Kashihara, A. et al. (Eds.) (2024). Proceedings of the 32nd International Conference on Computers in Education. Asia-Pacific Society for Computers in Education

UniSpLLM: An Integrated Approach for Enhancing Reasoning and Education with Large Language Models

Hanyu ZHAO[a], Yuzhuo WU[a], Yang YU[a], Xiaohua YU[b] & Liangyu CHEN[a*]
[a]*Shanghai Key Lab of Trustworthy Computing, East China Normal University, China*
[b]*Department of Education Information Technology, East China Normal University, China*
* lychen@sei.ecnu.edu.cn

Abstract: Large Language Models (LLMs) show constrained performance when confronted with a set of mathematical problems spanning various knowledge concepts. Unlike natural language tasks, the understanding and solution strategies for math problems significantly vary, presenting a great challenge for LLMs to consistently generate precise solutions for different problem types. To address this limitation, we propose UniSpLLM(A **Uni**versal Template integrated with **Sp**ecific methods using **L**arge **L**anguage **M**odels), a strategy devised to bolster LLMs' efficiency in solving problems enriched with varied knowledge concepts. UniSpLLM innovates by crafting a *Universal Template*, versatile enough to accommodate any problem type. Specifically, we design six *Specific Method*s that can be adapted to different problem types. **UniSpLLM** distinguishes itself from other prompt-based approaches, which typically cater to a singular problem type. By achieving an improvement of nearly 15% on the latest dataset TAL_SAQ6K_EN from AAAI2024 and surpassing the GPT-4 baseline by almost 17% on the MMLU-Math dataset, UniSpLLM significantly elevates the utility of LLMs within educational fields. Our approach and results hold significant educational value, as they aid students in acquiring diverse thinking skills tailored to various problem types.
Keywords: Large Language Models, Prompt-Based Approach, Math Education, Mathematical Reasoning, Math Word Problems Solving.

1. Introduction

The massive scaling up of Large Language Models (LLMs) has significantly propelled Natural Language Processing (NLP), particularly enhancing the solving of Math Word Problems (MWPs). Advanced LLMs like MathSensei (Das et al., 2024) and GPT-4 (Achiam et al., 2023) have significantly improved the automated solution of MWPs.

When students tackle complex MWPs, they need a variety of thinking skills such as knowledge transfer (Adams, 2014), Self Regulation (Vula et al., 2017), Thinking Step by Step (Zhang et al., 2022), Computational-aided Reasoning (Barana et al., 2017), Systematic Thinking (Tretter, 2010), Recall and Integration (Huang et al., 2021). To meet these needs, we develop six strategies, each targeting a specific thinking skill to enhance problem-solving accuracy. ChatGPT should also see increased accuracy when using these strategies. To show this, we apply these strategies to enhance GPT-4 with problem-solving abilities, similar to how we would cultivate students critical thinking and problem-solving skills. We introduce **UniSpLLM**, a comprehensive approach using GPT-4 to solve MWPs. This method provides a systematic solution and specialized prompt techniques that enhance students' cognitive abilities. Our key contributions are as follows.

The remainder of this paper is organized as follows. We first explain the structure and each module of UniSpLLM in detail (§2). Then, we conduct experiments and analyze each specific method's results (§3). Finally, we conclude this paper (§4).

2. Methodology

2.1 Overview of UniSpLLM

Our approach UniSpLLM is described in Figure 1. The approach can be divided into three phases:

1. **Problem Classifier.** Problems are first categorized into different types with specific knowledge concepts. This phase not only helps to find a similar problem for in-context learning but also presents the knowledge concepts to guide LLMs using the proper background knowledge.
2. **Prompt Constructor.** In this phase, for each problem, a special prompt is constructed according to its problem type and the knowledge concepts. The prompt will be input into GPT-4 with the problem to trigger the reasoning capacity of GPT-4.
3. **Answer Generator.** The last phase is designed for post-processing GPT-4 output. If Python codes are output, UniSpLLM will call a Python interpreter to execute codes for the result. If the output is in common text, UniSpLLM will use regular expressions to extract the results.

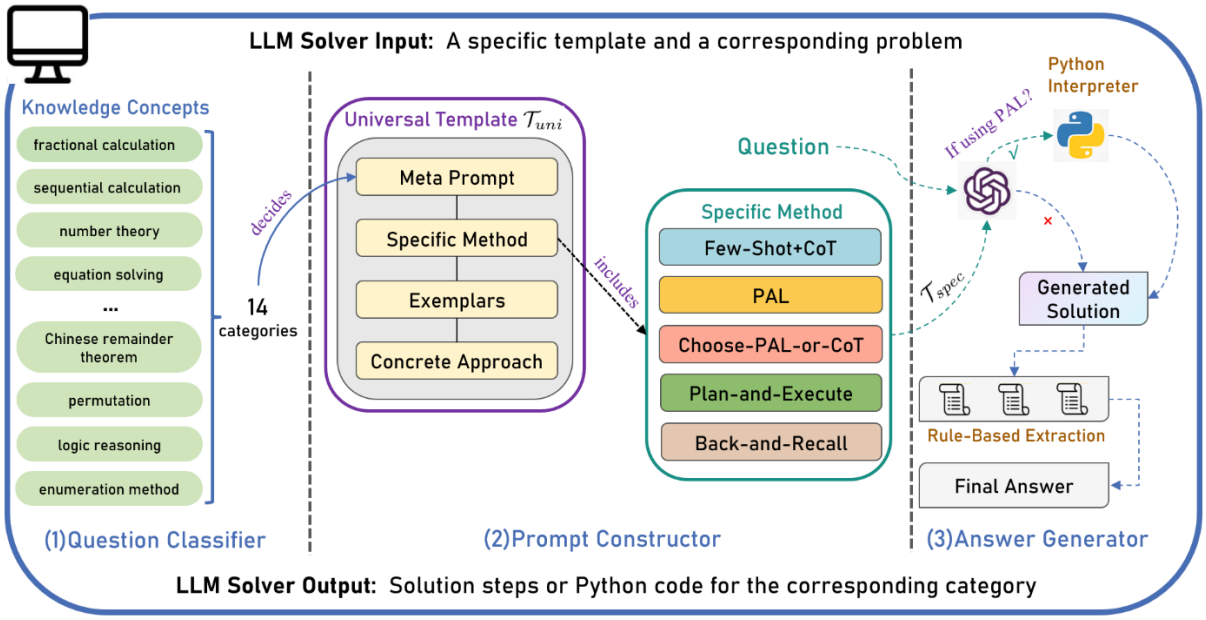

Figure 1. The architecture of UniSpLLM.

2.2 Module of Problem Classifier

In this module, we employ the KMeans algorithm (Ahmed et al., 2020) to partition the dataset into N clusters based on Term Frequency-Inverse Document Frequency (TF-IDF) vectors.

2.3 Module of Problem Classifier

2.3.1 Universal Template

In the Module of Question Classifier, the problem type C_i is identified with knowledge concept K, which then guides the selection of the *Universal Template* T_{uni}, which comprises the following four parts listed as follows.

1. **Meta Prompt:** A meta prompt directs GPT-4 to use certain knowledge concepts such as "*Let's use knowledge concept* K *to solve math word problems*".
2. **Specific Method:** A prompt guides GPT-4 to use which solving method. Upon specifying the Specific Method Prompt, T_{uni} transitions into T_{spec}.
3. **Exemplars:** Examples that align with specific knowledge concepts K.

4. **Concrete Approach:** Instructs GPT-4 on whether to use Python code.

2.3.2 Principles of Choosing Specific Method M and Students's Thinking Skills

We propose that students resemble LLMs in their MWP-solving processes. Just as LLMs require certain prompts during MWP-solving, so do students. To solve MWPs effectively, students should develop six thinking skills listed accordingly. Corresponding to these skills, we construct T_{spec} using six specific methods M, which also enhance the MWP-solving accuracy of LLMs and are assessed for their potential utility in student learning.

1. **CoT with Self-Regulation (§2.3.3)**: Students should practice **Self-Regulation**, a skill that involves self-reviewing their problem-solving steps and restarting their reasoning to correct any mistakes, thus aiding in maintaining accuracy and consistency (Vula et al., 2017).
2. **Few-Shot+CoT (§2.3.4):** Students should develop the skill of **Think Step by Step**, which involves thinking step by step (Zhang et al., 2022). Additionally, they should learn the skill of **Knowledge Transfer**, which involves applying mathematical concepts from one problem to another using examples (Adams, 2014).
3. **PAL (§2.3.5):** Students may utilize external aids such as calculators to ensure accuracy, referred as **Computational-aided Reasoning** (Barana et al., 2017). We use Program-Aided Language (PAL) to enable LLMs to utilize Python code for precise calculations.
4. **Plan-and-Execute (§2.3.7):** When faced with complex problems involving numerous variables, students should systematically plan their approach rather than resort to aimless solving. This thinking skill is named **Systematic Thinking** (Tretter, 2010). So we implement Plan-and-Execute, allowing LLMs to plan, list variables systematically, and execute accordingly.
5. **Back-and-Recall (§2.3.8):** Students should effectively utilize previous results by remembering and integrating them into subsequent reasoning steps. This thinking skill is named **Recall and Integration** (Huang et al., 2021). We employ Back-and-Recall, enabling LLMs to retrieve previous results for smoother reasoning steps and avoid the disadvantage of linear CoT.

2.3.3 CoT with self-Regulation

This approach guides the model with a Chain of Thought (CoT) to generate a preliminary reasoning path. If an intermediate check reveals errors in the derived solution, the model discards the incorrect path and generates a new one. This iterative process mimics human cognitive self-review during problem-solving and is formalized as Equation 1:

$$\hat{y}_{final} = \text{ReReason}\left(I, \{f(x_i, C_i, y_i)\}_{i=1}^{k}, \text{if Check}(y_{inter}) = \text{false then Restart}(x_{k+1})\right). \quad (1)$$

2.3.4 Few-Shot+CoT

Few-shot with Chain of Thought (Few-shot+CoT) is a method where LLMs learn new tasks and make predictions based on a few examples. Equation 2 formalizes this approach, showing how LLMs use task descriptions (I) and demonstrations $\{(f(x_i, C_i, y_i))\}$. Each demonstration includes a sequence of steps (C_i) guiding the model's thinking.

$$\hat{y}_{k+1} = \sum_{i=1}^{k} \alpha_i \cdot LLM(I, f(x_i, C_i, y_i)) + LLM(I, f(x_{k+1}, C_{k+1}, \_)). \quad (2)$$

2.3.5 PAL

Program-Aided Language (PAL) (Gao et al., 2023) incorporates programming as an intermediate reasoning step. Inspired by this concept, PAL tasks are understood and solved

by generating and executing Python code. The model's task is to generate the corresponding Python code that leads to the solution.

2.3.6 PAL-or-CoT

Inspired by the principles from both PAL and CoT, we propose an adaptive method that chooses the method leading to the most accurate and comprehensible solution. LLMs are prompted to choose from two previously solved methods, PAL and CoT, and determine which is more accurate. If neither is satisfactory, LLMs are prompted to solve from scratch.

2.3.7 Plan-and-Execute

While CoT and PAL methods effectively solve MWPs through step-by-step thinking or code generation, they sometimes overlook the need for deep understanding and systematic planning. We design the *Plan-and-Execute* strategy, which emphasizes thorough analysis and meticulous planning before problem-solving. The brief template is shown in Table 1.

Table 1. *The brief template of Plam-and-Execute method.*

Let us first understand the problem, extract relevant **variables** and their corresponding values, and devise a **plan**. Then, let us carry out the plan, **calculate** intermediate results (pay attention to calculation and common sense), solve the problem step by step, and show the **answer**.

2.3.8 Back-and-Recall

Linear CoT has disadvantages, which may involve forgetting previous key information in multi-step reasoning. Thus, we design the Back-and-Recall using residual connections prompt (Jiang et al., 2023) to let GPT-4 recall the previous steps. This approach ensures continuous tracing of important details throughout the problem-solving steps, creating a *memory* in the reasoning chain. This method enhances logical continuity and comprehensiveness.

3. Experiment and Results

3.1 Dataset and Baselines

We first evaluate UniSpLLM on the TAL-SAQ6K-EN dataset[1] from the AAAI2024 Global Competition on Math Problem Solving and Reasoning. This dataset hosts 5927 English math problems from global school contests. Additionally, we incorporate the MMLU-Math \autocite{MMLU} dataset. The baselines for the TAL-SAQ6K-EN dataset are listed as follows: GPT-4 achieves an accuracy of 69.52%, GPT-3.5 attains 50.84%, and MathGPT reaches 43.70%. These results are detailed on the website for the AAAI2024 competition[2]. For the MMLU-math dataset, our baseline is set using GPT-4, which achieves an accuracy of 63.64%.

3.2 Experiments and Results

3.2.1 Accuracy Comparison

Implementing different prompting strategies, we observe the following improvements

[1] TAL_SAQ6K-EN dataset: https://github.com/math-eval/aaai2024comp

[2] https://ai4ed.cc/competitions/aaai2024competition

in accuracy with UniSpLLM as shown in Table 2. We also summarize which problem types are best for PAL or CoT and their accuracy with the PAL-or-CoT automatic selection method, detailed in Table 3.

All figures and tables must be referred to in your text. See Table 1 and Figure 1 as examples. Table should be marked header row in the Table Design. Figures should have meaningful Alt-text to describe the contents in the picture setting. All figures and tables should be centered. Table captions are italicized and aligned left *above* the table and with major words capitalized with no full stop. Figure captions are centered and placed *below* the figure. Please leave one blank line with Style ICCE Normal Text (1st paragraph) before every table caption or figure. Similarly, please leave one blank line with Style ICCE Normal Text (1st paragraph) after every table or figure caption.

Table 2. *Accuracy on TAL_SAQ6K_EN and MMLU-Math. 5-shot means using 5 samples. One-shot means using an example.*

METHOD	TAL_SAQ6K_EN	MMLU-Math
Baseline	69.52%	63.64%
Zero-Shot+CoT with self-Regulation	67.55%	82.54%
Few-Shot+CoT with self-Regulation	77.66%(5-shot)	86.00%(one-shot)
Few-Shot+PAL	79.32%(5-shot)	-
PAL-or-CoT	83.63%(5-shot)	-
PAL-or-CoT+Plan	84.62%(5-shot)	-
PAL-or-CoT+Plan+Back-and-Recall	85.61%(5-shot)	-

Table 3. *Comparison of method accuracies. C indicates cluster ID. N the number in each cluster. Combine denotes accuracy using the PAL-or-CoT method. The highest accuracy is bolded, with a green checkmark highlighting superior results between COT and PAL.*

Knowledge Concept	C	N	CoT	PAL	Combine
Positional Value, Proportional Calculation	0	474	59.07%	71.88%✓	**82.85%**
Four Foundamental Operations	1	569	75.15%✓	75.10%	**76.50%**
Number Theory Principles	4	496	71.93%	**82.67%**✓	77.30%
Applied Mathematics, Basic Arithmetic	5	428	**85.75%**✓	84.09%	84.92%
Continuous Calculation, Equation Setting	9	267	82.77%	**88.87%**✓	85.82%
Chinese Remainder Theorem, Enumeration	10	117	49.57%	**80.73%**✓	65.15%
Equations, Logical Reasoning	11	2046	79.42%	80.20%✓	**87.80%**
Permutations, Combinations	12	518	60.23%	64.26%✓	**70.24%**
Enumeration Method	13	74	81.08%✓	79.73%	**86.49%**

3.3 Results Analysis and Educational Discussion

Our meticulously refined prompt construction workflows and templates have the potential to enhance AI-driven educational products. By utilizing our workflow, these products can offer accurate answers and facilitate the development and reinforcement of students' thinking skills.

3.4 Dataset Contribution

We present a dataset **OpenMWP** featuring diverse solving methods tailored for different types of math word problems, filling the gap left by the previous TAL-SAQ6K-EN dataset provided by the competition, which lacked disclosed answers. OpenMWP not only offers answers but also provides multiple solving strategies, each paired with corresponding thinking skill that students can learn. This dataset facilitates the training of open-source large language models

(LLMs) such as ChatGLM. By training these models with our dataset, they can learn to respond to problems using our six specific methods. The dataset is available at our github [3].

4. Conclusion & Future work

In this paper, we introduce **UniSpLLM**, an innovative and unified architecture designed to enhance the mathematical reasoning capabilities of LLMs, with profound implications for education. Our approach involves several key components: categorizing datasets, assigning knowledge concepts, and deploying a Universal Template. This template provides six specialized prompt templates tailored for various mathematical problem types, enabling UniSpLLM to proficiently address a wide range of mathematical problems by integrating knowledge concepts with specific solution methods.

Future explorations should delve into whether the six thinking skills we use encompass all the cognitive demands students face when solving problems. Additionally, further investigations are needed to identify additional thinking skills required and determine the appropriate weighting for each.

Acknowledgements

This work is supported by NSFC (No. 62272416), the National Key Research Project of China(No. 2023YFA1009402) and National Education Sciences Planning Project (grant number BCA240048). Additionally, we appreciate TAL for organizing the AAAI2024 Global Competition on Math Problem Solving and Reasoning competition.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). Gpt-4 technical report. arXiv preprint arXiv:2303.08774.

Adams, K. (2014). The effect of students' mathematical beliefs on knowledge transfer. Brigham Young University.

Ahmed, M., Seraj, R., & Islam, S. M. S. (2020). The k-means algorithm: A comprehensive survey and performance evaluation. Electronics, 9(8), 1295.

Barana, A., Fioravera, M., & Marchisio, M. (2017). Developing problem solving competences through the resolution of contextualized problems with an advanced computing environment. Proceedings of the 3rd International Conference on Higher Education Advances, 1015–1023.

Das, D., Banerjee, D., Aditya, S., & Kulkarni, A. (2024). Mathsensei: A tool-augmented large language model for mathematical reasoning. arXiv preprint arXiv:2402.17231.

Gao, L., Madaan, A., Zhou, S., Alon, U., Liu, P., Yang, Y., Callan, J., & Neubig, G. (2023). Pal: Program-aided language models. International Conference on Machine Learning, 10764–10799.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., & Steinhardt, J. (2020). Measuring massive multitask language understanding. arXiv preprint arXiv:2009.03300.

Huang, S., Wang, J., Xu, J., Cao, D., & Yang, M. (2021). Recall and learn: A memory-augmented solver for math word problems. arXiv preprint arXiv:2109.13112.

Jiang, S., Shakeri, Z., Chan, A., Sanjabi, M., Firooz, H., Xia, Y., Akyildiz, B., Sun, Y., Li, J., Wang, Q., et al. (2023). Resprompt: Residual connection prompting advances multi-step reasoning in large language models. arXiv preprint arXiv:2310.04743.

Tretter, T. R. (2010). Systematic and sustained: Powerful approaches for enhancing deep mathematical thinking. Gifted Child Today, 33(1), 16–26.

Vula, E., Avdyli, R., Berisha, V., Saqipi, B., & Elezi, S. (2017). The impact of metacognitive strategies and self-regulating processes of solving math word problems. International Electronic Journal of Elementary Education, 10(1), 49–59.

Zhang, Z., Zhang, A., Li, M., & Smola, A. (2022). Automatic chain of thought prompting in large language models. arXiv preprint arXiv:2210.03493.

[3] OpenMWP Dataset: \url{https://github.com/hot-zhy/mathEducatorsSolving/tree/main/dataset