instructions. LLMs for Text2SQL (Li et al., 2024b)(Wu et al., 2025) also provide a new avenue for exploring their role in information extraction.

Step Verification for LLMs. To enable fine-grained analysis of CoT reasoning, existing studies have attempted to evaluate model performance through reasoning process inspection. RECEVAL (Prasad et al., 2023) proposes reference-free metrics based on entailment relations and pointwise variational information to assess step correctness and information gain. Parser-based method (Saparov and He, 2023) parses model-generated CoT into symbolic proofs for formal analysis. Other works employ LLMs themselves for correctness verification. Inspired by RFT (Xia et al., 2025), most approaches sample multiple reasoning paths to inversely estimate step validity. (Zhang et al., 2024) uses correct answers to guide models in critiquing their own incorrect responses, then filters high-quality critiques to train verification models. (Wan et al., 2024) determines answer correctness through multi-path consistency checks and identifies erroneous steps though multi-agent debate. (Xia et al., 2025) fine-tunes models to score reasoning steps from both validity and redundancy perspectives. (Tyen et al., 2024) observes LLMs' underperformance in error localization tasks and trains small classifiers to identity errors. (Zeng

et al., 2023) introduces meta-reasoning to evaluate error-correction capabilities through recursive reasoning analysis.

Process reward model (PRM) dataset. In reinforcement learning, PRMs provide step-level feedback to align LLM reasoning with human expectations. For generating process-wise supervision data, PRM800K (Lightman et al., 2023) relies on manual annotation. Math-Shepherd (Wang et al., 2024a) automates this via rejection sampling: generating multiple reasoning paths from intermediate steps and assuming step correctness if most paths yield correct answers. rStar-Math (Guan et al., 2025) constructs positive-negative sample pairs to train Process Preference Model.

3 Methodology

In this section, we first introduce the detailed task description (Section 3.1), followed by presenting the complete extraction pipeline architecture along with the specific extraction methods for each module and the approach for generating pseudo-labeled datasets (Section 3.2). Subsequently, we elaborated on the application of GRPO for the extraction task (Section 3.3) and the method of employing verification to validate the statement-evidence pairs (Section 3.4). Our final solution is an LLM-powered pipeline for structured reasoning data construction, as shown in Figure 1.

3.1 Details of Challenge

This task requires participants to generate "question parsing" and "cot parsing" based on the content of "question" and "cot" for each given message. Specifically, the task is divided into two parts: Question Parsing and CoT Parsing. For Question Parsing, all relevant conditions required to solve the problem must be extracted from the given question text. For CoT Parsing, all statement-evidence pairs need to be extracted, and it must be logically verified whether the statement can be inferred from the evidence. Participants are only allowed to use Llama-3-8B-Instruct (Grattafiori et al., 2024) as the backbone model.

This task has released only 24 annotated examples as the training set, with questions sourced from LogiQA (Liu et al., 2021) and CoT generated from Llama3-8b-instruct.[2] Additionally, there are 50 test cases for evaluation set A that only release the questions and 97 test examples for evaluation set B. Appendix A shows part of the annotated examples.

3.2 Extraction Pipeline

For this task, accurately extracting the required information from the given question and CoT process is of vital importance. The task baseline proposes a method based on in-context learning for extraction and verification. This method employs few-shot learning to extract key points from the input question and CoT, directly outputs all components required. However, due to limitations in model size and the availability of annotated data, we believe that this method is difficult to further optimize. Therefore, we consider decomposing the entire task into two independent parts: Question Parsing and CoT Parsing. For CoT Parsing, we further break it down into Statement Extraction, Evidence Extraction, and Statement-Evidence Pair Verification. Subsequent steps may rely on the results of previous steps, meaning that this pipeline needs to run in a serial manner. However, optimization of different parts can be carried out independently. To achieve a higher score, we need to minimize the extraction of incorrect information during the extraction process to ensure a perfect match between the extracted content and the ground truth.

Question parsing Inspired by the data extraction strategy in REDSTONE (Chang et al., 2024), we divide the extraction approach into two components: extract and filter. First, the model performs sentence segmentation on the entire question, treating enumerated conditions as separate sentences. Subsequently, we filter all sentence segments that contain useful information to solve the problem. Notably, some conditions may appear in the question's interrogative clause (e.g. "If G goes to the United States, which of the following must be true?" provides the condition "G goes to the United States"). For such cases, we further extract the embedded conditions while removing irrelevant lexical items.

Statement extraction Since we cannot directly divide the steps based on explicit markers, it becomes challenging to estimate the number of statements in the CoT process. To figure out this issue, we divide each natural reasoning step into three substeps:

- Summarization of given conditions.
- Derivation of new information from known conditions (corresponding to **evidence**).

[2]All data can be found in https://huggingface.co/datasets/shuyi-zsy/LLMSR/tree/main/llmsr.

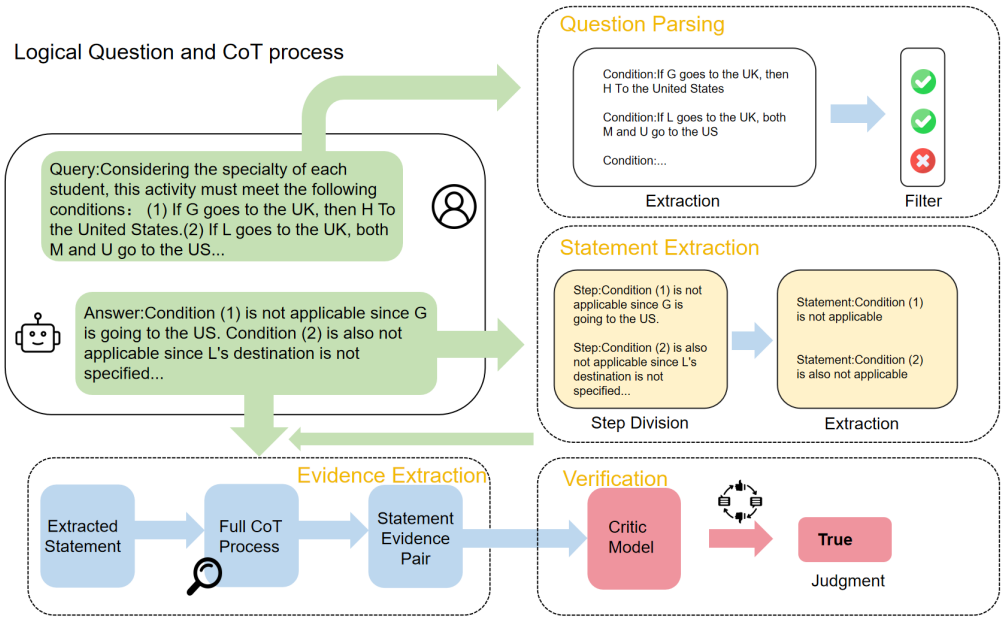

Figure 1: Architecture of CoT parsing pipeline. The pipeline consists of four key modules. The Question Parsing Module receives the question, extract all sentences and filter out valid conditions. The Statement Parsing Module takes the CoT process, divides it into multiply steps, then extract valid statements from each step. The Evidence Extraction Module process all statements and the CoT process to identify corresponding evidence. The verification Module takes all statement-evidence pairs, justify their validity.

- Generation of new conditions or conclusions (corresponding to **statements**).

Following this principle, we first instruct the model to segment the entire CoT process into these refined steps. This ensures that each reasoning step typically contains at most 0-1 statements, allowing the extraction model to focus on small contextual segments and significantly reducing extraction complexity. Additionally, this decomposition of natural steps enables us to expand the original 24 data samples 4-5 times, thereby facilitating the construction of high-quality fine-tuning data. After fine-tuning, the model's extraction capability is further improved.

Evidence extraction Initially we considered extracting evidence from the pre-segmented steps when identifying statements. However, we observed that certain pieces of evidence often span multiple steps. For instance, a concluding statement such as "From above, we can conclude" requires all previously obtained valid statements as supporting evidence. Moreover, extracting evidence directly from individual steps may introduce error propagation caused by incorrect step segmentation. Therefore, for each extracted statement, we need to search for its corresponding evidence throughout the entire CoT process. Since statements typically originate from the original text and their supporting evidence usually appears near clear discourse markers (e.g., connectives or adverbs), this provides sufficient context for the model to accurately locate the relevant evidence.

To further enhance the performance of each module, we consider utilizing the aforementioned pipeline to generate more pseudo-labeled data. First, we sample questions from LogiQA. Subsequently, to ensure that the synthesized CoT processes maintain distributional consistency with the task's given data, we sample one question-answer pair each from the training set and evaluation set A, serves as one-shot provided to Llama3-8B-Instruct as reference for generating responses.

3.3 Utilize GRPO for Extraction

Due to the insufficient extraction accuracy of this pipeline, the pseudo-labeled dataset generated by this method can hardly provide substantial improvements to the model. We discovered that instead of using human annotation experience as prompts for model learning or allowing the model to memorize patterns through more data, we can incorporate these as rules to provide rewards in GRPO. For statement extraction, we summarized the following potential guidelines from the training dataset:

- The statement must originate from original text
- The statement must end with a period
- The statement length must be no fewer than 4 words and no more than 50 words
- The statement must not contain connectives such as "since" or "there is"
- The statement must not duplicate conditions extracted from the question
- The statements should appear in sequence
- Because there must be evidence in between, no two statements should be consecutive in the original CoT

These guidelines can serve both as rules for manual annotation and as directions for model exploration during reinforcement learning. When the model's response violates these rules, it receives a negative reward, and only when it perfectly matches the correct answer does it receive a positive reward. This approach encourages the model to learn the human method of data extraction.

After fine-tuning with GRPO, the model can directly extract all statements from the CoT process, maintaining accuracy while reducing intermediate computational overhead. This method demonstrates the potential of incorporating prior rules into GRPO's rule-based rewards to enhance LLM performance on traditional NLP tasks.

3.4 Verification

The objective of this part is to determine whether each extracted statement can be inferred from its corresponding evidence. For this problem, we make the following assumptions:

1. The model needs to rely on all known conditions of the question to determine whether the statement holds. When judging whether the evidence supports the statement, the model should first determine whether the statement is valid in the context of the question before assessing whether the evidence sufficiently supports the statement.

2. All statement-evidence pairs are independent. When judging whether a statement holds, only its corresponding evidence is needed, not other evidence or statements from the context. The evidence should consist of all the sentences that can prove the statement. If a statement requires additional evidence beyond its corresponding evidence, it indicates that the evidence is not sufficient to fully support the statement.

3. Judging statement-evidence pairs using the model should not be a simple binary classification task but should fully leverage the model's reasoning process. However, due to the limitations of the PRM function, the reasoning process should not be overly lengthy.

Based on these assumptions and inspired by the approach in CFT (Wang et al., 2025) of criticizing noise, we believe that the output of the Verification model should be a critique with justification of the statement-evidence pair. The critique part should directly point out the reasons why the evidence support or does not support the statement and provide the final justification based on these reasons. We used DeepSeek v3-0324 (Liu et al., 2024) to generate a critique dataset from the extracted dataset and fine-tuned the discriminative model accordingly. The success rate after fine-tuning remained similar to that of DeepSeek v3, indicating that training the model to criticize noise to judge the correctness of reasoning steps is effective, and the model can acquire this ability with limited data.

4 Experiment

4.1 Setup

Pipeline Overview Our final pipeline operates as follows: For each question, the model first extracts all potential conditions followed by a filtering module. For statement-evidence pairs, the model directly extracts all statements from the CoT process. After removing duplicates with the extracted conditions, it searches for corresponding evidence in the CoT for each statement. Finally, a verification model justifies each statement-evidence pair.

Parameter Settings We trained four Llama3-8B-Instruct models for this pipeline: condition extraction, statement extraction, evidence extraction and verification. All models were full parameter fine-tuned for 3 epochs at a learning rate of $1e{-}5$ using pseudo-labeled data generated by above pipeline. The verification model training data outputs were produced by DeepSeek V3-0324. During inference,

Method/Team	Question(%)	Statement(%)	Evidence(%)	Reasoning(%)
Baselines				
ICL(Llama3-8b-Instruct)	73.01	42.40	18.10	10.32
ICL(Qwen2-7b-Instruct)	69.98	42.1	15.09	8.51
ICL(Telechat2)	72.18	46.39	16.82	7.71
ICL(DeepSeek-R1)	**81.87**	44.84	12.42	10.79
dcchen(2nd)	78.53	54.31	**23.57**	15.71
blazerblade(3rd)	76.7	40.44	11.32	6.20
TeleAI(Ours)	81.2	**55.07**	22.44	**17.09**

Table 1: Comparison of top3 teams with our submission, along with baseline method of different models.

we used rejection sampling to obtain $N = 31$ samples from the verification model to determine the final results.

Evaluation Metrics We assess extracted conditions, statements, and evidence using both semantic and lexical similarity against ground truth. Semantic similarity is computed using nli-deberta-v3-base (He et al., 2021)(Liu et al., 2023), while lexical similarity uses METEOR scores. The matching score is the geometric mean of these two measures. Thresholds are set at 0.95 for question parsing and 0.9 for CoT parsing - only scores exceeding these thresholds are considered matches. For evidence evaluation, we only consider evidence paired with matched statements. A statement-evidence pair is verified as correct only when both components match. The final evaluation metric is the macro F1 score across all four components.

Baseline We adopt the provided in-context learning method as our baseline framework. For consistency with the task requirements, we evaluated four baseline models: Llama3-8B-Instruct (Grattafiori et al., 2024), Qwen2-7B-Instruct (Yang et al., 2024), Telechat2 (He et al., 2024) and DeepSeek-R1 (Guo et al., 2025). All models were tested under identical experimental conditions to ensure fair comparison.

4.2 Main Result

Table 1 shows the comparison of our solution with the top three other teams and the baseline. Our solution achieved the highest scores in Statement and Reasoning parts, maintaining the best overall task performance. Since we failed to extend the reinforcement learning method to evidence extraction, the corresponding score was slightly lower than the highest score. However, our method still achieved a high Reasoning score while maintaining a small number of extracted statement-evidence pairs, which proves that our verification model is more powerful than what the score reflects.

4.3 Ablation

We conducted ablation studies to verify the effectiveness of each newly added module in the pipeline. Since some modules are only effective for certain subtasks among the four subtasks, we only list the evaluation of the parts affected by adding a particular module. The experimental results are shown in Table 2.

Compared to directly using the model for content extraction, employing an optimized pipeline for step-by-step extraction and filtering significantly enhances the success rate of question and statement extraction. After post-training with GRPO, the success rate of statement extraction is notably improved. Benefiting from an increased base for extracting statements, the evidence score also increases. We trained the model to use critique for verification, leading to a substantial improvement in reasoning accuracy. Our ablation study demonstrates the feasibility of LLM with GRPO to perform traditional NLU tasks, and for the model's verification process, learning to criticize statement-evidence pairs is easier to enhance verification accuracy than directly justify their validity.

5 Conclusion

In this article, we propose an effective method for the XLLM Shared Task-III in LLM for Structural Reasoning. We present a novel pipeline for fine-grained analysis of CoT processes that achieves extraction and verification performance comparable to state-of-the-art models while maintaining low resource requirements. Our work demonstrates GRPO's potential for enhancing LLM performance on traditional NLU tasks and validates the feasibility of using critique to develop model verification

Method	Question(%)	Statement(%)	Evidence(%)	Reasoning(%)
Directly Extraction and Verification	61.19	37.09	15.02	8.11
+ Step-wise Extraction Pipeline	81.20	46.81	16.74	5.45
+ Tuned with GRPO		55.07	22.44	4.68
+ Critique Verification				17.09

Table 2: Ablation results on Test set A.

capabilities. The proposed framework opens new possibilities for structured reasoning analysis in resource-constrained scenarios while maintaining competitive accuracy, with future work planned to explore applications to broader reasoning tasks and further optimization of the verification component.

Limitations

Our approach still has some limitations. First, the models are trained exclusively on pseudo-labeled data, whose inherent accuracy constraints impose an upper bound on the extraction and verification performance of the entire pipeline. Second, our experiments are conducted solely on the LogiQA dataset with CoT processes generated by Llama3-8B-Instruct, without validation on other types of chain-of-thought datasets or different LLM-generated reasoning paths. These limitations suggest directions for future improvements in data quality and generalization testing.

References

Yaoyao Chang, Lei Cui, Li Dong, Shaohan Huang, Yangyu Huang, Yupan Huang, Scarlett Li, Tengchao Lv, Shuming Ma, Qinzheng Sun, Wenhui Wang, Furu Wei, Ying Xin, Mao Yang, Qiufeng Yin, and Xingxing Zhang. 2024. Redstone: Curating general, code, math, and qa data for large language models. *Preprint*, arXiv:2412.03398.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. 2025. rstar-math: Small llms can master math reasoning with self-evolved deep thinking. *arXiv preprint arXiv:2501.04519*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *Preprint*, arXiv:2111.09543.

Zhongjiang He, Zihan Wang, Xinzhang Liu, Shixuan Liu, Yitong Yao, Yuyao Huang, Xuelong Li, Yongxiang Li, Zhonghao Che, Zhaoxi Zhang, and 1 others. 2024. Telechat technical report. *arXiv preprint arXiv:2401.03804*.

Xiang Li, Yiqun Yao, Xin Jiang, Xuezhi Fang, Chao Wang, Xinzhang Liu, Zihan Wang, Yu Zhao, Xin Wang, Yuyao Huang, and 1 others. 2024a. Tele-flm technical report. *arXiv preprint arXiv:2404.16645*.

Xuelong Li. 2022. Positive-incentive noise. *IEEE Transactions on Neural Networks and Learning Systems*.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023. Making language models better reasoners with step-aware verifier. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5315–5333, Toronto, Canada. Association for Computational Linguistics.

Zhongqiu Li, Zhenhe Wu, Mengxiang Li, Zhongjiang He, Ruiyu Fang, Jie Zhang, Yu Zhao, Yongxiang Li, Zhoujun Li, and Shuangyong Song. 2024b. Scalable database-driven kgs can help text-to-sql.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2021. Logiqa: a challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3622–3628.

Shixuan Liu, Chen Peng, Chao Wang, Xiangyan Chen, and Shuangyong Song. 2023. icsberts: Optimizing pre-trained language models in intelligent customer service. *Procedia Computer Science*, 222:127–136.

Keming Lu, Xiaoman Pan, Kaiqiang Song, Hongming Zhang, Dong Yu, and Jianshu Chen. 2023. Pivoine: Instruction tuning for open-world information extraction. *arXiv preprint arXiv:2305.14898*.

Archiki Prasad, Swarnadeep Saha, Xiang Zhou, and Mohit Bansal. 2023. ReCEval: Evaluating reasoning chains via correctness and informativeness. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10066–10086, Singapore. Association for Computational Linguistics.

Abulhair Saparov and He He. 2023. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *11th International Conference on Learning Representations*.

Jiawei Shao and Xuelong Li. 2025. Ai flow at the network edge. *IEEE Network*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Gladys Tyen, Hassan Mansoor, Victor Carbune, Peter Chen, and Tony Mak. 2024. LLMs cannot find reasoning errors, but can correct them given the error location. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13894–13908, Bangkok, Thailand. Association for Computational Linguistics.

Guangya Wan, Yuqi Wu, Jie Chen, and Sheng Li. 2024. Cot rerailer: Enhancing the reliability of large language models in complex reasoning tasks through error detection and correction. *arXiv preprint arXiv:2408.13940*.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024a. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, Bangkok, Thailand. Association for Computational Linguistics.

Yubo Wang, Xiang Yue, and Wenhu Chen. 2025. Critique fine-tuning: Learning to critique is more effective than learning to imitate. *Preprint*, arXiv:2501.17703.

Zihan Wang, Yitong Yao, Li Mengxiang, Zhongjiang He, Chao Wang, Shuangyong Song, and 1 others. 2024b. Telechat: An open-source billingual large language model. In *Proceedings of the 10th SIGHAN Workshop on Chinese Language Processing (SIGHAN-10)*, pages 10–20.

Haolun Wu, Ye Yuan, Liana Mikaelyan, Alexander Meulemans, Xue Liu, James Hensman, and Bhaskar Mitra. 2024. Learning to extract structured entities using language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6817–6834, Miami, Florida, USA. Association for Computational Linguistics.

Zhenhe Wu, Zhongqiu Li, Mengxiang Li, Jie Zhang, Zhongjiang He, Jian Yang, Yu Zhao, Ruiyu Fang, Yongxiang Li, Zhoujun Li, and Shuangyong Song. 2025. MR-SQL: Multi-Level Retrieval Enhances Inference for LLM in Text-to-SQL. In *Proceedings of the 2025 International Conference on Database Systems for Advanced Applications*. Accepted.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. 2025. Evaluating mathematical reasoning beyond accuracy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 27723–27730.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Zhongshen Zeng, Pengguang Chen, Shu Liu, Haiyun Jiang, and Jiaya Jia. 2023. Mr-gsm8k: A meta-reasoning benchmark for large language model evaluation. *arXiv preprint arXiv:2312.17080*.

Yunxiang Zhang, Muhammad Khalifa, Lajanugen Logeswaran, Jaekyeom Kim, Moontae Lee, Honglak Lee, and Lu Wang. 2024. Small language models need strong verifiers to self-correct reasoning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15637–15653, Bangkok, Thailand. Association for Computational Linguistics.

A CoT Parsing Example


```
{
        "question": "There are 7
    outstanding students G, H, L, M, U, W
    and Z in a school.During the summer
    vacation, the school will send them to
     the United Kingdom and the United
    States for inspection.The school has
    only 7 students participating in this
    activity, and each person happens to
    go to one of these two countries.
    Considering the specialty of each
    student, this activity must meet the
    following conditions? (1) If G goes to
     the UK, then H To the United States
    .(2) If L goes to the UK, both M and U
     go to the US......",
        "question_parsing": [
            "The school has only 7
    students participating in this
    activity, and each person happens to
    go to one of these two countries",
            "If G goes to the UK, then H
    To the United States",
            "If L goes to the UK, both M
    and U go to the US",
            ......
        ],
        "answer": "b",
        "cot": "Since G goes to the United
     States, we need to analyze the
    conditions that follow. Condition (1)
    is not applicable since G is going to
    the US. Condition (2) is also not
    applicable since L's destination is
    not specified......"
        "cot_parsing": [
            {
                "statement": "Condition
    (1) is not applicable",
                "evidence": "G is going to
     the US",
                "Verification": "true"
            },
            {
                "statement": "Condition
    (2) is also not applicable",
                "evidence": "L's
    destination is not specified",
                "Verification": "true"
            },
            .......
        ],
    },
```