Problem-Solving Logic Guided Curriculum In-Context Learning for LLMs Complex Reasoning

Anonymous ACL submission

Abstract

In-context learning (ICL) can significantly enhance the complex reasoning capabilities of large language models (LLMs), with the key lying in the selection and ordering of demonstration examples. Previous methods typically relied on simple features to measure the relevance between examples. We argue that these features are not sufficient to reflect the intrinsic connections between examples. In this study, we propose a curriculum ICL strategy guided by problem-solving logic. We select demonstration examples by analyzing the problem-solving logic and order them based on curriculum learning. Specifically, we constructed a problem-solving logic instruction set based on the BREAK dataset and fine-tuned a language model to analyze the problem-solving logic of examples. Subsequently, we selected appropriate demonstration examples based on problem-solving logic and assessed their difficulty according to the number of problem-solving steps. In accordance with the principles of curriculum learning, we ordered the examples from easy to hard to serve as contextual prompts. Experimental results on multiple benchmarks indicate that our method outperforms previous ICL approaches in terms of performance and efficiency, effectively enhancing the complex reasoning capabilities of LLMs. Our project will be publicly available subsequently.

1 Introduction

Large language models (LLMs) (Ouyang et al., 2022; Ye et al., 2023; Bahrini et al., 2023) can rapidly acquire new capabilities through in-context learning (ICL) to solve many new tasks (Wies et al., 2024; Xu et al., 2024), and can be extended through chain of thought (CoT) (Wei et al., 2022) to solve many tasks that require complex reasoning (Hao et al., 2023; Zhang et al., 2023). Researchers believe that through ICL, LLMs can implicitly learn the problem-solving patterns demonstrated in contextual examples and apply them to new tasks (Bhattamishra et al., 2023; Dai et al., 2023). This means that LLMs have the ability to learn and apply problem-solving patterns on the spot from given examples.

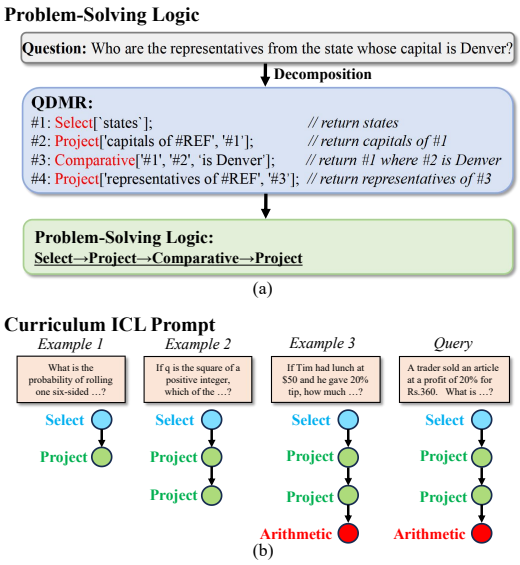

Figure 1: (a) The transformation from QDMR to problem-solving logic. (b) An example of curriculum ICL. Example selection depends on the similar problem-solving logic, and example ordering depends on the number of operations contained in the logic.

In recent years, supervised fine-tuning (SFT) methods (Dong et al., 2023) and reinforcement learning optimization reasoning methods (Du et al., 2023; Guo et al., 2025) have been able to significantly enhance the reasoning abilities of LLMs through training. Despite this, due to the unique characteristic of ICL that it can enhance problem-solving capabilities without training, it still holds value as significant as the methods mentioned above, especially when facing the need to reduce costs or quickly apply to new tasks. Relevant work (Hsieh et al., 2023) has already shown that LLMs possess a wealth of basic knowledge and fundamental capabilities that can be effectively ac-

tivated through a small number of examples. Particularly, LIMO (Ye et al., 2025) fine-tuned a large language model with only a few hundred examples and achieved results that are close to or even on par with the current state-of-the-art reinforcement learning optimization inference. Therefore, we believe that the ICL capabilities of current LLMs are still far from being fully realized. There is a need to design better prompts to effectively enhance the effectiveness of ICL.

ICL learns demonstration examples in sequence and then solves problems, which closely resembles the process of humans learning knowledge step by step. We believe that organizing demonstration examples in a way similar to human educational curriculum construction is crucial. It helps LLMs learn the knowledge and patterns shown in the examples and solve given problems effectively. Therefore, strategies for curriculum learning (Bengio et al., 2009) can be adopted for the organization of demonstration examples. The key to ICL lies in example selection and ordering, which requires measuring the relevance between examples. Traditional simple statistical information, such as similarity (Robertson et al., 2009; Wu et al., 2023a; An et al., 2023) and perplexity (Gonen et al., 2023; Margatina et al., 2023a), is not sufficient to reflect the intrinsic connections between examples, especially from the perspective of problem-solving.

In this work, we innovatively propose an problem-solving logic guided curriculum ICL method, which constructs the optimal ICL prompt for the query based on problem-solving logic. The Question Decomposition Meaning Representation (QDMR) (Wolfson et al., 2020) decomposes complex problems into several sub-questions for solving and formalizes these sub-questions with 13 custom "operations", which we refer to as *problem-solving logic*. Figure 1-(a) shows an example of problem decomposition and transformation into problem-solving logic. Although it cannot directly solve the problem, the problem-solving logic describes the steps required for solving and the order of these steps in formal language. Therefore, it can accurately measure the intrinsic connections between examples and construct a sequence of demonstration examples that are conducive to problem-solving. Figure 1-(b) shows an example of curriculum ICL. We select examples with similar problem-solving logic, which can help LLMs learn how to solve similar problems. Subsequently, we measure the difficulty of these examples by the number of problem-solving steps. The greater the number of steps, the more reasoning steps are involved, meaning the problem is more difficult to solve. Relying on the principles of curriculum learning, we order these examples from easy to hard to serve as the final in-context prompt.

Our main contributions are as follows:

(1) This paper proposes a problem-solving logic guided curriculum ICL strategy to enhance the reasoning performance of LLMs. We innovatively present problem-solving logic as the criterion for selection and ordering demonstration examples, which is expected to offer a novel perspective for future work.

(2) We constructed a problem-solving logic instruction set based on the BREAK dataset. Based on this, we fine-tuned a language model to automatically analyze the problem-solving logic of input questions.

(3) Extensive experiments are conducted on five datasets, and results show that our method achieves significant improvements in average performance and efficiency across all datasets, surpassing previous ICL methods and effectively enhancing the ability of LLMs in reasoning tasks.

2 Background

2.1 In-Context Learning

ICL is a capability that emerges as the training data and scale of LLMs increase (Dong et al., 2022). This allows LLMs to learn new tasks with only a few examples. Examples generally contain questions and answers. The query needs to maintain consistent formatting with the examples so that LLMs can provide accurate responses. This process is called few-shot.

Existing research shows that the key to enhancing ICL performance lies in the organization of demonstration examples, that is, the selection and ordering of examples. Taking text similarity as an example, the general process is to encode the candidate examples and the query into vector forms, and then select the examples most similar to the query by calculating the similarity between vectors. Subsequently, these examples are sorted according to text similarity. Finally, the sorted examples are then input into the LLMs together with the query for solving.

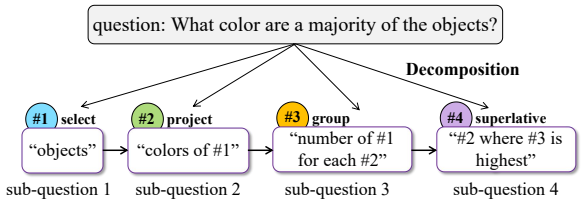

Figure 2: A QDMR example. The original question is decomposed into four sub-questions, each represented by an operation.

2.2 Problem-Solving Logic

QDMR is a general method for decomposing complex questions into several sub-questions for solving. They manually designed 13 operations, with each sub-question represented by an operation. The researchers proposed the BREAK dataset through manual annotation, which contains 60K question-answer pairs. Specific examples of each operation, as well as detailed information about the dataset, can be found in the Appendix A.

This work is inspired by QDMR and refers to the sequence of operators representing sub-questions as the *problem-solving logic*. The set of sub-questions decomposed by QDMR includes the required steps and the order between steps. Figure 2 shows a specific QDMR example. The original question is split into four sub-questions, each of which is described in a formal language with an operation, resulting in the corresponding problem-solving logic as follows:

$$\text{select} \rightarrow \text{project} \rightarrow \text{group} \rightarrow \text{superlative}$$

2.3 Curriculum Learning

Curriculum learning is a machine learning strategy (Bengio et al., 2009). It suggests that the training process should mimic human cognitive learning by starting with simple examples and gradually increasing in difficulty. The core of this method lies in how to measure the difficulty of examples, which often depends on the characteristics of the specific task. For example, in the field of computer vision, the number of objects in an image (Wei et al., 2016) or noise (Chen and Gupta, 2015) contained can be used to measure difficulty. In the field of natural language processing, sentence length (Platanios et al., 2019) can be used as a measure of difficulty. In addition to these, the difficulty can also be measured by human educational level (Lee et al., 2023) or evaluation models (Soviany et al., 2020).

3 Problem-Solving Logic Guided Curriculum ICL

This paper introduces a problem-solving logic guided curriculum ICL strategy. The overall methodology is illustrated in Figure 3. Specifically, we first constructed an instruction set based on the BREAK dataset and fine-tuned a language model to automatically analyze problem-solving logic. Then, we analyzed the problem-solving logic for all data in the benchmark training set to construct a dataset of candidate examples. When an actual query is input, its problem-solving logic is first analyzed and then compared with the candidate examples, selecting those with similar problem-solving steps as demonstration examples. Furthermore, the number of problem-solving steps serves as an appropriate metric for assessing the difficulty of each example. A greater number of steps means the problem is more difficult to solve. This inspired us to apply the principles of curriculum learning to order the demonstration examples from easy to hard. Finally, the ordered demonstration examples and the query are combined to form the final prompt, which is then input into the LLMs. The following sections will offer a detailed explanation of how problem-solving logic is analyzed, along with the process of selecting and ordering demonstration examples.

3.1 Problem-Solving Logic Analysis

We first need to train a language model to analyze the problem-solving logic, which is represented as an ordered set of several problem-solving steps.

Our approach constructs an instruction set based on the BREAK dataset. Specifically, the input to the instruction set is a problem, and the output is problem-solving logic and its formal language. The formal language ensures that the model correctly understands the problem-solving process. We then fine-tune a Llama3-8B model (Touvron et al., 2023; Dubey et al., 2024) with LoRA (Hu et al., 2022) on this instruction set. Once the model is trained, it can analyze problems from any dataset and extract their problem-solving logic. Examples of the instruction set can be found in the Appendix A. Details of fine-tuning and hyperparameters can be found in the Appendix B.

Analyzing the problem-solving logic is a crucial step in our work, providing the foundation for the subsequent curriculum ICL.

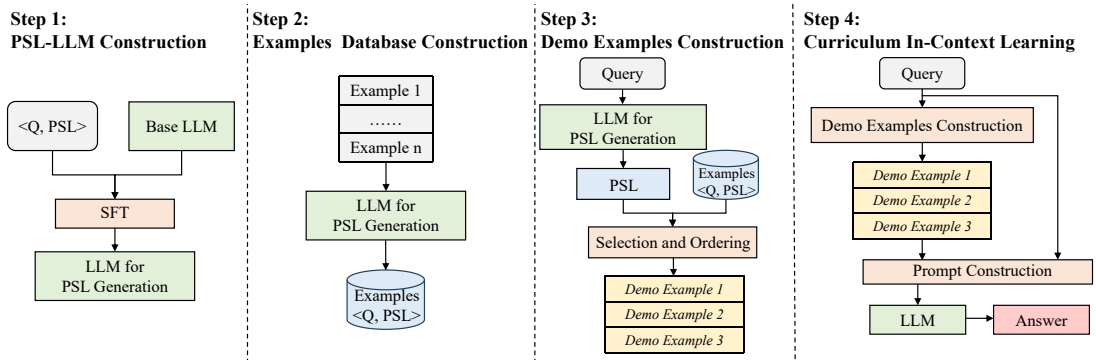

Figure 3: The overall flowchart of our method. First, a base LLM is fine-tuned using an instruction set for problem-solving logic (PSL) constructed from the BREAK dataset. Then, suitable demonstration examples are selected and ordered by analyzing the PSL of the candidate examples and the query. Finally, the selected demonstration examples and the query form the full prompt, which is fed into the LLM to obtain the results.

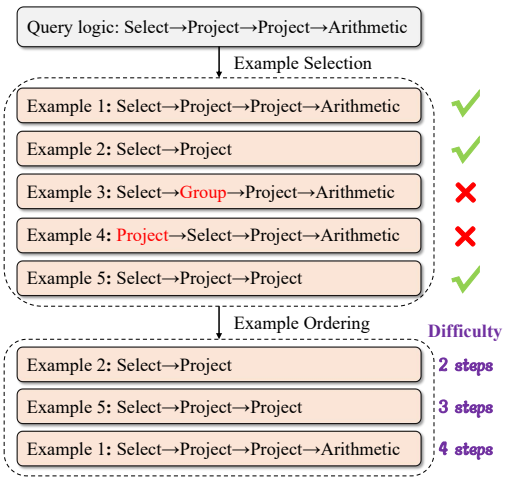

Figure 4: The process of example selection and ordering. (✓) denotes similar problem-solving logic, (×) indicates a matching failure, and **red** font indicates the reason for the matching failure. **Difficulty** is measured by the number of steps.

Algorithm 1 Demonstration Example Selection

Require: query T, LLM function $F(\cdot)$, set of candidate examples $\{E_1, E_2, \ldots, E_n\}$, each example E_i has its own solution logic $L_i = \{O_{i1}, O_{i2}, \ldots, O_{im_i}\}$.

Ensure: Mark matching demonstration examples.

$L_T \leftarrow F(T)$ {Obtain the solution logic for the query from LLM}
for each example E_i in $\{E_1, E_2, \ldots, E_n\}$ **do**
  $L_i \leftarrow \{O_{i1}, O_{i2}, \ldots, O_{im_i}\}$ {Retrieve solution logic of E_i}
  if L_i is a subsequence of L_T starting from the first operator **then**
    Mark E_i as a demonstration example
  end if
end for

3.2 Curriculum ICL

Based on the above problem analysis process, we can focus on problem-solving logic to guide the selection and ordering of demonstration examples. Figure 4 illustrates the process of example selection and ordering.

3.2.1 Demonstration Example Selection

First, we need to select appropriate demonstration examples. Compared to semantic information, we believe that selecting examples with similar problem-solving logic is more important. On one hand, similar problem-solving logic can guide LLMs in reasoning, and on the other hand, examples with similar logic but different semantics can enhance the model's generalization ability.

After analyzing the query and all candidate examples, our method selects demonstration examples based on the problem-solving logic. The selection criterion requires that the problem-solving operations set in each candidate example must be a subsequence of the query, meaning both the types of operations and their order must match exactly. Suppose the query has a problem-solving logic containing m operations, and the selected demonstration example has n operations ($m \geq n$); the n operations of the demonstration example must match the first n operations of the query. This method ensures that the demonstration example's problem-solving steps align with the first n steps of the query, avoiding any mismatch or additional problem-solving steps. The complete process is detailed in Algorithm 1.

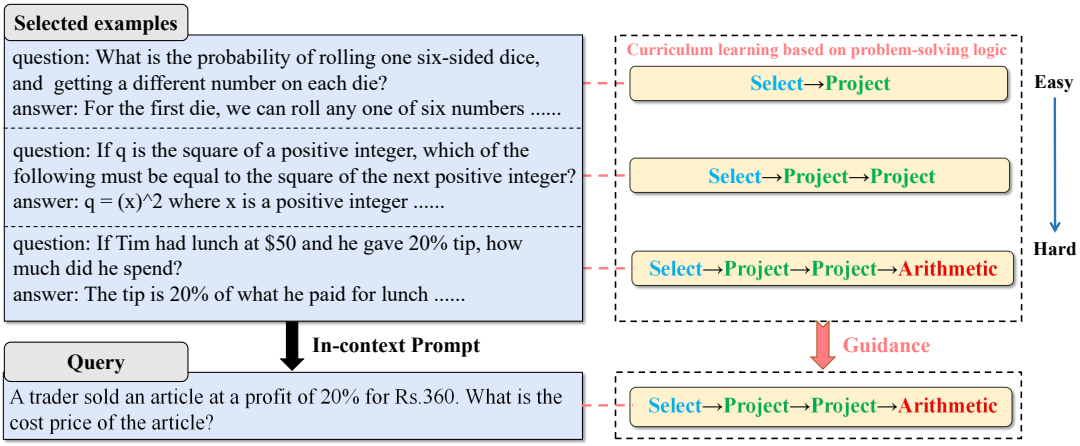

Figure 5: A complete example of curriculum ICL. The selected examples form the context information. The right half of the figure shows the problem-solving logic, which is the basis for example selection and ordering.

3.2.2 Demonstration Examples Ordering

The key to curriculum learning lies in how to measure the difficulty of examples. By introducing problem-solving logic, we can easily assess the difficulty of each example. The problem-solving logic consists of several operations, where a higher number of operations indicates more reasoning steps, thereby increasing the problem's difficulty.

Inspired by this, we applied curriculum learning principles, ordering examples from easy to hard. Specifically, we sorted the examples in increasing order based on the number of problem-solving steps, and used them along with the query to construct the final in-context prompt. Figure 5 shows a complete curriculum ICL example, including demonstration examples and the query.

4 Experiments and Analysis

4.1 Experimental Setup

Benchmarks. Our experiment includes two types of datasets, Arithmetic Reasoning and Commonsense Reasoning, and validation is conducted on five different datasets. Arithmetic Reasoning: (1) the AQuA (Ling et al., 2017) includes 254 test examples, (2) the SVAMP (Patel et al., 2021) includes 1000 test examples, (3) the Gsm8k includes 1319 test examples. Commonsense Reasoning: (1) the CommonsenseQA (Talmor et al., 2019) includes 1211 test examples, (2) the StrategyQA (Geva et al., 2021) includes 229 test examples.

Baselines. We compare our approach against seven methods that use ICL. Random selects demonstration examples and their order randomly. VoteK (Hongjin et al., 2022) selects the most similar k examples using k-nearest neighbors (KNN) and sorts them according to similarity scores. PromptSO (Shi et al., 2024) uses principal component analysis (Abdi and Williams, 2010) to select the most relevant basis questions and sorts them based on eigenvalue. AutoCoT (Zhang et al., 2022) uses k-means to automatically select the most representative examples that are closest to the cluster center. CoT+few-shot (Wei et al., 2022) manually designed fixed demonstration examples with reasoning processes. Self-Adaption ICL (SA-ICL) (Wu et al., 2023b) selects similar examples based on KNN and then chooses an appropriate order based on information compression. Active Learning ICL (AL-ICL) (Margatina et al., 2023b) selects most similar examples based on the principles of active learning and sorts them according to similarity.

Implement Details. We evaluate the effectiveness of our method on the Llama3-8B model. For each benchmark, we select demonstration examples from its training set to form prompt information to evaluate each test set data. For the SVAMP dataset, we adopted the same evaluation strategy as in previous work (Patel et al., 2021), using ASDiv-a (Miao et al., 2020) and MAWPS (Koncel-Kedziorski et al., 2016) together as the training set. To ensure a fair comparison, the number of selected examples is based on the settings in CoT (Wei et al., 2022) for different benchmarks, and our experiments do not exceed that limit.

4.2 Main Results and Analysis

We compare the performance of our approach with other ICL methods. All the comparison rusults are

Method	Selection Stategy	Ordering Stategy	Dataset					Avg.
			SVAMP	AQuA	Gsm8k	ComSenQA	StrategyQA	
Random	Random	Random	76.5%	46.5%	73.8%	75.8%	65.1%	67.53%
VoteK	KNN	Similarity	74.9%	44.9%	76.7%	75.4%	69.0%	68.19%
PromptSO	PCA	Eigenvalue	77.3%	43.7%	77.7%	75.6%	67.7%	68.40%
AutoCoT	K-means	Similarity	77.5%	47.2%	75.3%	76.0%	71.2%	69.44%
CoT + Fewshot	Fixed	Fixed	80.5%	44.5%	79.4%	75.1%	69.4%	69.79%
SA-ICL	KNN	Entropy	78.8%	47.6%	77.9%	**78.5%**	66.8%	69.95%
AL-ICL	KNN	Similarity	80.8%	45.7%	78.2%	77.9%	68.1%	70.13%
Ours	PSL	Curriculum	**83.4%**	**50.8%**	**81.1%**	75.0	**71.6%**	**72.37%**

Table 1: The table presents a comparison of experimental results across different benchmarks using Llama3-8B, demonstrating the accuracy contrast between various ICL methods. **Avg** represents the average accuracy across the different benchmarks. The best and second-best performances are highlighted in **bold** and underlined, respectively.

Difficulty Strategy	Ordering	Dataset					Avg.
		SVAMP	AQuA	Gsm8k	ComSenQA	StrategyQA	
Original Llama							
AL-ICL		80.8%	45.7%	78.2%	**77.9%**	68.1%	70.13%
Our Strategy							
Prioritize simplicity	w/ order	82.3%	47.6%	79.5%	75.5%	69.0%	70.79%
	w/o order	82.5%	47.2%	78.8%	76.1%	68.1%	70.55%
Prioritize difficulty	w/ order	81.8%	44.9%	77.9%	76.6%	67.7%	69.77%
	w/o order	81.6%	46.1%	79.6%	77.0%	67.2%	70.29%
Select Randomly	w/ order	81.3%	50.6%	80.2%	76.1%	70.3%	71.70%
	w/o order	80.9%	48.6%	79.2%	76.0%	71.2%	71.17%
Prioritize diversity	w/ order	**83.4%**	**50.8%**	**81.1%**	75.0%	**71.6%**	**72.37%**
	w/o order	80.5%	46.1%	80.1%	76.0%	65.9%	70.11%

Table 2: The table presents the accuracy of benchmarks under different difficulty selection strategies. "w/ order" indicates that the examples are ordered based on curriculum learning, while "w/o order" means the examples are randomly ordered. The best and second-best performances are highlighted in **bold** and underlined, respectively.

tabulated in Table 1. Experimental results show that compared with other ICL methods, we achieve the best performance on SVAMP, AQuA, Gsm8k and StrategyQA. Overall, our method improves the average accuracy of all benchmarks by 2.24%. This result shows that our method effectively improves the model's reasoning performance. To further demonstrate the effectiveness of the method, we conducted multiple sets of experiments for illustration.

4.2.1 Analysis of Example Selection and Ordering

For the selection and ordering strategies of demonstration examples in ICL, we designed several sets of experiments to verify the effectiveness of our method.

Regarding example selection, since each query may match far more examples than the specified limit during the problem-solving logic analysis, it is necessary to analyze specific difficulty sampling strategies. We designed four difficulty sampling strategies: (1) **Prioritize simplicity**: This strategy selects easy examples first. (2) **Prioritize difficulty**: This strategy selects difficult examples first. (3) **Select randomly**: This strategy randomly selects examples of any difficulty. (4) **Prioritize diversity**: This strategy aims to select as many difficulty levels as possible, sampling at most one example from each difficulty level.

Regarding the ordering of examples, to validate the effectiveness of curriculum learning, we designed two sets of controlled experiments. Under the four sampling strategies mentioned above, we applied two ordering strategy: (1) **difficulty increasing ordering (w/ order)** and (2) **random ordering (w/o order)**.

The complete experimental results are shown in Table 2, and through analysis, we have made the following observations:

First, it can be noted from the table that the performance of the strategies using the problem-solving logic and curriculum learning approach generally outperforms AL-ICL. The prioritize diversity (w/ order) strategy significantly outperforms the others, achieving an average accuracy of 72.37%.

Furthermore, the importance of curriculum learning is highlighted in our findings. For prioritize diversity strategies, the effect of ordering is particularly pronounced. In contrast, the impact of

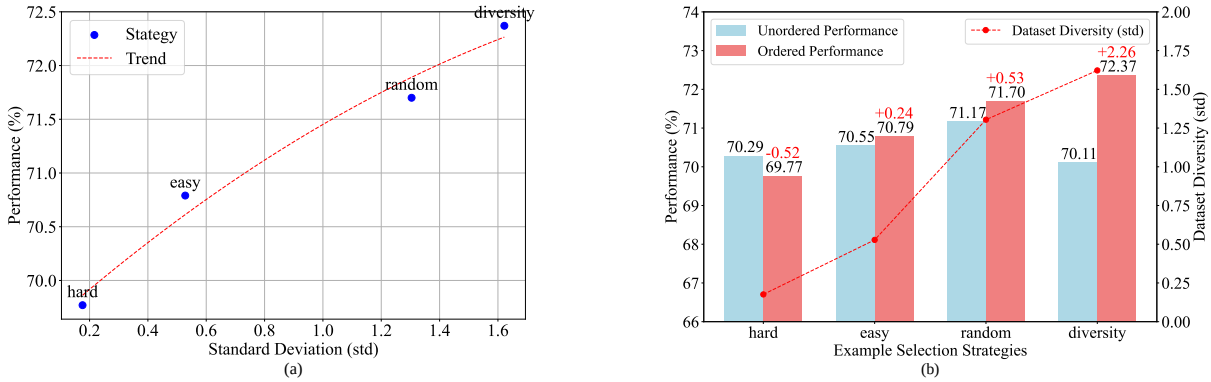

Figure 6: (a) shows the relationship between the average standard deviation of different example selection strategies and their performance across various benchmarks. (b) shows the impact of example ordering strategies on performance in relation to the average standard deviation under different selection strategies.

Strategy	Dataset					Avg.	Time
	SVAMP	AQuA	Gsm8k	ComSenQA	StrategyQA		
Fixed Examples	8	4	8	7	6	6.60	109%
Prioritize simplicity	7.27	4	7.73	7	5.88	6.38	117%
Prioritize difficulty	7.27	4	7.15	5.82	5.84	6.02	167%
Select Randomly	7.49	4	7.73	7	5.88	6.42	144%
Prioritize diversity	2.16	3.19	3.38	3.19	1.8	2.74	100%

Table 3: The number of demonstration examples selected by different selection strategies in benchmarks. **Avg** represents the average number of demonstration examples selected for each data. **Time** indicates the time cost comparison across different strategies. The highlighted part represent the strategy with most efficient.

ordering is less significant for the prioritize simplicity and prioritize difficulty strategies.

Based on the findings above and considering the characteristics of different selection strategies, we believe that the primary reason for these results is data diversity, or more specifically, difficulty diversity. To explain this phenomenon, we calculated the difficulty levels included in the demonstration examples for each data across all benchmarks and computed the average standard deviation. Standard deviation (std) is typically used to measure the degree of variation, and this metric helps illustrate the data diversity produced by different strategies.

We analyzed two sets of data: first, the relationship between difficulty diversity and strategy performance; and second, the impact of difficulty diversity on the four strategies, considering both the cases with and without ordering.

Figure 6-(a) depicts the relationship between performance and difficulty diversity across the four selection strategies. There is a clear positive correlation between difficulty diversity and performance, suggesting that data diversity is key to improving performance. Additionally, Figure 6-(b) shows the relationship between the performance difference (with and without ordering) and difficulty diversity across the four selection strategies. We found that ordering strategies are highly sensitive to difficulty diversity. Overall, the higher the difficulty diversity, the greater the improvement brought by ordering. Notably, the prioritize diversity strategy saw the largest performance improvement with ordering. This highlights the effectiveness of curriculum learning, where it is essential to order data according to difficulty. At the same time, it supports the idea that measuring example difficulty by the number of problem-solving steps is a valid approach.

4.2.2 Analysis of the Number of Examples

The number of demonstration examples for each query also has an important impact on the performance of ICL, as well as on the reasoning efficiency of LLMs. Table 3 presents the number of demonstration examples included with each test data across different strategies. For comparison, we use the fixed number of examples in CoT (Wei et al., 2022) as a reference.

We find that the prioritize diversity strategy has significantly superior performance while also hav-

ing the least average number of demonstration examples. The average number of demonstration examples for other strategies is more than 6, while priority diversity strategy only requires 2.74. Fewer examples indicate a shorter in-context length, which helps the reasoning speed of LLMs. Table 3 also presents the average time cost under different strategies. We uses the priority diversity strategy as the baseline at 100% to measure the time cost of other strategies. Experimental results show that, compared to other strategies, the prioritize diversity strategy has a time cost advantage, reducing consumption by 9% to 67%, effectively improving inference performance.

Current studies have shown that an increase in the number of demonstration examples usually leads to improved performance (Bertsch et al., 2024). Our method demonstrates that the quantity of examples is not the only influencing factor. This conclusion is consistent with numerous studies (Levy et al., 2023; Xie et al., 2024; Peng et al., 2024), which indicate that data diversity plays a critical role in enhancing the generalization capability of LLMs.

5 Related Work

5.1 In-Context Learning

GPT-3 (Brown et al., 2020) exhibited few-shot and zero-shot learning abilities during the pretraining phase. CoT (Wei et al., 2022) designed several fixed demonstration examples manually as in-context information, inspired further research on ICL (Yao et al., 2024).

Subsequent research has shown that the key to ICL lies in demonstration examples selection and ordering (Nguyen and Wong, 2023; Li and Qiu, 2023; Guo et al., 2024). Regarding example selection, AutoCoT (Zhang et al., 2022) used k-means clustering to select representative examples and leveraged zero-shot CoT to generate their reasoning process as demonstration examples. PromptSO (Shi et al., 2024) used principal component analysis (Abdi and Williams, 2010) to encode text and calculate similarity to select examples. Another work (Rubin et al., 2022) points out that a retriever can be trained using annotated data to determine whether an example is suitable for a query. Regarding example ordering, a study (Lu et al., 2022) randomly generated multiple combinations of example orderings to create probe sets. By analyzing the entropy of predicted labels for each probe set, the researchers selected the best-performing order. KATE (Liu et al., 2022) explored ordering examples based on task relevance as well as length-based sorting. Relevance-based ordering prioritizes examples closely related to the target task, while length-based sorting considers potential advantages for specific tasks.

5.2 Curriculum Learning in LLMs

Numerous applications across various fields have demonstrated that curriculum learning can effectively enhance model training outcome (Hacohen and Weinshall, 2019; Wang et al., 2021).

Currently, some works have applied curriculum learning to LLMs (Kim and Lee, 2024; Wang et al., 2024). A common approach is to train the model with examples progressing from easy to hard during fine-tuning. For instance, a study (Lee et al., 2023) conducted fine-tuning on a structured dataset that strictly covers multiple educational stages to simulate the progressive learning characteristics of humans. In the medical field, similarly, human-defined and automatically generated methods were used to annotate data difficulty, and LLMs in the medical question-answering domain were fine-tuned from easy to hard. (Lee et al., 2023). Additionally, another work (Pouransari et al., 2024) decomposed datasets into sequences of varying lengths, using sequence length as a metric to measure data difficulty.

Another common approach for applying curriculum learning to LLMs is ICL. For example, ICCL (Liu et al., 2024) utilized human experts or LLM-driven metrics to assess data difficulty, and gradually increased the difficulty of demonstration examples from easy to hard.

6 Conclusion

This paper proposes a problem-solving logic guided ICL strategy. By analyzing the problem-solving logic, we measure the similarity between problems and select demonstration examples. Additionally, the difficulty of problems is assessed based on the number of problem-solving steps, and the selected examples are ordered from easy to hard following the principles of curriculum learning. Experimental results across multiple benchmarks demonstrate that our proposed method outperforms other ICL methods in terms of average performance, significantly improving the reasoning capabilities of LLMs.

Limitations

Although our work improves the performance and efficiency of LLMs in reasoning tasks, there are still limitations for improvement. First, due to hardware resource constraints, we only conducted experiments on LLMs at the 8B scale, and further validation of our method is necessary on larger models, such as those at the 70B scale, to fully demonstrate its effectiveness. On the other hand, we observed in many-shot studies (Bertsch et al., 2024) that a significant increase in the number of examples leads to substantial improvements in reasoning performance. However, due to the limitations of benchmarks and hardware resources, we were unable to evaluate the effect of curriculum learning when applied to a large number of examples. We believe that when both the quantity and quality of examples are ensured, reasoning performance can be further improved, which will be a focus of our future work.

Potential Risks

Our work does not carry any obvious risks.

Acknowledgements

References

Hervé Abdi and Lynne J Williams. 2010. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459.

Shengnan An, Zeqi Lin, Qiang Fu, Bei Chen, Nanning Zheng, Jian-Guang Lou, and Dongmei Zhang. 2023. How do in-context examples affect compositional generalization? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11027–11052.

Aram Bahrini, Mohammadsadra Khamoshifar, Hossein Abbasimehr, Robert J Riggs, Maryam Esmaeili, Rastin Mastali Majdabadkohne, and Morteza Pasehvar. 2023. Chatgpt: Applications, opportunities, and threats. In *2023 Systems and Information Engineering Design Symposium (SIEDS)*, pages 274–279. IEEE.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Amanda Bertsch, Maor Ivgi, Uri Alon, Jonathan Berant, Matthew R Gormley, and Graham Neubig. 2024. In-context learning with long-context models: An in-depth exploration. *arXiv preprint arXiv:2405.00200*.

Satwik Bhattamishra, Arkil Patel, Phil Blunsom, and Varun Kanade. 2023. Understanding in-context learning in transformers and llms by learning to learn discrete functions. *arXiv preprint arXiv:2310.03016*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Xinlei Chen and Abhinav Gupta. 2015. Webly supervised learning of convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1431–1439.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. 2023. Why can gpt learn in-context? language models secretly perform gradient descent as meta-optimizers. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4005–4019.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. How abilities in large language models are affected by supervised fine-tuning data composition. *arXiv preprint arXiv:2310.05492*.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*.

Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. 2023. Guiding pretraining in reinforcement learning with large language models. In *International Conference on Machine Learning*, pages 8657–8677. PMLR.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Hila Gonen, Srini Iyer, Terra Blevins, Noah A Smith, and Luke Zettlemoyer. 2023. Demystifying prompts

in language models via perplexity estimation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10136–10148.

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

Qi Guo, Leiyu Wang, Yidong Wang, Wei Ye, and Shikun Zhang. 2024. What makes a good order of examples in in-context learning. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 14892–14904.

Guy Hacohen and Daphna Weinshall. 2019. On the power of curriculum learning in training deep networks. In *International conference on machine learning*, pages 2535–2544. PMLR.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173.

SU Hongjin, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, et al. 2022. Selective annotation makes language models better few-shot learners. In *The Eleventh International Conference on Learning Representations*.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017.

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 *International Conference on Learning Representations*.

Jisu Kim and Juhwan Lee. 2024. Strategic data ordering: Enhancing large language model performance through curriculum learning. *arXiv preprint arXiv:2405.07490*.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 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*, pages 1152–1157.

Bruce W Lee, Hyunsoo Cho, and Kang Min Yoo. 2023. Instruction tuning with human curriculum. *arXiv preprint arXiv:2310.09518*.

Itay Levy, Ben Bogin, and Jonathan Berant. 2023. Diverse demonstrations improve in-context compositional generalization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1401–1422.

Xiaonan Li and Xipeng Qiu. 2023. Finding support examples for in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6219–6235.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, William B Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for gpt-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114.

Yinpeng Liu, Jiawei Liu, Xiang Shi, Qikai Cheng, and Wei Lu. 2024. Let's learn step by step: Enhancing in-context learning ability with curriculum learning. *arXiv preprint arXiv:2402.10738*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098.

Katerina Margatina, Timo Schick, Nikolaos Aletras, and Jane Dwivedi-Yu. 2023a. Active learning principles for in-context learning with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5011–5034.

Katerina Margatina, Timo Schick, Nikolaos Aletras, and Jane Dwivedi-Yu. 2023b. Active learning principles for in-context learning with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5011–5034.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 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*, pages 975–984.

Tai Nguyen and Eric Wong. 2023. In-context example selection with influences. *arXiv preprint arXiv:2302.11042*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al.

2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 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*, pages 2080–2094.

Keqin Peng, Liang Ding, Yancheng Yuan, Xuebo Liu, Min Zhang, Yuanxin Ouyang, and Dacheng Tao. 2024. Revisiting demonstration selection strategies in in-context learning. *arXiv preprint arXiv:2401.12087*.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabás Poczós, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. 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)*, pages 1162–1172.

Hadi Pouransari, Chun-Liang Li, Jen-Hao Rick Chang, Pavan Kumar Anasosalu Vasu, Cem Koc, Vaishaal Shankar, and Oncel Tuzel. 2024. Dataset decomposition: Faster llm training with variable sequence length curriculum. *arXiv preprint arXiv:2405.13226*.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671.

Fobo Shi, Peijun Qing, Dong Yang, Nan Wang, Youbo Lei, Haonan Lu, Xiaodong Lin, and Duantengchuan Li. 2024. Prompt space optimizing few-shot reasoning success with large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1836–1862.

Petru Soviany, Claudiu Ardei, Radu Tudor Ionescu, and Marius Leordeanu. 2020. Image difficulty curriculum for generative adversarial networks (cugan). In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3463–3472.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. 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)*, pages 4149–4158.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Xin Wang, Yudong Chen, and Wenwu Zhu. 2021. A survey on curriculum learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):4555–4576.

Xin Wang, Yuwei Zhou, Hong Chen, and Wenwu Zhu. 2024. Curriculum learning: Theories, approaches, applications, tools, and future directions in the era of large language models. In *Companion Proceedings of the ACM on Web Conference 2024*, pages 1306–1310.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yunchao Wei, Xiaodan Liang, Yunpeng Chen, Xiaohui Shen, Ming-Ming Cheng, Jiashi Feng, Yao Zhao, and Shuicheng Yan. 2016. Stc: A simple to complex framework for weakly-supervised semantic segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2314–2320.

Noam Wies, Yoav Levine, and Amnon Shashua. 2024. The learnability of in-context learning. *Advances in Neural Information Processing Systems*, 36.

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *Transactions of the Association for Computational Linguistics*, 8:183–198.

Zhiyong Wu, Yaoxiang Wang, Jiacheng Ye, and Lingpeng Kong. 2023a. Self-adaptive in-context learning: An information compression perspective for in-context example selection and ordering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1423–1436.

Zhiyong Wu, Yaoxiang Wang, Jiacheng Ye, and Lingpeng Kong. 2023b. Self-adaptive in-context learning: An information compression perspective for in-context example selection and ordering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1423–1436.

Shan Xie, Man Luo, Chadly Daniel Stern, Mengnan Du, and Lu Cheng. 2024. Demoshapley: Valuation of demonstrations for in-context learning. *arXiv preprint arXiv:2410.07523*.

Xin Xu, Yue Liu, Panupong Pasupat, Mehran Kazemi, et al. 2024. In-context learning with retrieved demonstrations for language models: A survey. *arXiv preprint arXiv:2401.11624*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Junjie Ye, Xuanting Chen, Nuo Xu, Can Zu, Zekai Shao, Shichun Liu, Yuhan Cui, Zeyang Zhou, Chao Gong, Yang Shen, et al. 2023. A comprehensive capability analysis of gpt-3 and gpt-3.5 series models. *arXiv preprint arXiv:2303.10420*.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*.

Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew Chi-Chih Yao. 2023. Cumulative reasoning with large language models. *arXiv preprint arXiv:2308.04371*.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, YeYanhan YeYanhan, and Zheyan Luo. 2024. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410, Bangkok, Thailand. Association for Computational Linguistics.

A BREAK Dataset Description

BREAK is a dataset proposed by the Allen Institute (Wolfson et al., 2020). This work introduces the Question Decomposition Meaning Representation (QDMR), which breaks down a question into several sub-questions for solving and represents it as a sequence of steps. The dataset collects 60,150 question and QDMR pairs from several public datasets. To represent various questions as a unified sequence of steps, they customized 13 types of operations, converting the solution process for all questions into sequences of these operations. The specific operations and their templates are shown in Table 4. The decomposition and formalization of questions can be found in Figure 1 and Figure 2. Table 5 shows the distribution of operations in the BREAK dataset, that is, the proportion of each operation appearing in a single data point. Table 6 shows the distribution of the total number of sub-questions after decomposition in the dataset.

Based on the BREAK dataset, we constructed an instruction set to analyze the problem-solving logic. Specific examples and explanations of the instruction set are provided in Table 7.

B Fine-Tuning Details

We performed LoRA fine-tuning on the Llama3-8B model using the aforementioned instruction set. The specific hyperparameters are as follows: the `cutoff_len` is set to 1024, the learning rate is set to 5×10^{-5}, the fine-tuning parameters are specified as `all`, `lora_rank` is set to 8, `lora_alpha` is set to 16, the optimizer used is AdamW, the model is trained for 4 epochs, and the best model is selected based on the BLEU score.

C Prompt Template

Table 8 shows the prompt templates used for fine-tuning problem-solving logic analysis.
Table 9–13 shows the full prompt example for in-context learning on the different benchmarks.

D Supplementary Details

Our experiments utilized the llama-factory (Zheng et al., 2024) project, which includes model fine-tuning and in-context learning. The CPU used in the experiments is an Intel(R) Xeon(R) Platinum 8358 CPU @ 2.60GHz, and the GPU is an NVIDIA Tesla A800 80G. The hyperparameters were set according to the default configuration file provided by llama-factory. The prompt length was set to 4096, and the maximum answer output length was set to 1024. To ensure output stability, the temperature was set to 0.01. In our study, we used ChatGPT to assist in coding.

Operator	Template / Signature	Question	Decomposition
Select	Return [entities] $w \rightarrow S_e$	How many touchdowns were scored overall?	1. Return touchdowns 2. Return the number of #1
Filter	Return [ref] [condition] $S_o, w \rightarrow S_o$	I would like a flight from Toronto to San Diego please.	1. Return flights 2. Return #1 from Toronto 3. Return #2 to San Diego
Project	Return [relation] of [ref] $w, S_e \rightarrow S_o$	Who is the head coach of the Los Angeles Lakers?	1. Return the Los Angeles Lakers 2. Return the head coach of #1
Aggregate	Return [aggregate] of [ref] $w_{agg}, S_o \rightarrow n$	How many states border Colorado?	1. Return Colorado 2. Return border states of #1 3. Return the number of #2
Group	Return [aggregate] [ref1] for each [ref2] $w_{agg}, S_o, S_e \rightarrow S_n$	How many female students are there in each club?	1. Return clubs 2. Return female students of #1 3. Return the number of #2 for each #1
Superlative	Return [ref1] where [ref2] is [highest / lowest] $S_e, S_n, w_{sup} \rightarrow S_e$	What is the keyword, which has been contained by the most number of papers?	1. Return papers 2. Return keywords of #1 3. Return the number of #1 for each #2 4. Return #2 where #3 is highest
Comparative	Return [ref1] where [ref2] [comparison] [number] $S_e, S_n, w_{com}, n \rightarrow S_e$	Who are the authors who have more than 500 papers?	1. Return authors 2. Return papers of #1 3. Return the number of #2 for each of #1 4. Return #1 where #3 is more than 500
Union	Return [ref1] , [ref2] $S_o, S_o \rightarrow S_o$	Tell me who the president and vice-president are?	1. Return the president 2. Return the vice-president 3. Return #1 , #2
Intersection	Return [relation] in both [ref1] and [ref2] $w, S_e, S_e \rightarrow S_o$	Show the parties that have representatives in both New York state and representatives in Pennsylvania state.	1. Return representatives 2. Return #1 in New York state 3. Return #1 in Pennsylvania state 4. Return parties in both #2 and #3
Discard	Return [ref1] besides [ref2] $S_o, S_o \rightarrow S_o$	Find the professors who are not playing Canoeing.	1. Return professors 2. Return #1 playing Canoeing 3. Return #1 besides #2
Sort	Return [ref1] sorted by [ref2] $S_e, S_n \rightarrow \langle e_1 ... e_k \rangle$	Find all information about student addresses, and sort by monthly rental.	1. Return students 2. Return addresses of #1 3. Return monthly rental of #2 4. Return #2 sorted by #3
Boolean	Return [if / is] [ref1] [condition] [ref2] $S_o, w, S_o \rightarrow b$	Were Scott Derrickson and Ed Wood of the same nationality?	... 3. Return the nationality of #1 4. Return the nationality of #2 5. Return if #3 is the same as #4
Arithmetic	Return the [arithmetic] of [ref1] and [ref2] $w_{ari}, n, n \rightarrow n$	How many more red objects are there than blue objects?	... 3. Return the number of #1 4. Return the number of #2 5. Return the difference of #3 and #4

Table 4: The 13 operator types of QDMR steps. Listed are, the natural language template used to express the operator, the operator signature and an example question that uses the query operator in its decomposition.

Operator	QDMR
SELECT	100%
PROJECT	69.0%
FILTER	53.2%
AGGREGATE	38.1%
BOOLEAN	30.0%
COMPARATIVE	17.0%
GROUP	9.7%
SUPERLATIVE	6.3%
UNION	5.5%
ARITHMETIC	5.4%
DISCARD	3.2%
INTERSECTION	2.7%
SORT	0.9%
Total	60,150

Table 5: Operator prevalence in BREAK, that is, the proportion of each operator appearing in a single data point.

Steps	QDMR
1-2	10.7%
3-4	44.9%
5-6	27.0%
7-8	10.1%
9+	7.4%

Table 6: The distribution of the total number of QDMR sub-questions.

Input
The input is a problem to be solved, such as: what flights are available tomorrow from denver to philadelphia?
Label
\\ *The label contains <operator> and <formal language>.* \\ *<operator> is an ordered set composed of the aforementioned custom operations.* \\ *<formal language> is the formalized language that provides a detailed description of each operator.* <operators>: ['select', 'filter', 'filter', 'filter'] <formal language>: ["SELECT['flights']", "FILTER['#1', 'from denver']", "FILTER['#2', 'to philadelphia']", "FILTER['#3', 'if available']"]

Table 7: Examples and Explanation of Instruction Sets Based on the BREAK Dataset

Prompt
You are a helpful assistant. Please break down in order the operations <operations> required to solve the following problems, and the process of solving the problem according to the operations <programs>: what flights are available tomorrow from denver to philadelphia?
Label
<operators>: ['select', 'filter', 'filter', 'filter'] <formal language>: ["SELECT['flights']", "FILTER['#1', 'from denver']", "FILTER['#2', 'to philadelphia']", "FILTER['#3', 'if available']"]

Table 8: Fine-tuning prompts for problem-solving logic analysis

Prompt

System prompt

Please provide the answer in the following format: "The final answer is <answer>"

User input

question: Being his favorite, he saved checking on the grapevines for his last stop. He was told by 235 of the pickers that they fill 100 drums of raspberries per day and 221 drums of grapes per day. How many drums of grapes would be filled in 77 days?
answer: Equation is (221.0 * 77.0). The final answer is 17017.0

question: Tiffany was collecting cans for recycling. On Monday she had 4 bags of cans. The next day she found some more bags worth of cans. If she had a total of 6 bags altogether, how many bags did she find on the next day?
answer: Equation is (6.0 - 4.0). The final answer is 2.0

question: After a typhoon, 13 trees in Haley's backyard died. If she had grown 3 trees initially, how many more trees died in the typhoon than those that survived?
answer: Equation is (13.0 - (3.0 - 13.0)). The final answer is 23.0

question: Brenda's mother made cookies for 5 people. She prepared 22 cookies but had to throw away 17 cookies. If each of them had the same number of cookies, how many did each of them have?
answer: Equation is ((22.0 - 17.0) / 5.0). The final answer is 1.0

question: Haley grew 9 trees in her backyard. After a typhoon 4 died. Then she grew 5 more trees. How many trees does she have left?

Number of problem-solving logic step:

4->5->6->7->7

Table 9: Full prompt example for in-context learning on the SVAMP dataset

Prompt

System prompt

Please provide the answer in the following format: "The final answer is <answer>"

User input

question: What is the probability of rolling one six-sided dice, and getting a different number on each die? A)1/12 B)1/3 C)1 D)5/9 E)7/18
answer: Let's think step by step. MAGOOSHOFFICIAL SOLUTION. For the first die, we can roll any one of six numbers. For the second die, we can roll any number save for the number we rolled on the first die, giving us 5 possibilities. For the third die, we can roll four different numbers (we can't roll the number we rolled on the first or second die). 6 possibilities out of 216 total possibilities. 6/6 = 1. The final answer is C.

question: If q is the square of a positive integer, which of the following must be equal to the square of the next positive integer? A)$\sqrt{n+1}$ B)$n+1$ C)n^2+1 D)$q+2\sqrt{q}+1$ E)n^2+2n+1
answer: Let's think step by step. If q is the square of a positive integer, then $q=(x)^2$ where x is a positive integer. To calculate, $(x+1)^2=x^2+2x+1$, which is $q+2\sqrt{q}+1$. This should be D. The final answer is D.

question: If Tim had lunch at $50 and he gave 20% tip, how much did he spend? A)$60.00 B)$35.42 C)$60.60 D)$21.56 E)$78.45
answer: Let's think step by step. The tip is 20% of what he paid for lunch. Tip = 20% of 50.00 = $10.00. Total spent = 50.00 + 10.00 = $60.00. The final answer is A.

question: Carl is facing very difficult financial times and can only pay the interest on a $10,000 loan he has taken. The bank charges him a quarterly compound rate of 4%. What is the approximate interest he pays annually? A)$1600 B)$2000 C)$2150 D)$2500 E)$12000
answer: Let's think step by step. The bank charges a 4% quarterly compounded annual rate. Per quarter rate is (16/4)% = 4%. Thus, the quarterly compounded interest will be slightly more than $1600. The final answer is A.

question: A shopkeeper employed a servant at a monthly salary of 1500. In addition to it, he agreed to pay him a commission of 15% on the monthly sale. How much sale in Rupees should the servant do if he wants his monthly income as 6000? A)30000 B)415000 C)31500 D)50000 E)None of these

Number of problem-solving logic step:

2->3->4->5->6

Table 10: Full prompt example for in-context learning on the AQuA dataset

Prompt

System prompt

Please provide the answer in the following format: "The final answer is <answer>"

User input

question: A shopkeeper bought 150 packets of milk. Each packet contained 250 ml of milk. If one fluid ounce is equal to 30 ml, how many ounces of milk did he buy?
nanswer: Let's think step by step. If the shopkeeper bought 150 packets of milk, each packet containing 250ml of milk, all the packets had a total of 250*150 =«150*250=37500»37500ml.Since one ounce equal 30 ml, the total amount of milk that the shopkeeper bought in oz is 37500/30=«37500/30=1250»1250 oz of milk. The final answer is 1250

question: Twenty gallons of tea were poured into 80 containers. Geraldo drank 3.5 containers. How many pints of tea did Geraldo drink?
answer: Let's think step by step. 20 gallons = 160 pints. 160/80 = «160/80=2»2 pints.3.5 * 2 pints = «3.5*2=7»7 pints. Geraldo drank 7 pints of tea. The final answer is 7

question: During the holidays, Lance works as a merchandiser. He works 35 hours a week, spread equally over 5 workdays. If Lance earns $9 an hour, how much does he make on each workday?
answer: Let's think step by step. Lance works 35 / 5 = «35/5=7»7 hours a day. So he makes $9 x 7 = $«9*7=63»63 on each workday. The final answer is 63

question: A snack machine accepts only quarters. Candy bars cost ¢25, each piece of chocolate costs ¢75, and a pack of juice costs ¢50. How many quarters are needed to buy three candy bars, two pieces of chocolate, and one pack of juice?

answer: Let's think step by step. Three candy bars cost ¢25 x 3 = ¢«25*3=75»75. Two pieces of chocolate cost ¢75 x 2 = ¢«75*2=150»150. So, the total amount needed to buy those is ¢75 + ¢150 + ¢50 = ¢«75+150+50=275»275. Since a quarter is equal to ¢25, therefore ¢275/¢25 = «275/25=11»11 quarters are needed. The final answer is 11

question: Mark makes custom dog beds. A bed for a Rottweiler takes 8 pounds of stuffing, a bed for a chihuahua takes 2 pounds of stuffing, and a bed for a collie takes the average amount of stuffing between the first two kinds of beds. How many pounds of stuffing does Mark need to make 4 chihuahua beds and 3 collie beds?

Number of problem-solving logic step:

5->6->7->8->8

Table 11: Full prompt example for in-context learning on the Gsm8k dataset

Prompt

System prompt

Please provide the answer in the following format: "The final answer is <answer>"

User input

question: What is the only was to recover from exhaustion? A. mediate B. have rest C. stay in bed D. run out of steam E. go to sleep
answer: B

question: Google Maps and other highway and street GPS services have replaced what? A. united states B. mexico C. countryside D. atlas E. oceans
answer: D

question: You can share files with someone if you have a connection to a what? A. freeway B. radio C. wires D. computer network E. electrical circuit
answer: D

question: If a person isn't able to pay their bills what must they do? A. know everything B. acknowledgment C. make more money D. throw a party E. spare time

Number of problem-solving logic step:

1->2->3->3

Table 12: Full prompt example for in-context learning on the ComSenQA dataset

Prompt

System prompt

Please provide the answer in the following format: "The final answer is yes or no"

User input

question: Can you buy Casio products at Petco?
answer: Casio is a manufacturer of consumer electronics and watches. Petco is a chain store that sells pet supplies like food, bowls, litter, toys, cages and grooming equipment. The final answer is no

question: Did Clark Gable appear in any movies scored by John Williams?
answer: Clark Gable died in 1960. John Williams scored his first movie in 1961. The final answer is no

question: Could a dandelion suffer from hepatitis?
answer: Only creatures that contain a liver can suffer from hepatitis. The liver is an organ only found in vertebrates. Vertebrates exist in the kingdom Animalia. Dandelions are plants in the kingdom Plantae. The final answer is no

question: Did Mozart ever buy anything from Dolce & Gabbana?

Number of problem-solving logic step:

2->3->4->4

Table 13: Full prompt example for in-context learning on the StrategyQA dataset