Template-Driven LLM-Paraphrased Framework for Tabular Math Word Problem Generation

Xiaoqiang Kang[1,2,*], **Zimu Wang**[1,2,*], **Xiaobo Jin**[1], **Wei Wang**[1], **Kaizhu Huang**[3], **Qiufeng Wang**[1,†]

[1]School of Advanced Technology, Xi'an Jiaotong-Liverpool University
[2]University of Liverpool
[3]Duke Kunshan University
{Xiaoqiang.Kang23, Zimu.Wang19}@student.xjtlu.edu.cn, Qiufeng.Wang@xjtlu.edu.cn

Abstract

Solving tabular math word problems (TMWPs) has become a critical role in evaluating the mathematical reasoning ability of large language models (LLMs), where large-scale TMWP samples are commonly required for fine-tuning. Since the collection of high-quality TMWP datasets is costly and time-consuming, recent research has concentrated on automatic TMWP generation. However, current generated samples usually suffer from issues of either correctness or diversity. In this paper, we propose a **Te**mplate-driven **LL**M-paraphrased (TeLL) framework for generating high-quality TMWP samples with diverse backgrounds and accurate tables, questions, answers, and solutions. To this end, we first extract templates from existing real samples to generate initial problems, ensuring correctness. Then, we adopt an LLM to extend templates and paraphrase problems, obtaining diverse TMWP samples. Furthermore, we find the reasoning annotation is important for solving TMWPs. Therefore, we propose to enrich each solution with illustrative reasoning steps. Through the proposed framework, we construct a high-quality dataset TabMWP-TeLL by adhering to the question types in the TabMWP dataset, and we conduct extensive experiments on a variety of LLMs to demonstrate the effectiveness of TabMWP-TeLL in improving TMWP-solving performance.

Code — https://github.com/Jason8Kang/TELL
Extended version — https://arxiv.org/abs/2412.15594

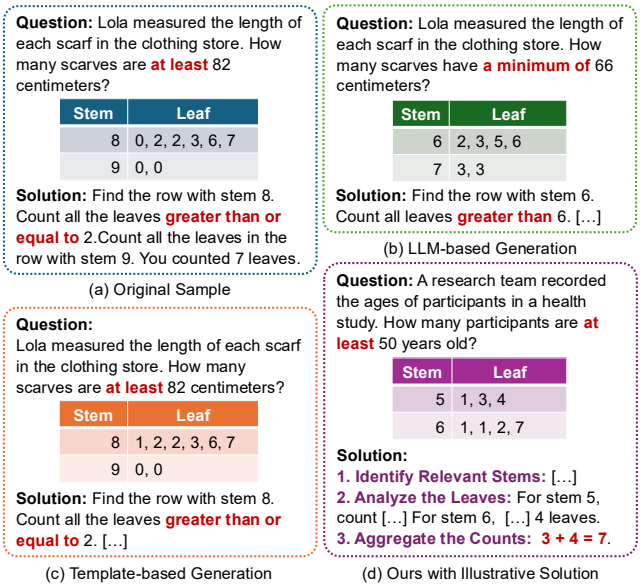

Figure 1: Illustration of an original TMWP sample and the generated samples with LLMs, templates, and ours with an illustrative solution. (Notes: "Stem" means the first digit, and "Leaf" means the last digit in stem-leaf plots).

Introduction

The rise of large language models (LLMs) has achieved unprecedented success in a variety of reasoning tasks (Peng et al. 2023; Li et al. 2024; Wang et al. 2024); however, solving math word problems (MWPs), particularly based on heterogeneous tabular and textual data is still challenging for LLMs (Lu et al. 2023b; Zheng et al. 2023). For various complex MWPs, training models usually require a large amount of data, but the collection and annotation of MWPs are usually costly and time-consuming, resulting in the scarcity of public tabular MWP datasets.

To mitigate the data issue, numerous studies have explored the ability to automatically generate MWP samples, mainly categorizing template-based (Williams 2011; Polozov et al. 2015), rewriting-based (Koncel-Kedziorski et al. 2016), neural network-based (Liyanage and Ranathunga 2020; Liu et al. 2021), and LLM-based (Luo et al. 2023; Tang et al. 2024) methods. Despite the progress, existing methods still face three major challenges for tabular data:

(1) **Lack of correctness**. Generation-based methods, like rewriting-based and LLM-based methods, misunderstand the meaning of the questions due to the hallucination problem (Zhang et al. 2023) and thus get wrong answers. As shown in Figure 1(b), the LLM-generated solution calculates leaves greater than 66 but ignores the case equal to 66, which is inconsistent with the requirements of the question. (2) **Lack of diversity in problems**. Because all problems are generated from abstract templates (Williams 2011), template-based methods have limited diversity. As shown in Figure 1(c), the generated problem simply replaces some

[*]These authors contributed equally.
[†]Corresponding author.

numbers that do not affect the answer while keeping the overall content unchanged. (3) **Lack of illustrative steps in solutions**. As shown in Figures 1(a) and 1(d), our data annotations have more clearly described solution steps than other data annotations. The model induces multi-step reasoning behaviors through clear intermediate reasoning steps such as Chain-of-Thought (CoT) (Wei et al. 2022).

To overcome the aforementioned challenges, we propose a **Te**mplate-driven **LL**M Paraphrased (TeLL) framework for generating Tabular MWPs (TMWPs) using both templates and LLMs. Different from the previous template-based generation, which rewrites questions with minor modifications based on a pre-defined template as shown in Figure 1(c), our templates are extracted from an existing TMWP dataset, each of which is abstract and summarizes mathematical logic. To generate flexible templates, we utilize an LLM to extend those extracted templates with a broader range of question types, maintaining mathematical logic to ensure correctness. Although these extended templates can generate various TMWPs, the background and linguistic description of those generated problems are monotonous. To pursue high-quality, realistic generated samples, we leverage the powerful language ability of LLMs to paraphrase problems with various contexts, obtaining diverse problems. Since our LLM-based paraphrasing does not change the mathematical logic, the correctness can be ensured. The overall framework is shown in Figure 2, and the details can be found in the section of Methodology. In summary, our generation method combines the advantages of both templates and LLM, ensuring the correctness and diversity of the generated samples.

Based on the framework, we construct a high-quality dataset named TabMWP-TeLL based on the question types in the TabMWP dataset (Lu et al. 2023b). We find that the step-by-step reasoning annotations are significant for using LLMs; therefore, we propose refining the original solutions with more illustrative steps, as shown in Figure 1(d). In our generated dataset, we utilize an LLM, Yi (Young et al. 2024), to paraphrase the template-based problems. In experiments, we fine-tune three LLMs, including Mistral (Jiang et al. 2023), Qwen 2 (Yang et al. 2024), and Llama 3 (Dubey et al. 2024). Experimental results show that TabMWP-TeLL is effective in improving TMWP solving, outperforming the baselines by a large margin, and is particularly effective in improving performance on challenging problems while maintaining the performance on simple ones.

Our contributions are as follows: (1) We propose TeLL, a template-driven, LLM-paraphrased framework, to generate high-quality TMWPs. To the best of our knowledge, we are the first to leverage templates and LLMs on TMWP generation, ensuring both correctness and diversity. (2) We propose to enrich TMWP solutions with more illustrative annotations, eliciting the multi-step reasoning ability of LLMs. (3) We construct a high-quality TMWP dataset, TabMWP-TeLL, which is an extension of the TabMWP dataset. The results of human verification illustrate certain correctness and diversity of the TMWP generation strategy. (4) Extensive experimental results demonstrate the effectiveness of TabMWP-TeLL in improving TMWP solving, outperforming the baselines by a significant margin.

Related Work

Math Word Problems. Recent research has primarily focused on addressing MWPs using generative models, such as sequence-based (Wang, Liu, and Shi 2017) and tree-based (Xie and Sun 2019; Zhang et al. 2020) models. With the rapid advancements of LLMs and the development of few-shot (Brown et al. 2020) and CoT (Wei et al. 2022) prompting, there has been a growing trend toward leveraging prompt engineering (Chen et al. 2023; Fu et al. 2023; Zhou et al. 2023a; Wang et al. 2023) and fine-tuning (Liu et al. 2023; Liang et al. 2024) strategies with notable performance. Furthermore, LLMs with math reasoning abilities have also been incorporated in the field of intelligent education (Macina et al. 2023; Wang and Demszky 2023). In this context, generating high-quality MWPs with both correctness and diversity is worthwhile.

Tabular Math Word Problems. Recent years have witnessed extensive research into solving TMWPs. TabMWP (Lu et al. 2023b) is the first TMWP reasoning dataset that contains $38,431$ grade-level problems with tabular context. PromptPG has been subsequently proposed, which utilizes policy gradient to select in-context examples for the test examples. Considering that LLMs often make errors in mathematical calculations, subsequent research has primarily focused on using external tools, such as calculators, to improve calculation accuracy. TaCo (Zheng et al. 2023) coordinates two Tabular LMs (TaLMs), which are responsible for CoT generation and answer inference, integrated with an external calculator. Chameleon (Lu et al. 2023a) composes various tools to accomplish complex reasoning tasks, such as LLMs, Python functions, and row and column look-up. CREATOR (Qian et al. 2023) and CRAFT (Yuan et al. 2024) create new tools for specific problems rather than calling the existing ones.

Math Word Problem Generation. Existing research in MWP generation can be broadly classified into four categories, including template-based, rewriting-based, neural network-based, and LLM-based methods. Template-based methods start by abstracting the existing MWPs into a template or a skeleton and then generating new problems from the abstract templates (Williams 2011; Polozov et al. 2015). Rewriting-based methods edit existing MWPs, altering the background of the problems while preserving their contents and logic (Koncel-Kedziorski et al. 2016; Moon-Rembert and Gilbert 2019). Neural network-based methods generate MWPs from topics and equations in an end-to-end manner (Liyanage and Ranathunga 2020; Liu et al. 2021; Zhou et al. 2023b). Inspired by the development of LLMs and their notable performance in various downstream applications, recent attempts have been focused on exploiting LLMs to generate MWPs, such as based on concepts (Tang et al. 2024) and key points (Huang et al. 2024). However, the aforementioned methods usually suffer from issues of either correctness or diversity. Different from the previous approaches, we propose a template-driven, LLM-paraphrased framework to generate high-quality TMWP samples with diverse descriptions and correct answers.

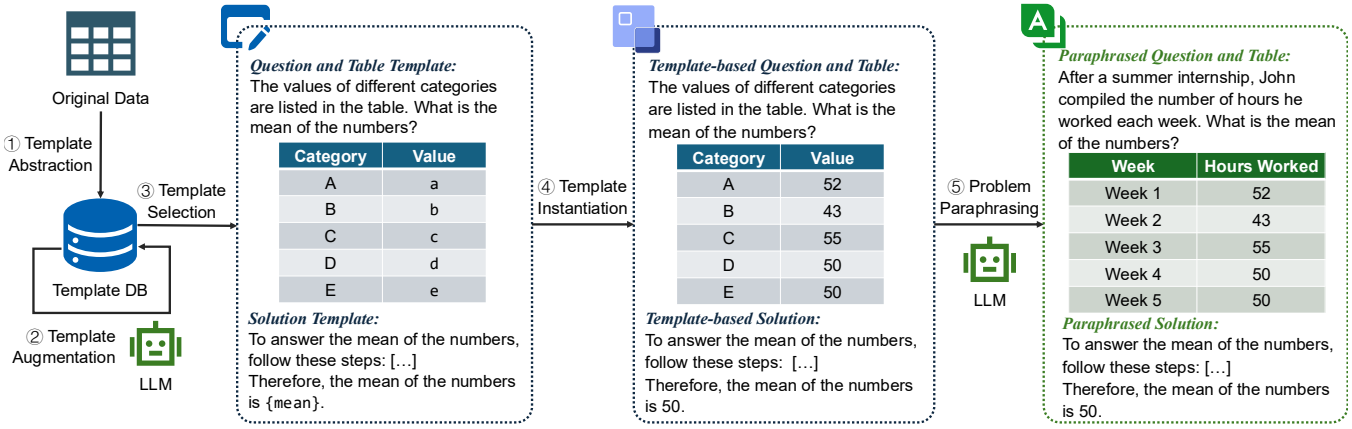

Figure 2: Overall framework of TeLL to generate TMWPs with correctness and diversity, consisting of five steps: (1) template abstraction, (2) template augmentation, (3) template selection, (4) template instantiation, and (5) problem paraphrasing.

Methodology

Problem Definition

We define Tabular Math Word Problems (TMWPs) as follows: given a table t containing multiple rows and columns and a question q about the table t, where the table could be a visual image, natural language text, or a structured database, our task is to generate a correct answer a that matches the ground truth of the question, derived by solution steps s. In our work, we focus on solving TMWPs using LLMs.

Note that the problems we are concerned with, regardless of the questions or the tables, are described in natural language texts; therefore, they usually follow certain rules. Further, for a given class of **problems**, we can abstract the template $P(\boldsymbol{x}) = (Q(\boldsymbol{x}), T(\boldsymbol{x}), A(\boldsymbol{x}), S(\boldsymbol{x}))$, containing a **question** template, a **table** template, an **answer** template, and a **solution** template, with respect to the problems, each of which contains placeholders that can be filled in the TMWP generation process. $\boldsymbol{x}$ contains all the numbers and their corresponding categories in the table. Intuitively, for a given template, we can directly replace the placeholders to obtain a new template-based question q^*, table t^*, answer a^*, and solution s^* via the following functions:

$$q^* = Q(\boldsymbol{x}),\ t^* = T(\boldsymbol{x}),\ a^* = A(\boldsymbol{x}),\ s^* = S(\boldsymbol{x}). \quad (1)$$

At the same time, to achieve diverse problem generation, we introduce an LLM to paraphrase the template-based problems to adapt to real-world scenarios. For q^*, t^*, a^*, and s^* mentioned above, we can obtain the corresponding paraphrased question q, table t, answer a, and solution s by the LLM as follows:

$$(q, t, a, s) = \text{LLM}(q^*, t^*, a^*, s^*). \quad (2)$$

TeLL for TMWP Generation

Overview. Figure 2 shows how the TeLL framework generates TMWPs while achieving the correctness and diversity of the generated problems. It consists of five stages: 1) **Template Abstraction**: abstracting the templates of mainstream TMWPs from existing datasets to build a template database; 2) **Template Augmentation**: using an LLM to expand the database to cover broader question types; 3) **Template Selection**: randomly selecting a template from the database for instantiation; 4) **Template Instantiation** (Equation (1)): instantiating the selected template into a problem by assigning random numbers and categories with predefined constraints; 5) **Problem Paraphrasing** (Equation (2)): with the support of an LLM, rewriting the problem into a problem with different contexts that conforms to human cognition. In the following, we will describe the details of each step.

Template Abstraction. We first abstract the problems and build a template database of mainstream TMWPs under the guidance of existing datasets. Each template contains a question template $Q(\boldsymbol{x})$, a table template $T(\boldsymbol{x})$, an answer template $A(\boldsymbol{x})$, and a solution template $S(\boldsymbol{x})$. Each of the abstracted templates contains some placeholders with predefined arithmetic operations, which are then filled with specific numbers and categories during the instantiation process, thereby updating the question, table, answer, and solution accordingly. Before abstraction, we first generate the illustrative step-by-step solution, denoted by $\hat{s}$, that corresponds to the original solution s_0 within the dataset:

$$\hat{s} = \text{LLM}(q, t, s_0). \quad (3)$$

Afterwards, we select a list of representative question types from the dataset and create a seed template database.

Template Augmentation. Due to the time-consuming nature of generating specific templates for each TMWP type, we propose template augmentation to generalize templates for a certain class of question types (such as questions about average calculation) to other categories with similar characteristics (such as median, mode, and range calculation questions). Specifically, we construct an LLM prompt, as shown in Figure 3, to drive the model to infer templates related to new categories (including questions, tables, answers, and solutions). This method leverages the inherent understanding and generalization capabilities of LLM to increase the diversity of generated TMWPs while improving the efficiency of template creation.

```
You are given a math word problem with tabular
contents, and your task is to develop a versatile
exercise template that can generate a wide array of
exercises with a table. Please consider the
following guidelines for this assignment:

(1) You can use pandas, numpy, random, etc., or
other packages if necessary.
(2) You should construct a general dataframe and
transform it into a human-readable table format by
Python.
(3) Generate the functions that are used to create
the question, answer, and step-by-step solution
based on the created table.

This is a demonstration for <Task 1>:
<Demonstration for Task 1>
Please generate the exercise template for <Task 2>.
```

Figure 3: Prompt for template augmentation.

Template Selection and Instantiation. After the template database is built, we randomly select a template from the database for instantiation. For each generation, we first generate random numbers and their corresponding categories to maintain diversity, where the values of the numbers meet specific constraints for different types of questions, such as integers in a certain interval, non-negative numbers, etc. Then, we use the generated numbers and categories to fill in the question and table, calculate the answer, and finally, fill them into the illustrative solution.

Problem Paraphrasing. Though ensuring correctness, the template-based TMWPs lack contextual backgrounds, which limits their applicability in practical scenarios. To address this issue, we use an LLM to paraphrase these problems into a more natural and contextual form, improving their generality without sacrificing the original data and solution logic. As shown in Figure 4, the prompts for the paraphrase process include instructions, three guidelines for each component of a problem (question, table, answer, and solution), two in-context examples, and a template-based problem. After the paraphrase process, we filter out problems whose answers obtained by the solution are inconsistent with those calculated by the template. We also remove questions in the test set where at least one sample has a BLEU score greater than δ to prevent potential data leakage issues.

In summary, our approach ensures both the correctness and diversity of problems compared with previous work. First, we classify and abstract the problems into a class of templates and randomly generate a series of questions, tables, answers, and solutions through a rigorous algorithmic procedure. By regarding the template-based problems as effective supervision, this approach avoids hallucination when generating problems with LLMs. At the same time, we introduce different contexts to the problems through the LLM with high language understanding and generation capabilities, enhance the complexity and authenticity of the problems, and ensure that the description of the problems and solutions conforms to the expression habits of English. In

```
You need to rewrite the given math word problem to
increase data diversity and semantic richness, whose
questions and solutions are generated according to a
uniform template. You should keep the original
problem, data, and solution logic unchanged.
Specific requirements are as follows:

(1) Question (`question`): You need to add a
background to the question, set the problem in a
specific scenario before introducing the question,
and rewrite the question without changing its
original meaning.

(2) Solution (`solution`): Since a background has
been introduced to the question, the solution
process should also be correspondingly rewritten,
but the idea and logic should remain the same. You
can appropriately modify any unreasonable parts in
the reasoning process based on the actual situation.

(3) Table Content (`table_for_pd`), Choices (`
choices`), and Answer (`answer`): These parts should
remain unchanged unless the keys of the tables and
the choices and answers for multiple-choice
questions.

Here are two examples:
<Two In-context Examples>

Please rewrite the following problem based on the
aforementioned requirements and examples:
<Template-based Problem>
```

Figure 4: Prompt for paraphrasing template-based problems to natural problems.

general, our method combines the accuracy of program algorithms in generating mathematical problems with the flexibility of LLMs so that high-quality and diverse TMWPs can be generated on a large scale. In addition, our method can be easily extended to new and unseen types of questions, showing its robustness and adaptability.

Experiments

Baselines

We compare our performance against the following baselines: (1) *Heuristic Baselines* include heuristic guess and human performance. (2) *Fine-tuned LMs*: We consider UnifiedQA, TAPEX, and TaCo under the fine-tuning setting to predict the final answers. **UnifiedQA** (Khashabi et al. 2020) is a T5-based model pre-trained on 8 question answering datasets of multiple formats. **TAPEX** (Liu et al. 2022) is a BART-based TaLM pre-trained on tabular data. **TaCo** (Zheng et al. 2023) coordinates two separate TaLMs for CoT generation and answer inference, respectively. We select the large version for the models. (3) *Few-shot Prompting* includes **GPT-3** (Brown et al. 2020) and **ChatGPT**. (4) *Few-shot CoT Prompting*: We consider standard **GPT-3**, **ChatGPT**, and **GPT-4** (OpenAI et al. 2024). We also select the following models: **PromptPG** (Lu et al. 2023b) selects in-context examples for test samples with a policy

	Train	Valid	Test	Total
#Question	23,059	7,686	7,686	38,431
#Free-text	17,135	5,710	5,694	28,719
#MCQ	5,744	1,976	1,992	9,712
#Table	22,620	7,546	7,549	37,644
#Solution	21,623	7,365	7,378	35,442

Table 1: Statistics of the TabMWP dataset.

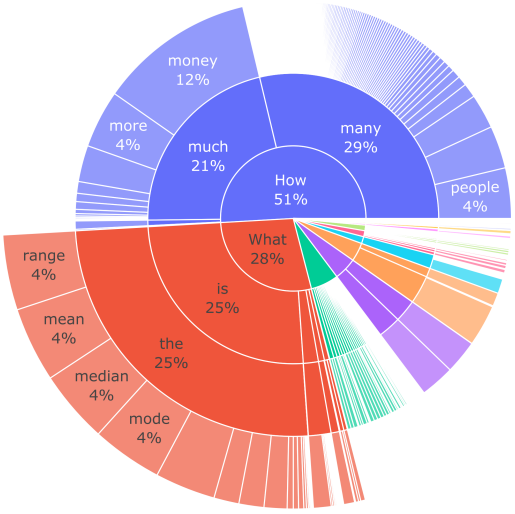

Figure 5: Question distribution of the TabMWP dataset.

gradient method. **PoT** (Program-of-Thoughts) (Chen et al. 2023) exploits Codex to generate the text and Python program for mathematical computations, where GPT-4 is used as the backbone model. **Chameleon** (Lu et al. 2023a) composes various tools, such as LLM, table verbalizer, and program generator, to accomplish the task.

Datasets and Evaluation

We conduct evaluations on TabMWP (Lu et al. 2023b), a recent large-scale dataset containing $38,431$ grade-level MWPs with tabular context, whose statistics are presented in Table 1. It consists of two types of questions: $28,719$ free-text questions (FREE) with integer (INT) and decimal (DEC) answers, and $9,712$ multiple-choice questions (MC) with extractive text answers (EXTR), Boolean text answers (BOOL) and other text answers (OTH). Apart from the golden answers, the dataset also contains problem solutions in a free-form format. We visualize the distributions of the questions in Figure 5, among which we select 25 main question types to create our TabMWP-TeLL dataset. For evaluation, we employ exact match accuracy to evaluate overall performance and the performance with respect to each question type, and we adopt the official evaluation script to evaluate the model performance on the test set.

Experimental Setup

We perform template augmentation and problem paraphrasing with Yi (`Yi-Large-Turbo`), an LLM that has close performance GPT-4 but with high cost-effectiveness. We access the model via its official API[1], and we set the threshold δ for the BLEU score as 0.95. In main experiments, we fine-tune three commonly used LLMs, including Mistral-7B (Jiang et al. 2023), Qwen 2-7B (Yang et al. 2024), and Llama 3-8B (Dubey et al. 2024) with the training set of TabMWP and TabMWP-TeLL, and the evaluations are conducted on the TabMWP test set. During the fine-tuning process, we set the number of epochs as 2, the batch size per device as 12, the gradient accumulation steps as 4, and the learning rate as $2e-4$. To achieve parameter-efficient fine-tuning, we adopt the QLoRA strategy (Dettmers et al. 2023) with XTuner[2]. All experiments are conducted on 8 NVIDIA GeForce RTX 3090 graphics cards.

Main Results

Table 2 illustrates the comparison of our models jointly trained with TabMWP and TabMWP-TeLL against baselines. By analyzing the experimental results, we have the following observations:

(1) The fine-tuned LLMs outperform almost all baselines, except Chameleon, built with the most advanced, closed-source GPT-4. The performance hierarchy among different approaches is clearly established, with few-shot prompting being the least effective, followed by few-shot CoT prompting, and culminating in the fine-tuned LLMs demonstrating the highest efficacy. This gradient of performance highlights the significant impact of large-scale, high-quality training data and reasoning steps on the model's capability in solving TMWPs.

(2) After incorporating TabMWP-TeLL, the performance of LLMs could be further enhanced. Specifically, Mistral, Qwen 2, and Llama 3 obtain an increase of 3.96%, 3.70%, and 3.78%, respectively. Among the experimented models, Llama 3 demonstrates substantial improvements and outperforms human performance by 7.85%, close to the performance of Chameleon by a significantly narrow margin. However, it only has 0.4% of parameters and without deliberated tools compared with Chameleon, underscoring the critical role of strategic augmentation of training data in elevating LLMs' reasoning performance.

(3) TabMWP-TeLL plays a critical role in improving performance on challenging problems while maintaining performance on simple ones. Initially, LLMs demonstrate strong capabilities in handling simple questions but struggle significantly with more challenging ones. Taking GPT-3 as an example, it achieves an accuracy of 68.62% on problems in grades 1-6 but only 55.31% on questions in problems 7-8. However, by jointly training with TabMWP-TeLL, LLMs, like Llama 3, mark substantial improvements in answering difficult questions (i.e., 97.42%) without compromising on simpler ones (i.e., 98.57%). This finding emphasizes the importance of high-quality data augmentation in enhancing the robustness and versatility of LLMs, thereby ensuring more consistent performance across a diverse range of question complexities.

[1]https://platform.01.ai/

[2]https://github.com/InternLM/xtuner

Model	#Para.	Question Type		Answer Type					Grade		Avg.
		FREE	MC	INT	DEC	EXTR	BOOL	OTH	1-6	7-8	
Heuristic Baselines											
Heuristic Guess	–	6.71	39.81	8.37	0.26	30.80	51.22	26.67	17.55	12.27	15.29
Human Perform.	–	84.61	93.32	84.95	83.29	97.18	88.69	96.20	94.27	81.28	90.22
Fine-tuned LMs											
UnifiedQA$_{\text{LARGE}}$	770M	48.67	82.18	55.97	20.26	94.63	68.89	79.05	65.92	45.92	57.35
TAPEX$_{\text{LARGE}}$	400M	51.00	80.02	59.92	16.31	95.34	64.00	73.33	67.11	47.07	58.52
TaCo$_{\text{LARGE}}$	800M	91.69	93.47	92.54	88.41	96.05	91.44	86.67	92.37	91.86	92.15
Few-shot Prompting											
GPT-3	175B	54.69	64.11	58.36	40.40	75.95	52.41	53.02	63.10	49.16	57.13
ChatGPT	UNK	65.84	64.61	66.55	63.09	74.67	54.67	55.24	69.75	59.88	65.52
Few-shot CoT Prompting											
Mistral	7B	47.32	48.34	47.62	46.17	48.30	48.87	44.09	47.66	47.49	47.59
Qwen 2	7B	67.84	68.44	68.06	66.94	68.12	69.49	62.56	67.94	68.06	67.99
Llama 3	8B	71.87	72.34	72.35	70.00	72.36	72.91	67.12	72.36	71.51	71.99
GPT-3	175B	60.76	69.09	60.04	63.58	76.49	69.19	67.30	68.62	55.31	62.92
ChatGPT	UNK	80.89	87.50	79.36	86.87	81.86	94.00	84.76	82.68	82.51	82.60
GPT-4	1.8T*	90.81	88.48	97.49	86.16	97.51	96.86	99.11	89.52	92.40	88.70
PromptPG	175B	66.17	74.11	64.12	74.16	76.19	72.81	65.81	71.20	64.27	68.23
PoT (GPT-4)	1.8T*	97.40	95.58	98.48	93.22	96.25	98.00	68.57	96.97	96.87	96.93
Chameleon	1.8T*	98.95	98.29	99.34	97.42	98.58	98.56	93.33	98.95	98.54	98.78
Fine-tuned LLMs (Trained with TabMWP)											
Mistral	7B	90.87	97.79	90.93	90.64	97.77	99.44	83.81	92.67	92.65	92.66
Qwen 2	7B	92.10	97.84	92.54	90.39	96.76	99.67	92.38	94.24	92.71	93.59
Llama 3	8B	94.13	94.73	94.81	91.50	98.78	90.11	96.19	96.61	91.19	94.29
Ours (Trained with TabMWP + TabMWP-TeLL)											
Mistral	7B	96.08	98.14	96.69	93.73	98.07	99.22	89.52	96.77	96.42	96.62
Qwen 2	7B	97.07	97.94	97.39	**95.79**	97.47	**99.44**	89.52	97.22	97.39	97.29
Llama 3	8B	**97.91**	**98.54**	**98.56**	95.36	**98.58**	99.33	**91.43**	**98.57**	**97.42**	**98.07**

Table 2: Experimental results of our models trained with TabMWP and TabMWP-TeLL against baselines. "Human Perform." denotes human performance, "UNK" denotes unknown, and * denotes the estimated size of GPT-4 (Bambhaniya et al. 2024).

Ablation Study

Effectiveness of TeLL for Problem Generation. We first analyze the effectiveness of TeLL, the template-driven, LLM-paraphrased TMWP generation strategy by comparing it with three additional methods: generating template-based problems, utilizing LLMs to generate new questions and tables, and utilizing LLMs to only rewrite the questions. To ensure a fair comparison, all three methods generate 23K problems. Experimental results are organized in Table 3. The template-based problems perform the worst due to their inability to adapt to real questions with diverse backgrounds. The generated questions and tables with LLMs show subpar performance due to inaccuracies that occur within. Besides, when using LLMs only to rewrite questions, despite some performance improvements, they are limited by the unchanged numbers within the questions and tables. In contrast, our proposed method, which generates questions based on templates and LLMs, effectively balances correctness and diversity in the augmented questions, leading to superior results on TMWP solving.

Effectiveness of Illustrative Solutions. We then analyze the efficacy of the illustrative solutions compared with the original free-form solutions for TMWP solving by fine-tuning the LLMs with the same questions but with different solution types, i.e., free-form solutions and the illustrative solutions, in which the illustrative solutions are constructed by LLMs given the free-form solutions. As shown in Table 4, the model trained with free-form solutions performs worse due to the lack of detail and structure. Our proposed illustrative solutions ultimately produce the best performance, indicating that the step-by-step solutions offer a more systematic and detailed approach, thus significantly improving the model performance.

Effectiveness across Multiple Question Types. We also examine the accuracy of various question types by training Llama 3 on three distinct datasets: TabMWP, TabMWP-TeLL, and a mix of both. As depicted in Table 5, the models trained solely on TabMWP demonstrate competence in solving straightforward problems, such as calculating mean and median; however, they struggle with more complex

Model	FREE	MC	Overall	Δ
Mistral	90.87	97.79	92.66	–
w/ Template	91.41	98.29	93.20	0.54
w/ Ques.+Table	91.27	97.64	92.92	0.26
w/ Ques. Only	92.48	**98.64**	94.08	1.42
Ours	**96.08**	98.14	**96.62**	**3.96**
Qwen 2	92.10	97.84	93.59	–
w/ Template	92.20	**98.19**	93.75	0.16
w/ Ques.+Table	92.48	97.79	93.86	0.27
w/ Ques. Only	93.47	97.84	94.60	1.01
Ours	**97.07**	97.94	**97.29**	**3.70**
Llama 3	94.13	94.73	94.29	–
w/ Template	93.08	97.99	94.35	0.06
w/ Ques.+Table	93.26	98.24	94.55	0.26
w/ Ques. Only	94.33	98.69	95.46	1.17
Ours	**97.91**	**98.54**	**98.07**	**3.78**

Table 3: Comparison of different TMWP generation methods. **Δ** calculates the overall improvement.

Solution	Mistral	Qwen 2	Llama 3
Free-form	92.66	93.59	94.29
Illustrative	**94.29**	**95.86**	**96.21**

Table 4: Performance of models trained on free-form and illustrative solutions.

Type	#Problem	TableMWP	TeLL	Combination
3	273	66.67	83.88	94.14
6	501	69.46	73.65	96.51
8	502	68.92	83.27	95.82
10	100	100.00	100.00	100.00
11	110	100.00	100.00	100.00
22	266	98.12	96.24	98.12
23	280	100.00	99.64	100.00
25	281	100.00	99.64	100.00
Avg.	2313	82.49	88.24	97.45

Table 5: Experimental results of Llama 3 trained with TabMWP, TabMWP-TeLL, and the combination of both.

tasks, particularly those involving stem-and-leaf plots. For instance, the performance on three types of stem-leaf plots is only 66.67%, 69.46%, and 68.92%, respectively. In contrast, the model trained on TabMWP-TeLL exhibits superior performance, showing an average improvement of 11.92% on the sampled stem-leaf plot problems, attributed to the inclusion of more detailed and illustrative solutions in our generated dataset, which enhanced the models' multi-step reasoning capabilities.

Effects of the Size of Augmented Data. We further investigate the effects of the size of augmented TabMWP-TeLL data with the TabMWP dataset on LLMs' TMWP-solving performance. As shown in Figure 6, incorporating more TabMWP-TeLL data consistently contributes to the performance across all models, even with smaller amounts

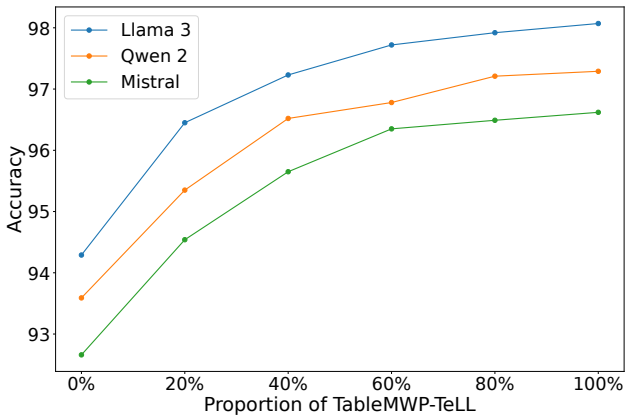

Figure 6: Effects of the proportion of TabMWP-TeLL on TMWP-solving performance.

of augmented data. Among these, a small amount of data (e.g., 20%) leads to a substantial improvement in model performance. As the data volume increases, the improvement rate slows, but the model continues to benefit from the generated data. This finding not only demonstrates the usefulness of additional augmented data in enhancing TMWP results but also highlights the critical importance of high-quality data in achieving superior model performance.

Human Verification. To guarantee the quality of our generated TMWPs, we sample 1,000 examples from TabMWP-TeLL for human verification. Generally, the generated data achieve high correctness, with a rate of 97.5%. We categorize the generation errors into three primary types during the problem paraphrasing step: incomplete paraphrased questions (LLMs may generate questions that are incomplete and unanswerable), incorrect paraphrased solutions (LLMs may generate solutions that are not logically correct with hallucination issues), and grammatical errors. We will address these errors in our future work.

Conclusion and Future Work

We introduce TeLL, a template-driven, LLM-paraphrased framework to generate high-quality TMWPs by leveraging both templates and LLMs. The framework consists of five steps, including abstraction, augmentation, selection, instantiation of templates, and the paraphrasing of problems. We construct a high-quality dataset called TabMWP-TeLL by adhering to the question types in the TabMWP dataset with illustrative step-by-step solutions, and we conduct experiments with three commonly used LLMs. Experimental results demonstrate the effectiveness of TabMWP-TeLL in improving TMWP solving by incorporating it with TabMWP, making LLMs' performance outperform the baselines, and it is particularly effective in improving performance on challenging problems while maintaining the performance on simple ones. In the future, we will generalize our method to more complex reasoning tasks, such as commonsense reasoning and multi-hop reasoning.

Acknowledgments

We thank all anonymous reviewers for their valuable comments. This work was partially supported by the following: National Natural Science Foundation of China under No. 92370119, 62436009, and 62376113, XJTLU Funding REF-22-01-002, RDF-TP-0019, and Suzhou Municipal Key Laboratory for Intelligent Virtual Engineering (SZS2022004).

References

Bambhaniya, A.; Raj, R.; Jeong, G.; Kundu, S.; Srinivasan, S.; Elavazhagan, M.; Kumar, M.; and Krishna, T. 2024. Demystifying Platform Requirements for Diverse LLM Inference Use Cases. arXiv:2406.01698.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, 1877–1901.

Chen, W.; Ma, X.; Wang, X.; and Cohen, W. W. 2023. Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks. *Transactions on Machine Learning Research*.

Dettmers, T.; Pagnoni, A.; Holtzman, A.; and Zettlemoyer, L. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. In Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems*, volume 36, 10088–10115.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The Llama 3 Herd of Models. arXiv:2407.21783.

Fu, Y.; Peng, H.; Sabharwal, A.; Clark, P.; and Khot, T. 2023. Complexity-Based Prompting for Multi-step Reasoning. In *The Eleventh International Conference on Learning Representations*.

Huang, Y.; Liu, X.; Gong, Y.; Gou, Z.; Shen, Y.; Duan, N.; and Chen, W. 2024. Key-Point-Driven Data Synthesis with its Enhancement on Mathematical Reasoning. arXiv:2403.02333.

Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; Casas, D. d. l.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; et al. 2023. Mistral 7B. arXiv:2310.06825.

Khashabi, D.; Min, S.; Khot, T.; Sabharwal, A.; Tafjord, O.; Clark, P.; and Hajishirzi, H. 2020. UNIFIEDQA: Crossing Format Boundaries with a Single QA System. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 1896–1907.

Koncel-Kedziorski, R.; Konstas, I.; Zettlemoyer, L.; and Hajishirzi, H. 2016. A Theme-Rewriting Approach for Generating Algebra Word Problems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 1617–1628.

Li, R.; Wang, Z.; Tran, S. Q.; Xia, L.; and Du, X. 2024. MEQA: A Benchmark for Multi-hop Event-centric Question Answering with Explanations. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Liang, Z.; Yu, D.; Pan, X.; Yao, W.; Zeng, Q.; Zhang, X.; and Yu, D. 2024. MinT: Boosting Generalization in Mathematical Reasoning via Multi-view Fine-tuning. In Calzolari, N.; Kan, M.-Y.; Hoste, V.; Lenci, A.; Sakti, S.; and Xue, N., eds., *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, 11307–11318. Torino, Italia: ELRA and ICCL.

Liu, Q.; Chen, B.; Guo, J.; Ziyadi, M.; Lin, Z.; Chen, W.; and Lou, J.-G. 2022. TAPEX: Table Pre-training via Learning a Neural SQL Executor. In *International Conference on Learning Representations*.

Liu, T.; Fang, Q.; Ding, W.; Li, H.; Wu, Z.; and Liu, Z. 2021. Mathematical Word Problem Generation from Commonsense Knowledge Graph and Equations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 4225–4240.

Liu, Y.; Singh, A.; Freeman, C. D.; Co-Reyes, J. D.; and Liu, P. J. 2023. Improving Large Language Model Fine-tuning for Solving Math Problems. arXiv:2310.10047.

Liyanage, V.; and Ranathunga, S. 2020. Multi-lingual Mathematical Word Problem Generation using Long Short Term Memory Networks with Enhanced Input Features. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, 4709–4716.

Lu, P.; Peng, B.; Cheng, H.; Galley, M.; Chang, K.-W.; Wu, Y. N.; Zhu, S.-C.; and Gao, J. 2023a. Chameleon: Plug-and-Play Compositional Reasoning with Large Language Models. In *Advances in Neural Information Processing Systems*, volume 36, 43447–43478.

Lu, P.; Qiu, L.; Chang, K.-W.; Wu, Y. N.; Zhu, S.-C.; Rajpurohit, T.; Clark, P.; and Kalyan, A. 2023b. Dynamic Prompt Learning via Policy Gradient for Semi-structured Mathematical Reasoning. In *The Eleventh International Conference on Learning Representations*.

Luo, H.; Sun, Q.; Xu, C.; Zhao, P.; Lou, J.; Tao, C.; Geng, X.; Lin, Q.; Chen, S.; and Zhang, D. 2023. WizardMath: Empowering Mathematical Reasoning for Large Language Models via Reinforced Evol-Instruct. arXiv:2308.09583.

Macina, J.; Daheim, N.; Chowdhury, S.; Sinha, T.; Kapur, M.; Gurevych, I.; and Sachan, M. 2023. MathDial: A Dialogue Tutoring Dataset with Rich Pedagogical Properties Grounded in Math Reasoning Problems. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Findings of the Association for Computational Linguistics: EMNLP 2023*, 5602–5621. Singapore: Association for Computational Linguistics.

Moon-Rembert, D. G.; and Gilbert, J. E. 2019. Illmatics: A Web-based Math Word Problem Generator for Students' Distal and Proximal Interests. In *Proceedings of E-Learn: World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education 2019*, 842–848.

OpenAI; Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; et al. 2024. GPT-4 Technical Report. arXiv:2303.08774.

Peng, H.; Wang, X.; Chen, J.; Li, W.; Qi, Y.; Wang, Z.; Wu, Z.; Zeng, K.; Xu, B.; Hou, L.; et al. 2023. When does In-context Learning Fall Short and Why? A Study on Specification-Heavy Tasks. arXiv:2311.08993.

Polozov, O.; O'Rourke, E.; Smith, A. M.; Zettlemoyer, L.; Gulwani, S.; and Popović, Z. 2015. Personalized Mathematical Word Problem Generation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Qian, C.; Han, C.; Fung, Y.; Qin, Y.; Liu, Z.; and Ji, H. 2023. CREATOR: Tool Creation for Disentangling Abstract and Concrete Reasoning of Large Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 6922–6939.

Tang, Z.; Zhang, X.; Wang, B.; and Wei, F. 2024. MathScale: Scaling Instruction Tuning for Mathematical Reasoning. In *Forty-first International Conference on Machine Learning*.

Wang, L.; Xu, W.; Lan, Y.; Hu, Z.; Lan, Y.; Lee, R. K.-W.; and Lim, E.-P. 2023. Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2609–2634. Toronto, Canada: Association for Computational Linguistics.

Wang, R.; and Demszky, D. 2023. Is ChatGPT a Good Teacher Coach? Measuring Zero-Shot Performance For Scoring and Providing Actionable Insights on Classroom Instruction. In Kochmar, E.; Burstein, J.; Horbach, A.; Laarmann-Quante, R.; Madnani, N.; Tack, A.; Yaneva, V.; Yuan, Z.; and Zesch, T., eds., *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, 626–667. Toronto, Canada: Association for Computational Linguistics.

Wang, Y.; Liu, X.; and Shi, S. 2017. Deep Neural Solver for Math Word Problems. In Palmer, M.; Hwa, R.; and Riedel, S., eds., *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 845–854. Copenhagen, Denmark: Association for Computational Linguistics.

Wang, Z.; Xia, L.; Wang, W.; and Du, X. 2024. Document-level Causal Relation Extraction with Knowledge-guided Binary Question Answering. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Findings of the Association for Computational Linguistics: EMNLP 2024*, 16944–16955. Miami, Florida, USA: Association for Computational Linguistics.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; ichter, b.; Xia, F.; Chi, E.; Le, Q. V.; and Zhou, D. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*, volume 35, 24824–24837.

Williams, S. 2011. Generating Mathematical Word Problems. In *2011 AAAI Fall symposium series*.

Xie, Z.; and Sun, S. 2019. A Goal-Driven Tree-Structured Neural Model for Math Word Problems. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, 5299–5305. International Joint Conferences on Artificial Intelligence Organization.

Yang, A.; Yang, B.; Hui, B.; Zheng, B.; Yu, B.; Zhou, C.; Li, C.; Li, C.; Liu, D.; Huang, F.; et al. 2024. Qwen2 Technical Report. arXiv:2407.10671.

Young, A.; Chen, B.; Li, C.; Huang, C.; Zhang, G.; Zhang, G.; Li, H.; Zhu, J.; Chen, J.; Chang, J.; et al. 2024. Yi: Open Foundation Models by 01.AI. arXiv:2403.04652.

Yuan, L.; Chen, Y.; Wang, X.; Fung, Y.; Peng, H.; and Ji, H. 2024. CRAFT: Customizing LLMs by Creating and Retrieving from Specialized Toolsets. In *The Twelfth International Conference on Learning Representations*.

Zhang, J.; Wang, L.; Lee, R. K.-W.; Bin, Y.; Wang, Y.; Shao, J.; and Lim, E.-P. 2020. Graph-to-Tree Learning for Solving Math Word Problems. In Jurafsky, D.; Chai, J.; Schluter, N.; and Tetreault, J., eds., *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 3928–3937. Online: Association for Computational Linguistics.

Zhang, Y.; Li, Y.; Cui, L.; Cai, D.; Liu, L.; Fu, T.; Huang, X.; Zhao, E.; Zhang, Y.; Chen, Y.; et al. 2023. Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models. arXiv:2309.01219.

Zheng, M.; Yang, H.; Jiang, W.; Lin, Z.; Lyu, Y.; She, Q.; and Wang, W. 2023. Chain-of-Thought Reasoning in Tabular Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 11006–11019.

Zhou, D.; Schärli, N.; Hou, L.; Wei, J.; Scales, N.; Wang, X.; Schuurmans, D.; Cui, C.; Bousquet, O.; Le, Q. V.; and Chi, E. H. 2023a. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. In *The Eleventh International Conference on Learning Representations*.

Zhou, Z.; Ning, M.; Wang, Q.; Yao, J.; Wang, W.; Huang, X.; and Huang, K. 2023b. Learning by Analogy: Diverse Questions Generation in Math Word Problem. In *Findings of the Association for Computational Linguistics: ACL 2023*, 11091–11104.