Learning by Analogy: Enhancing Few-Shot Prompting for Math Word Problem Solving with Computational Graph-Based Retrieval

Anonymous EMNLP submission

Abstract

Large language models (LLMs) are known to struggle with complicated reasoning tasks such as math word problems (MWPs). In this paper, we present how analogy from similarly structured questions can improve LLMs' problem-solving capabilities for MWPs. Specifically, we rely on the retrieval of problems with similar *computational graphs* to the given question to serve as exemplars in the prompt, providing the *correct reasoning path* for the generation model to refer to. Empirical results across six math word problem datasets demonstrate the effectiveness of our proposed method, which achieves a significant improvement of up to 6.7 percent on average in absolute value, compared to baseline methods. These results highlight our method's potential in addressing the reasoning challenges in current LLMs.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable success across a wide range of tasks (Achiam et al., 2023; Dubey et al., 2024; Jiang et al., 2023; Labrak et al., 2024; Lin et al., 2024). However, solving math word problems (MWPs) remains a significant challenge for LLMs (Ahn et al., 2024; Srivatsa and Kochmar, 2024). Unlike tasks that primarily rely on linguistic or general knowledge, MWPs demand a nuanced integration of language comprehension and mathematical reasoning, posing unique difficulties for LLMs. Overcoming this challenge is critical, as proficiency in solving MWPs could expand the applications of LLMs to education, automated tutoring, and complex reasoning tasks.

Human problem-solving for MWPs offers an insightful source of inspiration. People often solve new problems by analogy, leveraging prior examples to adapt solutions to novel scenarios. Inspired by this analogy-driven learning process, recent research has employed few-shot prompting techniques to enhance MWP performance in LLMs (Jiang et al., 2023; Melz, 2023; Henkel et al., 2024). Most existing approaches for selecting few-shot examples rely either on random selection (Jiang et al., 2023; Dubey et al., 2024) or retrieval based solely on semantic similarity (Huang et al., 2023; Melz, 2023; Henkel et al., 2024). Although providing examples can improve LLM performance, these methods often fail to ensure that the selected examples align with the mathematical structure of the target problem. Specifically, randomly selected examples lack relevance to the target problem, while semantic retrieval tends to prioritize superficial linguistic similarity over deep structural alignment. This mismatch between the provided examples and the target problem ultimately constrains the effectiveness of LLMs in solving MWPs.

To address this limitation, we propose a novel computational graph-based retrieval method for selecting examples that align more closely with the underlying structure of the target math word problem. Our approach identifies examples with computational graphs that are structurally similar to the target problem and incorporates these examples into few-shot prompting, providing the LLM with more relevant problem-solving guidance. Specifically, we design a lightweight retriever model trained using contrastive learning to identify structurally analogous examples. Examples with similar graphs are treated as positive pairs, while those with dissimilar graphs are treated as negative pairs. Once trained, the retriever can be seamlessly integrated into the LLM inference workflow without requiring updates to the LLM's parameters, making our approach modular and easily adaptable. We evaluate our method on six math word problem datasets, demonstrating that our computational graph-based retrieval approach achieves significant performance improvements over both semantic-based retrieval and random selection baselines. Furthermore, we conduct case studies and

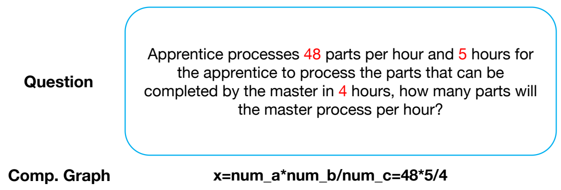

Figure 1: An example of a math word problem with its computational graph.

detailed analyses to highlight the effectiveness of our method.

Our contributions are summarized as follows:

- **Proposing Computational Graph-Based Retrieval for Few-Shot Prompting.** We introduce a computational graph-based retrieval method specifically tailored for math word problem-solving. This approach selects examples with structural similarity to the target problem, enhancing few-shot prompting by providing LLMs with examples that align with the underlying mathematical structure of the problem.

- **Training a Structural Similarity Retriever.** We develop a retriever model trained with contrastive learning to identify structural similarity in math word problems. This lightweight and modular retriever integrates seamlessly into the LLM inference workflow without requiring parameter updates to the LLM itself.

- **Conducting Extensive Evaluation and Analysis.** We conduct comprehensive experiments on six math word problem datasets, demonstrating that our approach significantly outperforms both semantic-based and random selection baselines, with average exact matching (EM) score improvements of up to 6.7% and 19.5% respectively. Additionally, we present in-depth case studies and analyses to validate the effectiveness of our method in capturing structural nuances essential for MWP-solving. We also provide an automated approach to construct the training data without any human labors.

2 Methodology

2.1 Overview of the Proposed Framework

When solving a new reasoning problem, humans often draw upon known problems with similar reasoning paths and address them by analogy. In the context of math word problems, the reasoning path corresponds to its computational graph, as illustrated in Figure 1. Large language models (LLMs) are observed to fail to conduct genuine logical reasoning (Mirzadeh et al., 2024) and exhibit strong token biases (Li et al., 2024) when addressing reasoning tasks. Therefore, providing LLMs with the *correct reasoning path* from analogous problems can guide them to mimic the problem-solving process. This paper aims to develop a math word problem-solving system comprising a retriever and a generator. The retriever identifies problems and solutions with computational graphs similar to the query problem from a corpus, while the generator leverages these retrieved exemplars through in-context learning to enhance problem-solving performance.

2.2 Retriever Model Training

Figure 2 shows the training process of the retriever. Given a batch of questions $\{q_i\}_{i=1}^n$ and their corresponding computational graphs $\{G_i\}_{i=1}^n$, we search in the training dataset for positive examples $\{q_i^+\}_{i=1}^n$ where their computational graphs are the same as those of the query questions: $G_i^+ = G_i,\ i = 1, 2, ...n$ where n is the batch size.[1] Then we forward the $\{q_i, q_i^+\}_{i=1}^n$ with the retriever (an encoder model) $f_{\theta r}$ to get the embeddings $\{f_{\theta r}(q_i), f_{\theta r}(q_i^+)\}_{i=1}^n$. By applying infoNCE loss (Oord et al., 2018) with the in-batch negative strategy, the training loss objective L of the retriever becomes:

$$L = \frac{1}{n}\sum_{i=1}^{n} -\log(e^{\mathrm{sim}(f_{\theta r}(q_i), f_{\theta r}(q_i^+))/\tau} / \\ (\sum_{j=1, j\neq i}^{n} e^{\mathrm{sim}(f_{\theta r}(q_i), f_{\theta r}(q_j))/\tau} + \\ \sum_{j=1}^{n} e^{\mathrm{sim}(f_{\theta r}(q_i), f_{\theta r}(q_j^+))/\tau})) \quad (1)$$

where sim indicates a similarity function and τ is the temperature. Note that we do not need to train the generator.

2.3 Inference

Given a trained retriever $f_{\theta r^*}$, a question-solution pair corpus C and a given question q, the retriever select the top-k similar question-solution

[1]We discard the examples if there's no positive samples matched in the training dataset.

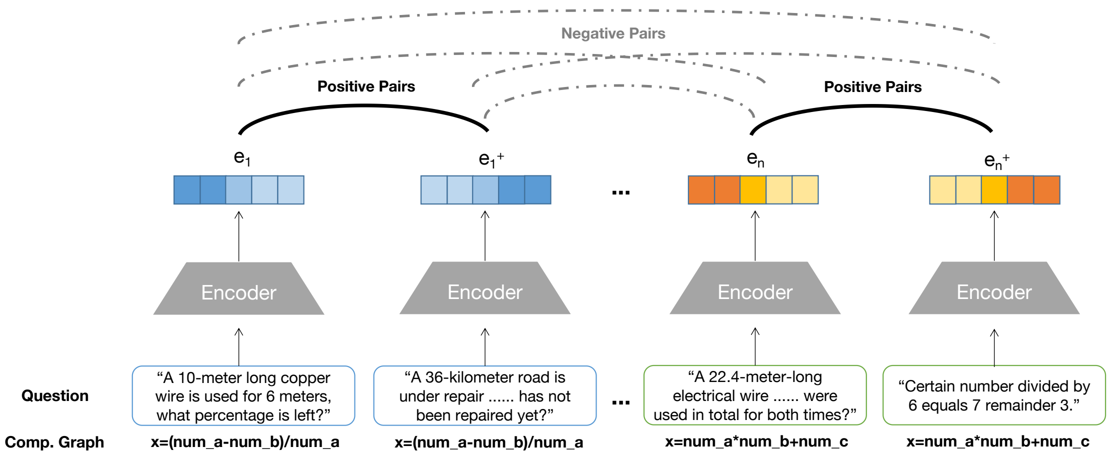

Figure 2: **Flowchart of Retriver Training.** This figure illustrates the process of training a retriever model (encoder) with contrastive learning to identify structurally similar math word problems. Each question is encoded into an embedding based on its text. Positive pairs are formed by pairing examples with matching computational graph structures, while in-batch negatives serve as contrasting examples with different structures.

pairs $\{q_i, a_i\}_{i=1}^k$ based on the similarity score:

$$\{q_i, a_i\}_{i=1}^k = topk(\text{sim}(f_{\theta r^*}(q), f_{\theta r^*}(q_j))) \quad (2)$$

where $q_j \in C$. Then we concatenate the retrieved question-answer pairs and the given question as the prompt to the generator $f_{\theta g}$ to get the output answer a:

$$a = f_{\theta g}(concat(q_1, a_1, ..., q_k, a_k, q)) \quad (3)$$

where $concat$ denotes the concatenation operation.

3 Experiment

3.1 Setup

Implementation Details. In our experiments, we use BGE-large-en-v1.5 (Xiao et al., 2023) as retriever and LLaMA-3 model series (Dubey et al., 2024) as generator for English datasets (except for 0.5B size experiments, where we use Qwen2.5-0.5B-Instruct as the generator since no similar sized LLaMA-3 model is available), and BGE-large-zh-v1.5 as retriever and Qwen2.5 model series (Team, 2024) as generator for Chinese datasets, with bfloat16 precision for all models. We add an extra pooler (a two-layer MLP module) to the retriever, following the practice in (Chen et al., 2020). We train the retriever on 25% randomly selected data from Math23k training set[2] (Wang et al., 2017) where the computational graphs are provided, using AdamW (Loshchilov and Hutter, 2019) with a learning rate of 3e-5 for 5 epochs, a temperature τ of 0.05, and cosine similarity as the similarity function. We set the batch size equal to 16 for the training process.

[2]Math23k dataset is provided in Chinese, and we use LLaMA-3.1-70B-Instruct to translate it into English for the training of English model.

Datasets. We evaluate our retrieval-generation system on the following six math word problem datasets: Math23k (Wang et al., 2017), ape210k (Zhao et al., 2020), gsm8k (Cobbe et al., 2021), math_qa (Amini et al., 2019), Calc-ape210k (Kadlčík et al., 2023) and aqua_rat (Ling et al., 2017), as shown in Table 1. For all datasets, we use the corresponding training set as the retrieval corpus and evaluate on the test set, and use $k = 8$ for top-k example retrieval.

Metrics. We report exact match (EM) accuracy for all datasets. During inference, we require the generator to generate answers following the same format of the given exemplars to facilitate the parsing of the solution to obtain the final answer, and consider the generated solution correct if the parsed final answer matches the golden answer. We use string matching for the datasets where the solutions are provided in text format, and use float number matching if the solutions are provided in equation format.

3.2 Main Results

Table 2 presents a detailed summary of our experimental results, highlighting the superiority of our method across various datasets and model sizes. Specifically, for the Chinese datasets Math23k and

	# Samples (train/val/test)	Language	Solution type	Comp. Graph	Options
Math23k	21.2k/1k/1k	ZH	Equation	✓	✗
ape210k	200.5k/5k/5k	ZH	Equation	✗	✗
gsm8k	7.5k/-/1.3k	EN	Text	✗	✗
math_qa	29.8k/4.5k/3.0k	EN	Text	✗	✓
Calc-ape210k	195k/1.8k/1.8k	EN	Equation	✗	✗
aqua_rat	97.5k/254/254	EN	Text	✗	✓

Table 1: Details of datasets evaluated."ZH" and "EN" refers to Chinese and English. An example of equation solution is "*x=(5*1000)-2000*" where x is the final answer, and an example of text solution is "*Natalia sold 48/2 = «48/2=24»24 clips in May. Natalia sold 48+24 = «48+24=72»72 clips altogether in April and May. #### 72.*"Options refer to if the candidate answers are provided in the question.

	Math23k	ape210k	gsm8k	math_qa	Calc-ape210k	aqua_rat	Avg.
Random$_{\text{Qwen-0.5B}}$	28.9	19.2	17.1	16.5	12.0	18.1	18.6
BGE$_{\text{Qwen-0.5B}}$	43.1	39.7	21.2	**27.3**	17.6	16.9	27.6
Ours$_{\text{Qwen-0.5B}}$	**57.6**	**49.2**	**22.7**	26.6	**30.5**	**18.9**	**34.3**
Random$_{\text{LLaMA-1B/Qwen-1.5B}}$	50.3	32.7	38.6	17.2	22.8	14.2	27.6
BGE$_{\text{LLaMA-1B/Qwen-1.5B}}$	58.7	50.4	38.7	45.9	20.4	29.9	40.7
Ours$_{\text{LLaMA-1B/Qwen-1.5B}}$	**66.6**	**59.2**	**40.7**	**47.3**	**31.3**	**37.4**	**47.1**
Random$_{\text{LLaMA-3B/Qwen-3B}}$	68.0	44.3	71.4	52.9	32.6	46.9	52.7
BGE$_{\text{LLaMA-3B/Qwen-3B}}$	73.1	54.6	71.5	**64.9**	31.5	50.0	57.6
Ours$_{\text{LLaMA-3B/Qwen-3B}}$	**78.3**	**59.9**	**71.9**	64.3	**39.8**	**50.6**	**60.8**
Random$_{\text{LLaMA-8B/Qwen-7B}}$	83.9	62.8	80.1	51.3	30.6	49.6	59.7
BGE$_{\text{LLaMA-8B/Qwen-7B}}$	87.6	73.8	**80.4**	66.4	39.5	49.6	66.2
Ours$_{\text{LLaMA-8B/Qwen-7B}}$	**90.4**	**76.7**	79.2	**66.8**	**46.5**	**53.1**	**68.8**
Random$_{\text{LLaMA-70B/Qwen-72B}}$	84.7	68.9	84.7	60.6	39.3	59.8	66.3
BGE$_{\text{LLaMA-70B/Qwen-72B}}$	90.9	79.5	86.0	**68.5**	47.9	**64.2**	72.8
Ours$_{\text{LLaMA-70B/Qwen-72B}}$	**92.4**	**80.9**	**87.3**	68.0	**53.5**	**64.2**	**74.4**

Table 2: Main results of our system. We report exact match (EM) for all tasks. Our approach outperforms the baselines on most tasks except for math_qa, which is because the semantic similarity and computational graph similarity are overlapped in this dataset. While our method is effective for generators of all sizes, the performance gain is larger for smaller models.

ape210k, our approach consistently and significantly outperforms both the random and BGE baselines. Similarly, strong performance gains are observed across four English datasets, further demonstrating the effectiveness of our method. The only exception is the math_qa dataset, where our method performs comparably to the BGE baseline. This anomaly arises because, in math_qa, the semantic similarity often coincides with computational graph similarity. Many example pairs in this dataset differ only in the numerical values while maintaining identical semantic structures and computational graphs (e.g., *"The banker's gain of a certain sum due 3 years hence at 10% per annum is Rs. 36. What is the present worth?"* and *"The banker's gain of a certain sum due 2 years hence at 10% per annum is Rs. 24. What is the present worth?"*). Since these pairs exhibit similar semantics and identical computational graphs at the same time, the BGE model can effectively retrieve them by focusing solely on semantic similarity, leaving little room for improvement through retriever training. Furthermore, our method demonstrates larger performance gains when the generator model is smaller in size. This could be attributed to the enhanced reasoning capabilities of larger LLMs, which allow them to solve problems more independently, reducing their reliance on retrieving similar examples.

3.3 Ablation Study

In this section we conduct ablation study to prove that the performance gain in Table 2 is due to the utilization of computational graph structures, instead of additional training on the Math23k dataset. We compare the performance of retriever trained with computational graphs and retriever trained on semantic similarity, both on Math23k dataset. Since there are no explicit labels on this dataset for contrastive training, in our main experiments we constructed positive pairs based on the similarity of computational graphs, and similarly, we construct positive pairs based on the semantic similarity criteria for this ablation study. Specifically, we use a strong embedding model to calculate the semantic similarity of samples on Math23k to get pseudo labels. We utilize gte-Qwen2-7B-Instruct (Li et al., 2023), the best open-source multilingual embedding model on the MTEB benchmark to calculate the semantic similarity of each sample with all other samples, and choose the most similar sample as the positive. Then we use these pseudo-labeled data to train our retriever. We denote this retriever as *semantic retriever*. As shown in Table 3, we find that the semantic retriever results in a similar performance to the BGE baseline, which indicates that the performance gain of our approach does come from the utilization of computational graphs instead of training on Math23k dataset alone.

3.4 Analysis

3.4.1 The Performance Upper Bound

In this work, we hypothesize that problems with similar computational graphs can facilitate answering the given question. Under this assumption, the upper bound of our method's performance is achieved by using computational graphs directly for retrieval. Since computational graphs are available only on Math23k dataset, we focus on this dataset to compare the upper bound performance with performance of our trained retriever, thereby evaluating the quality of the retriever training process. To measure similarity between computational graphs for retrieval, we utilize the normalized Levenshtein Distance, which quantifies the string-based similarity of computational graph representations. Table 4 compares the performance of our method against the hypothesized upper bound. The results indicate that, compared to the original BGE model, our trained retriever achieves performance significantly closer to the upper bound. This highlights the effectiveness of our training approach in improving retrieval quality.

3.4.2 Case Study on Retrieved Data

Next, we present a case study on the retrieved data from Calc-ape210k using both our trained model and the BGE model. As shown in Figure 3, for the query question, *"The 'Scientist' series is 2.5 yuan/book, and the 'Inventor' series is 4 yuan/book. It costs a total of 22 yuan to buy two sets of books. There are 4 'Scientists', how many books are there in the 'Inventor' series?"*, our trained retriever successfully retrieves examples with similar computational graphs, even though the semantics of these examples are quite different. In contrast, the original BGE model relies primarily on semantic similarity for retrieval. As illustrated in the figure, while all the retrieved questions in the BGE model relate to "books", their computational graphs are entirely different from the query's graph. Additionally, we include a scatter plot on the Math23k dataset, where we analyze the correlation between computational graph similarity and embedding similarity for the top-8 retrieved data points from 100 random samples, as depicted in Figure 4. The results show that the Pearson correlation coefficient for our trained model is significantly higher than that for the BGE model, indicating that our approach is more effective in retrieving examples with similar computational graphs based on question embeddings.

3.4.3 Effect of Corpus Choice

Finally, we investigate whether the choice of retrieval corpus affects performance. Specifically, we explore the case where data with the same distribution as the query are not available to serve as the corpus, a scenario that is common in real-world applications. In this experiment, we use the SuperCLUE-Math6 dataset (Xu et al., 2024), where only the test set is available, and select the training set from ape210k dataset as the retrieval corpus. The results, shown in Table 5, demonstrate that our approach remains effective even when the corpus and the query data do not share the same distribution. This suggests that, despite the different data distributions, our trained retriever can still find problems with similar computational graphs in the large ape210k corpus. This capability indicates that our method can be applied in a broad and flexible manner, making it suitable for various real-world scenarios.

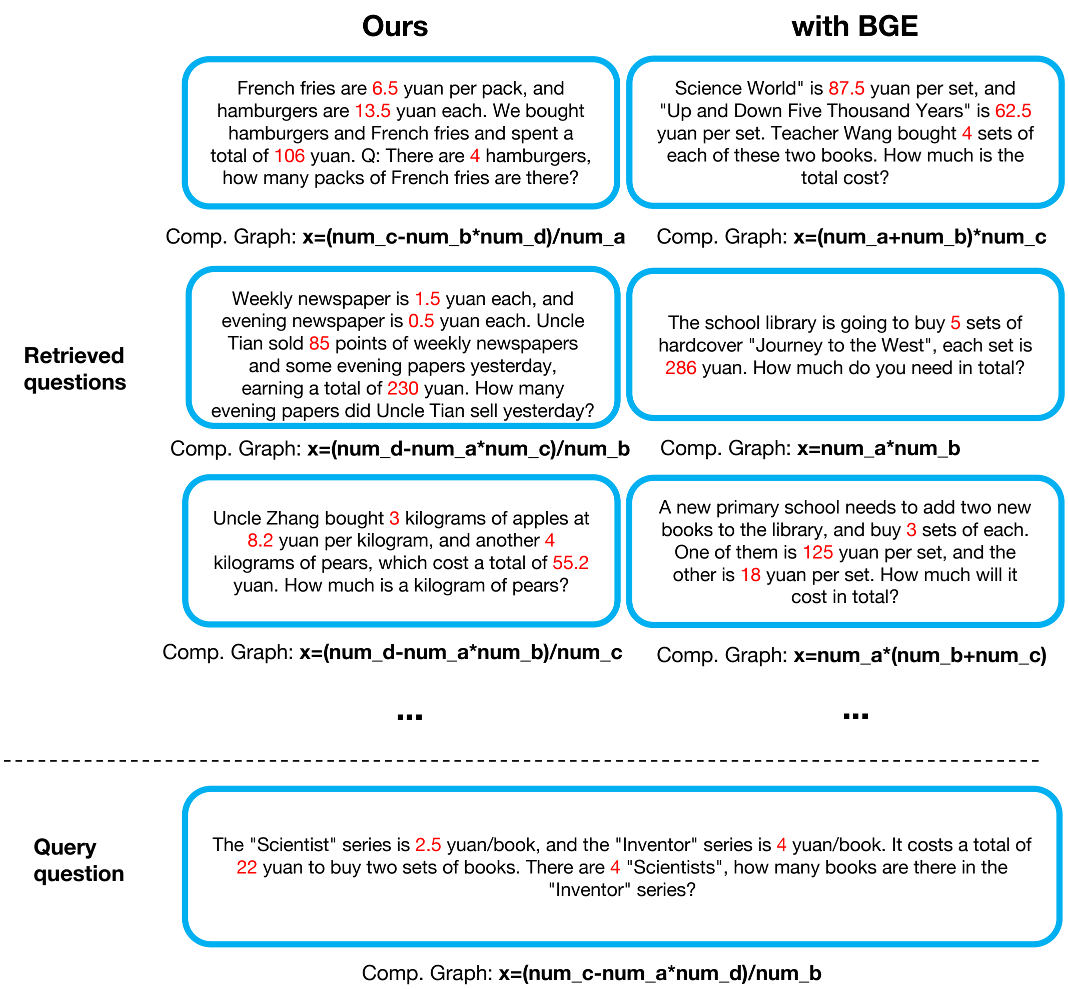

Figure 3: Case study on the retrieved data with our model and BGE respectively. The retrieved data using trained retriever have similar computational graphs with the query question, while the computational graphs are different for retrieved data using BGE model.

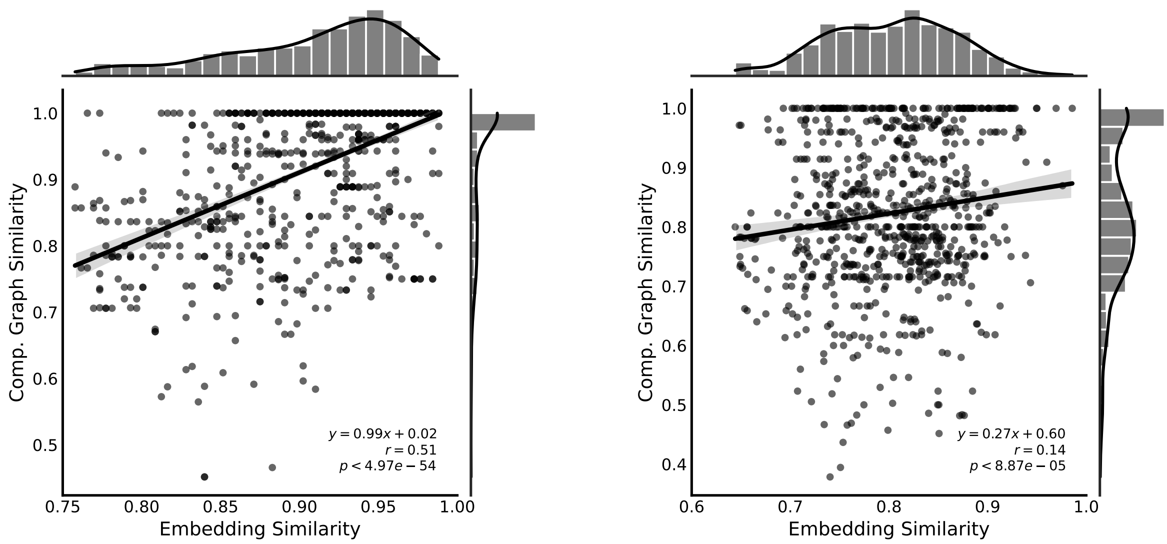

Figure 4: The scatter plot of our trained retriever (left) and BGE (right) on 100 random samples from Math23k. There is a stronger positive correlation between computational graph similarity and embedding similarity for data with trained retriever than the BGE model.

	Math23k	ape210k	gsm8k	math_qa	calc_ape210k	aqua_rat	Avg.
Semantic Retriever$_{\text{Qwen-0.5B}}$	44.0	40.8	21.3	26.1	22.1	13.4	28.0
BGE$_{\text{Qwen-0.5B}}$	43.1	39.7	21.2	**27.3**	17.6	16.9	27.6
Ours$_{\text{Qwen-0.5B}}$	**57.6**	**49.2**	**22.7**	26.6	**30.5**	**18.9**	**34.3**

Table 3: Results of ablation study. *Semantic retriever* refers to the retriever trained with semantic similarity data on Math23k, which results in a similar performance to the BGE baseline, indicating that the performance gain of our approach does come from the utilization of computational graphs instead of training on Math23k dataset alone.

	Ours	BGE	Upper Bound
Math23k	66.6	58.7	**68.2**

Table 4: Comparison of our methods with the upper bound with Qwen2.5 1.5B model. Our approach results in a large performance gain compared to the original BGE model and a score close to the upper bound, suggesting the effectiveness of our training process.

	Ours	BGE	Random
SuperCLUE-Math6	**27.2**	20.6	18.6

Table 5: Results with Qwen2.5 0.5B model on SuperCLUE-Math6 test set. Here we use the training set of ape210k as the retrieval corpus, as the training set of SuperCLUE-Math6 is not availale. The results suggest that our approach is robust in the case where the distributions of the corpus and the task data are different.

4 Computational Graph-Free Training Data Acquisition

Notably, in our training pipeline, we only need data pairs that contain either the same or different computational graphs, rather than requiring the computational graphs themselves. This allows us to avoid the explicit need for computational graph annotations. Instead, we can leverage large language models (LLMs), such as Claude-3.5 or GPT-4 (OpenAI et al., 2024), to generate training data.

To do this, we prompt the LLM to rewrite the questions so that all details, such as numerical values and entity names, differ from the original question, while maintaining the same computational graph. We use the following prompt of this rewritting: *"Generate a problem with the same computation graph as the input math problem, ensuring that the semantics, numerical values, and sentence structure are as different as possible. Output only one rewritten example, without any additional information."* We randomly select 5,000 samples from the training set of gsm8k and use this approach to generate 5,000 positive pairs to train the retriever. The downstream results, shown in Table 6, indicate that while the retriever trained with distilled data performs slightly below that trained with labeled data, it consistently outperforms the BGE baseline, demonstrating the effectiveness of the distilled data. Examples of this rewriting process are presented in Figure 5. Empirically, we observe that the sentence structure before and after rewriting is more similar than in the labeled data pairs, which the retriever may rely on to capture similarity between positive pairs during training, rather than focusing on the true computational graphs.

5 Related Work

Few-shot Prompting for MWP Solving. Large Language Models have shown promising results in tackling math word problems (Toshniwal et al.; Yang et al., 2024; Yu et al., 2024a; Mirzadeh et al., 2024; Wei et al., 2022b). To enhance model performance on math word problems, few-shot prompting has become a widely adopted approach (Wei et al., 2022b; Jiang et al., 2023; Melz, 2023; Henkel et al., 2024). Existing methods for example selection generally fall into two categories: semantic similarity-based retrieval (Huang et al., 2023; Melz, 2023; Henkel et al., 2024) and random selection (Wei et al., 2022b; Jiang et al., 2023; Dubey et al., 2024). By contrast, our approach leverages a computational graph-based retrieval strategy. Rather than relying solely on superficial linguistic features, our method retrieves examples that match the mathematical structure of the target problem. This structurally informed selection enables LLMs to draw from examples that better align with the mathematical reasoning required.

Retrieval-Augmented Generation. Retrieval-Augmented Generation (RAG) has recently gained attention to improve the quality of LLM outputs by integrating relevant external information during generation (Lewis et al., 2020; Gao et al., 2023; Fan et al., 2024). For math word problems, Henkel et al. (2024) proposed a RAG system by retrieving con-

	gsm8k	math_qa	Calc-ape210k	aqua_rat	Avg.
BGE	38.7	45.9	20.4	29.9	33.7
Ours$_{\text{w/ labeled data}}$	**40.7**	**47.3**	**31.3**	**37.4**	**39.2**
Ours$_{\text{w/ distillation data}}$	39.4	46.4	27.5	35.0	37.1

Table 6: Results with training data distilled from GPT-4o with LLaMA-3.2-1B-Instruct generator. Retriever trained with distilled data outperforms the BGE baseline while underperforms the model trained with labeled data on all tasks.

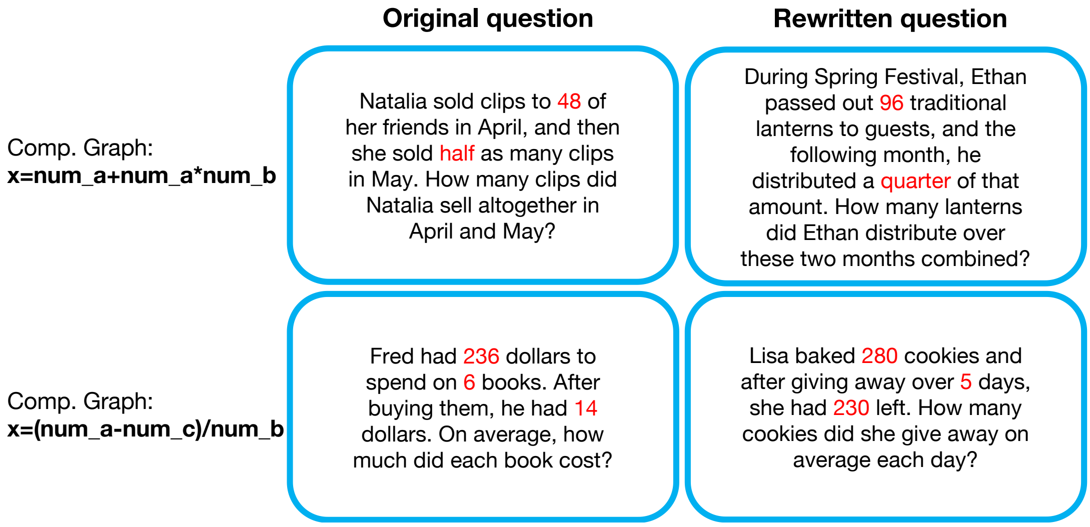

Figure 5: Some cases of the original and rewritten questions. The entity names, value of numbers and semantics are different after rewritting, while the computational graphs remain the same.

tent from an open-source math textbook. Similarly, Dixit and Oates (2024) introduced a schema-based RAG framework for math word problems, using structured schemas to guide LLMs in selecting appropriate mathematical operations, ultimately enhancing reasoning clarity and problem-solving structure. Our framework can also be viewed as a RAG system, where the corpus consists of structurally relevant MWP examples.

Reasoning Ability in LLMs. Large language models (LLMs) have often been criticized for lacking "system 2" thinking ability (Yu et al., 2024b), which limits their performance on complex reasoning tasks. Many prior studies have raised concerns about the "genuine" reasoning capabilities of current LLMs (Hazra et al., 2024; Wei et al., 2022a), noting that LLMs struggle to distinguish between causality and correlation (Ashwani et al., 2024) and are not strong abstract reasoners (Gendron et al., 2024). These findings suggest that, despite their extensive pretraining on large-scale corpora, current LLMs are essentially pattern matchers (Mirzadeh et al., 2024). While reasoning ability can be partially elicited through prompt engineering techniques like Chain-of-Thought (Wei et al., 2022b), this paper explores an alternative approach—providing the LLM with pre-existing reasoning paths rather than relying on the model to generate them independently.

6 Conclusion

In this work, we have explored computational graph-based retrieval for solving math word problems, drawing inspiration from the analogy of reasoning paths between similarly structured problems. Our experiments on both English and Chinese math datasets demonstrate the effectiveness of our approach across models of different scales, with performance gains being more pronounced for smaller models. Additionally, by leveraging LLMs, we can automatically construct training data without relying on human labor. We hope this paper inspires future research on tackling a variety of reasoning tasks, extending beyond math word problems.

Limitations

Our computational graph-based retrieval method has demonstrated significant improvements in solving math word problems (MWPs). However, our experiments focus on MWPs, where problems can be represented as computational graphs. It remains unclear whether this approach can generalize effectively to more complex mathematical problems, such as formal proofs or multi-step algebraic derivations, which may require different forms of structural reasoning. And its applicability to non-mathematical reasoning tasks, such as commonsense reasoning or scientific problem-solving, has not been explored. Additionally, our method relies on the acquisition of training data with computational graphs, posing extra costs.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 225–237.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. 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 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Swagata Ashwani, Kshiteesh Hegde, Nishith Reddy Mannuru, Mayank Jindal, Dushyant Singh Sengar, Krishna Chaitanya Rao Kathala, Dishant Banga, Vinija Jain, and Aman Chadha. 2024. Cause and effect: Can large language models truly understand causality? *Preprint*, arXiv:2402.18139.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. *Preprint*, arXiv:2002.05709.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Prakhar Dixit and Tim Oates. 2024. Sbi-rag: Enhancing math word problem solving for students through schema-based instruction and retrieval-augmented generation. *arXiv preprint arXiv:2410.13293*.

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*.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6491–6501.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Gaël Gendron, Qiming Bao, Michael Witbrock, and Gillian Dobbie. 2024. Large language models are not strong abstract reasoners. *Preprint*, arXiv:2305.19555.

Rishi Hazra, Gabriele Venturato, Pedro Zuidberg Dos Martires, and Luc De Raedt. 2024. Can large language models reason? a characterization via 3-sat. *Preprint*, arXiv:2408.07215.

Owen Henkel, Zach Levonian, Chenglu Li, and Millie Postle. 2024. Retrieval-augmented generation to improve math question-answering: Trade-offs between groundedness and human preference. In *Proceedings of the 17th International Conference on Educational Data Mining*, pages 315–320.

Xijie Huang, Li Lyna Zhang, Kwang-Ting Cheng, and Mao Yang. 2023. Boosting llm reasoning: Push the limits of few-shot learning with reinforced in-context pruning. *arXiv e-prints*, pages arXiv–2312.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Marek Kadlčík, Michal Štefánik, Ondřej Sotolář, and Vlastimil Martinek. 2023. Calc-x and calcformers: Empowering arithmetical chain-of-thought through interaction with symbolic systems. In *Proceedings of the The 2023 Conference on Empirical Methods in Natural Language Processing: Main track*, Singapore, Singapore. Association for Computational Linguistics.

Yanis Labrak, Adrien Bazoge, Emmanuel Morin, Pierre-Antoine Gourraud, Mickael Rouvier, and Richard Dufour. 2024. Biomistral: A collection of open-source pretrained large language models for medical domains. *arXiv preprint arXiv:2402.10373*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Bangzheng Li, Ben Zhou, Fei Wang, Xingyu Fu, Dan Roth, and Muhao Chen. 2024. Deceptive semantic shortcuts on reasoning chains: How far can models go without hallucination? In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7675–7688, Mexico City, Mexico. Association for Computational Linguistics.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Jiacheng Lin, Hanwen Xu, Zifeng Wang, Sheng Wang, and Jimeng Sun. 2024. Panacea: A foundation model for clinical trial search, summarization, design, and recruitment. *medRxiv*, pages 2024–06.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *ACL*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. *Preprint*, arXiv:1711.05101.

Eric Melz. 2023. Enhancing llm intelligence with arm-rag: Auxiliary rationale memory for retrieval augmented generation. *arXiv preprint arXiv:2311.04177*.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2024. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *Preprint*, arXiv:2410.05229.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, et al. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Kv Aditya Srivatsa and Ekaterina Kochmar. 2024. What makes math word problems challenging for llms? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1138–1148.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Shubham Toshniwal, Wei Du, Ivan Moshkov, Branislav Kisacanin, Alexan Ayrapetyan, and Igor Gitman. Openmathinstruct-2: Accelerating ai for math with massive open-source instruction data. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24*.

Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep neural solver for math word problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 845–854, Copenhagen, Denmark. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *Preprint*, arXiv:2206.07682.

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

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding. *Preprint*, arXiv:2309.07597.

Liang Xu, Hang Xue, Lei Zhu, and Kangkang Zhao. 2024. Superclue-math6: Graded multi-step math reasoning benchmark for llms in chinese. *Preprint*, arXiv:2401.11819.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. 2024. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024a. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. 2024b. Distilling system 2 into system 1. *Preprint*, arXiv:2407.06023.

Wei Zhao, Mingyue Shang, Yang Liu, Liang Wang, and Jingming Liu. 2020. Ape210k: A large-scale and template-rich dataset of math word problems. *Preprint*, arXiv:2009.11506.