On LLM Prompting Techniques for Arabic Language Arithmetic Reasoning

Reem Khaled Alenezi, Ayed Atallah Salman

Department of Computer Engineering, Kuwait University
Kuwait City, Kuwait
r.alenezi@grad.ku.edu.kw, ayed.salman@ku.edu.kw

Abstract

Math word problems (MWPs) require complex reasoning to extract mathematical relationships from textual descriptions. While Large Language Models (LLMs) have shown remarkable performance on English mathematical reasoning tasks, their effectiveness on Arabic MWPs remains largely unexplored. This paper introduces three Arabic datasets (AGSM8K, Qudurat, and ArabicMWPs) and evaluates six LLMs using three prompting techniques: Manual Chain-of-Thought (CoT), Zero-shot CoT, and Self-consistency. Performance is assessed using accuracy and BERTScore metrics (precision, recall, F1-score). Our findings demonstrate that GPT-4o with Self-consistency achieves the highest accuracy of 97.65% on AGSM8K. It also obtains a precision of 71.94%, a recall of 71.31%, and an F1-score of 71.50%. The Arabic-specific LLM ALLaM achieves 84.41% accuracy on ArabicMWPs and 43.97% on AGSM8K. Fine-tuning experiments are further conducted on models using Arabic mathematical data. This work addresses the critical gap in Arabic mathematical reasoning resources and provides insights for developing Arabic-capable AI systems. Prompt-engineering methods combined with LLMs are regarded as a strong approach for advancing education and scientific research in solving Arabic mathematical problems.

Keywords: Large Language Models, Prompt Engineering, Mathematical Reasoning, Chain-of-Thought, Self-Consistency, Arabic Language

1. Introduction

Math Word Problems (MWPs) require integrating language comprehension with mathematical reasoning, making them critical for mathematical education. Large Language Models (LLMs) have demonstrated significant progress in various natural language processing (NLP) tasks. MWPs are questions described through a written context, and the solution is found by using mathematical equations. Solving MWPs requires accurate comprehension, analytical reasoning, and information extraction (Lu et al., 2022). MWPs are used in education, scientific research and everyday life.

Prompt engineering plays an important role in improving LLM performance by designing effective prompts that guide models toward optimal solutions, enabling them to handle complex tasks more effectively without additional training. (Amatriain, 2024). The Chain-of-Thought (CoT) technique, which mimics human logical reasoning by breaking down a problem into steps, has been employed to solve this type of task. CoT prompting can be implemented as a few-shot examples or zero-shot (Wei et al., 2022; Kojima et al., 2022). Additionally, for more complex mathematical tasks, the self-consistency prompting method generates multiple potential solutions and selects the most consistent response (Wang et al., 2022).

Arabic, spoken by over 300 million native speakers (World Population Review, 2024), presents unique linguistic challenges for MWP solving due to its rich morphology, dialectal variations, and frequent absence of diacritics. Despite progress in MWP solving using prompting techniques, most advancements focus on non-Arabic datasets (Ahn et al., 2024). Arabic MWP resources remain scarce, and advanced prompting methods have not been systematically explored for Arabic mathematical reasoning tasks.

This study addresses a critical gap in Arabic NLP by evaluating LLM capabilities in solving Arabic MWPs using three prompting techniques (Manual CoT, Zero-shot CoT, Self-Consistency). The research examines how different LLMs understand Arabic mathematical problems, extract numerical relationships, and generate coherent reasoning steps. The primary objective is to compare the performance of closed-source and open-source LLMs using different prompting methods to identify the most effective approach for Arabic mathematical reasoning.

Our contributions include: (1) Introduction of three new Arabic MWP with varying difficulty levels and question types, (2) Comprehensive evaluation of six LLMs using three prompting techniques on Arabic mathematical reasoning, (3) Analysis of fine-tuning effects on Arabic mathematical reasoning performance, and (4) Identification of optimal configurations for Arabic MWP solving.

To provide the necessary background, the next section reviews key prompting techniques before introducing the methodology and experiments.

Proceedings of The 7th Workshop on Open-Source Arabic Corpora and Processing Tools (OSACT7) with 5 Shared Tasks, pages 106–114

2. Prompt Engineering

Prompt engineering is the process of designing prompts that guide LLMs to correctly understand the task and produce accurate responses. After creating an effective prompt, it is passed to the LLM to obtain a better response (Liu et al., 2023). This technique allows researchers to use the prompt as an input with modifications to perform various tasks, which is easier than training the model from scratch for each task.

2.1 Standard Prompting

Studies have shown that LLM outputs are highly influenced by prompt design, highlighting the importance of crafting prompts that align with task requirements. A prompt may simply consist of instructions or a direct question. Zero-shot prompting relies solely on a natural language task description without examples, which is effective for simple tasks but often insufficient for problems requiring multi-step reasoning (Brown et al., 2020; Liu et al., 2023). A Few-shot prompting technique improves performance by incorporating a small number of illustrative examples that guide the model's reasoning and output structure. When this approach remains insufficient for complex mathematical problems, tasks can be decomposed into intermediate steps using Chain-of-Thought (CoT) prompting.

2.2 Chain-of-thought Prompting

Chain-of-Thought prompt method was proposed by Wei et al. (2022) to improve LLMs' ability to understand complex reasoning NLP task. This is achieved by providing the model with a prompt to generate intermediate reasoning steps to think about the question logically before giving the final answer. The main idea of a CoT prompt is to emulate human thinking when attempting to solve a complex problem. An effective approach is to break the question into smaller parts and solve each component individually. This is done by thinking step by step in a logical way before getting the outcome. In few-shot CoT, the model is provided with question-and-answer examples with intermediate logical steps. The CoT method depends on manually constructed examples, leading to its alternative name, Manual CoT (Wei et al., 2022).

Another advanced prompting method used for reasoning tasks is Zero-shot CoT (Kojima et al., 2022). Unlike manual CoT, this technique does not provide explicit reasoning examples. Instead, it relies on adding the phrase "Let's think step by step" to the prompt. This simple instruction has been shown to encourage the LLM to generate intermediate reasoning steps before producing the final answer, thereby improving its logical performance.

2.3 Self-Consistency Prompting

Self-consistency is an advanced prompting technique that is used to enhance the LLMs abilities when dealing with logical and reasoning tasks (Wang et al., 2022). LLMs generate a single answer to the question when using traditional prompting techniques like CoT. These techniques may cause logical errors, especially with complicated tasks. To avoid this issue, self-consistency, which contains two main ideas: sampling and majority voting, is applied to LLM. This method works by generating multiple reasoning paths with different answers for the same problem, then selecting the most frequent answer as the final output. The number of reasoning paths affects the model's performance. Figure 1 outlines self-consistency method.

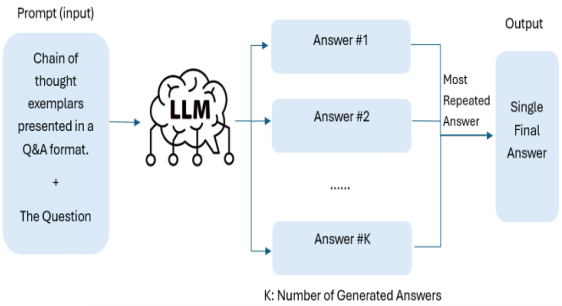

Figure 1: Self-consistency prompting method

The next section discusses prior studies related to Arabic language tasks and the use of prompting techniques for mathematical reasoning.

3. Related Work

3.1 Mathematical Reasoning with LLMs

There have been various research studies that have proposed prompting techniques for solving MWPs. These methods have produced different results in terms of accuracy and the responsiveness of the LLM when solving problems. The Chain-of-Thought (CoT) technique has shown impressive results when used with LLMs to solve MWPs that require logical reasoning to find the solution. Few-shot CoT provides the model with manually created demonstrations to generate intermediate steps to think logically about the question (Wei et al., 2022).

Kojima et al. introduced Zero-shot CoT, showing that simply adding "Let's think step by step" can elicit reasoning without examples. Wang et al. proposed Self-consistency, which samples multiple reasoning paths and selects the most frequent answer through majority voting. This approach achieved substantial improvements over standard CoT on various reasoning benchmarks.

Another prompt technique is presented called Plan-and-Solve prompting (Wang et al., 2023). In this method, LLM generates a plan to solve a problem then the model applies this plan to find the answer for the question correctly step by step. Recent research includes Active Prompting (Diao et al., 2023), which adaptively selects examples based on uncertainty, and Tree of Thoughts (Yao et al., 2023), which explores multiple reasoning branches systematically.

These previous studies highlight different prompting techniques that have been used to solve MWPs and have demonstrated significant improvements in the performance of LLMs.

Most previous studies on solving mathematical word problems using prompting techniques have been conducted in non-Arabic languages, largely due to the lack of widely available Arabic benchmark datasets. Consequently, research on Arabic mathematical word problems remains limited despite the importance of the field and the growing role of artificial intelligence in mathematical reasoning tasks. Recent work has also introduced Arabic-specific mathematical reasoning datasets, such as ArMATH (Alghamdi et al., 2022), which provides a benchmark for solving Arabic mathematical word problems using deep learning techniques.

One of the Arabic MWPs datasets is QIYAS which mainly contain multiple-choice math problems that are typically solved using zero, one, and few-shot prompting (Al-Khalifa & Al-Khalifa, 2024). In contrast, this work examines not only the final answer but also the generated reasoning steps. To the best of our knowledge, no study has systematically evaluated multiple prompting strategies across both closed-source and open-source LLMs on Arabic MWP datasets. This study addresses this gap by benchmarking performance across three Arabic datasets with varying levels of difficulty.

To place these prompting techniques within the Arabic context, the following subsection reviews relevant research in Arabic NLP.

3.2 Arabic Natural Language Processing

Artificial intelligence has made significant progress in NLP, with modern language models demonstrating multilingual comprehension and high accuracy across diverse tasks. Arabic requires special attention due to its complex morphology and distinctive linguistic features, prompting increased research focus on developing AI-based tools for Arabic text processing, including sentiment analysis and machine translation (Zouidine & Khalil, 2025), named entity recognition (Albahli, 2025), question answering (Basem et al., 2025), and grammatical error correction (Kwon et al., 2023), and text generation (Allam et al., 2025).

4. Methodology

4.1 Datasets

Three Arabic mathematical reasoning datasets are introduced in this work with different levels of difficulty:

AGSM8K: The Arabic Grade School Math 8K (AGSM8K) dataset is a novel dataset translated from the English GSM8K dataset (Cobbe et al., 2021). AGSM8K in Modern Standard Arabic consists of 7,473 training questions and 1,319 test questions, requiring between two and eight steps to solve.

Questions were translated using a ChatGPT model by providing it with a specific prompt designed to ensure that the final translation output followed the JSONL format. Manual review of all translations ensured mathematical consistency, eliminated translation errors, and validated that question meaning remained intact. After completing the translation and initial review, the translated questions were given to a language reviewer to ensure the correctness of the translation.

Qudurat: A novel dataset of 140 Academic Aptitude Test problems from Kuwait University (Kuwait University,2024). The number of steps in Qudurat questions ranges between 1 and 8 to find the final answer. Some resources of these questions include details of how to solve with logical steps. There were some questions with a final answer without any steps, so a specialist in mathematics was assigned to solve them in detail. To find solutions to these mathematic problems, basic arithmetic operations (addition, subtraction, multiplication, and division) are used. Additionally, one question involved comparison using “less than” and “greater than” relations.

ArabicMWPs: A novel dataset contains 725 MWPs in Arabic. It is translated from two English MWPs datasets: AddSub (Hosseini et al., 2014) and MultiArith (Roy & Roth, 2016). AddSub is a simple dataset that needs only addition and subtraction to solve questions. While MultiArith contains questions need more calculations. AddSub has 395 questions and MultiArith consists of 600 questions. ArabicMWPs include 725 questions, 265 from AddSub and 460 from MultiArith.

These datasets can support educational applications by enabling students to practice and develop new problem-solving strategies for Arabic MWPs. They can also assist educators in teaching mathematical reasoning.

4.2 Models and Prompting Techniques

Six LLMs were evaluated in the main experiments: four closed-source (GPT-4o, GPT-

3.5-turbo, GPT-4o-mini, O3-mini) and two open-source (Llama-3.1-8B-Instruct, Qwen2.5-Coder-7B-Instruct) models. In addition to Arabic-Specific Model (ALLaM) (Bari et al., 2024).

Three prompting techniques were employed:

1. **Manual CoT:** Eight manually few-shot examples from (Wei et al., 2022) for English, while translated these examples for Arabic.

2. **Zero-shot CoT:** Adding (Let's think step by step) for English and (" دعونا نفكر خطوة بخطوة") for Arabic.

3. **Self-consistency:** Generating multiple reasoning paths (K=10, 20, 40) for Arabic and selecting majority answer. For English (K=10 and 20).

4.3 Experimental Setup

All experiments were conducted using consistent settings across models. Temperature values of 0.0 and 0.5 were applied for closed-source models, while 0.1 and 0.5 were used for open-source models. Experiments were executed on Google Colaboratory using an NVIDIA A100 GPU with a Jupyter Notebook environment, and all code was implemented in Python.

For self-consistency prompting, multiple reasoning paths were sampled using fixed values of K (e.g., 10, 20, and 40), and the final answer was determined through majority voting. To ensure reproducibility, a single self-consistency prompting template was used across all experiments, with identical temperature settings, while only the underlying language model was varied.

4.4 Evaluation Approach

In the main experiments, AGSM8K is used for Arabic MWPs and GSM8K for English MWPs. Performance of LLMs is evaluated using accuracy, precision, recall, and F1-score metrics.

Accuracy measures the LLM's ability to answer correctly by calculating the number of correct questions divided by the total number of questions.

BERTScore is used to evaluate the quality of the generated output. It reports precision, recall, and F1-score (Zhang et al., 2019):

• Precision: the average similarity between each token in the candidate text and the most similar token in the reference.

• Recall: the average similarity between each token in the reference text and the most similar token in the candidate.

• F1-score: the harmonic mean of precision and recall.

Not all temperature values and experimental settings are presented; only the most informative configurations are presented to maintain clarity and highlight meaningful trends.

4.5 Fine-tuning

Fine-tuning experiments were conducted on the AGSM8K dataset using a 90% training and 10% validation split, with appropriate hyperparameters applied for each model type. Fine-tuning of the three closed-source models (GPT-3.5-turbo, GPT-4o, and GPT-4o-mini) were performed using OpenAI's fine-tuning API. In addition, two open-source models (Llama-3.1-8B and Qwen-3-8B) were fine-tuned using AdaLoRA (Zhang et al., 2023) with 4-bit NF4 quantization. Fine-tuning for the closed-source language models was relatively quick, with the slowest model being GPT-3.5-turbo, which completed training in approximately 3 hours. In contrast, fine-tuning the open-source language models requires significantly more time. Fine-tuning was not available for O3-mini.

4.5.1 Configuration for OpenAI Models

OpenAI models were fine-tuned using a batch size of 20 and trained for 16 epochs. A learning rate multiplier of 1e-3 was applied to enable gradual adaptation to Arabic mathematical patterns. A seed value of 42 was used to ensure reproducibility across all fine-tuning experiments.

4.5.2 Configuration for Open-source Models

Open-source models were fine-tuned with a batch size of 2 and trained for 50 epochs to ensure adequate adaptation to Arabic mathematical patterns. A learning rate of 3e-4 and weight decay of 0.001 were applied to balance effective learning with regularization against overfitting.

To support reproducibility for this work, all resources are publicly available at: https://github.com/ReemAlenezi/Arabic-MWPs-Reasoning

5. Main Results

5.1 Arabic AGSM8K Results

Table 1 shows the accuracy results across different prompting techniques on Arabic AGSM8K datasets. Self-consistency consistently outperformed other methods, with GPT-4o achieving 97.65% accuracy using Self-consistency with K=40 with precision 71.94%, recall 71.31%, and F1-score 71.50%. In addition, performance of ALLaM-7B is better than open-source models. Temperature is set to 0 for OpenAI models and 0.1 for open-source models.

Model	Manual CoT	Zero-shot	Self-consistency K=40
GPT-3.5-turbo	52.91	58.33	70.58
GPT-4o	94.47	95.07	**97.65**
GPT-4o-mini	79.53	86.50	91.36
O3-mini	93.94	94.85	96.13
Llama-3.1-8B	36.09	35.48	56.06
Qwen2.5-Coder	18.27	21.91	26.35
ALLaM-7B	42.38	38.52	--

Table 1: Accuracy (%) of LLMs across prompting methods on AGSM8K

5.2 Self-Consistency with Varying Sample Sizes

Table 2 examines the effect of increasing sample size (K) in Self-consistency prompting on AGSM8K. Increasing K generally improves accuracy, with larger gains observed for weaker models; however, performance tends to saturate or slightly degrade at higher sample sizes for stronger models. Temperature is set to 0 for OpenAI models and 0.1 for open-source models.

Model	Sc (K=10)	Sc (K=20)	Sc (K=40)
GPT-3.5-turbo	58.38	60.96	70.58
GPT-4o	95.22	96.29	97.65
GPT-4o-mini	88.86	90.60	91.36
O3-mini	95.15	95.76	96.13
Llama-3.1-8B	50.95	54.55	56.06
Qwen2.5-Coder	25.15	26.06	26.35

Table 2: Accuracy (%) of Self-consistency with different K values on AGSM8K

5.3 English GSM8K Results

Table 3 reports the accuracy of LLMs across different prompting methods on the English GSM8K dataset. Temperature is set to 0 for OpenAI models and 0.1 for open-source models.

Model	Manual CoT	Zero-shot	Self-consistency K=10
GPT-3.5-turbo	77.56	75.44	79.45
GPT-4o	93.78	93.40	96.33
GPT-4o-mini	87.95	82.41	91.13
O3-mini	98.48	97.27	98.48
Llama-3.1-8B	69.60	58.61	73.44
Qwen2.5-Coder	65.81	77.63	79.83

Table 3: Accuracy (%) of LLMs across prompting methods on GSM8K

Overall, self-consistency prompting with K=10 achieves the highest accuracy for most models. Closed-source models, particularly GPT-4o and O3-mini, demonstrate consistently strong performance across all prompting strategies. Notably, O3-mini achieves the highest accuracy at 98.48%. GPT-4o-mini also shows noticeable improvement under self-consistency compared to Zero-shot and Manual CoT prompting.

Open-source models achieve lower overall accuracy but benefit from self-consistency, which yields higher performance compared to other prompting methods. For cross-lingual comparison, self-consistency results at K=10 are reported consistently for both Arabic and English datasets.

5.4 Fine-tuning Results

Fine-tuning demonstrates a clear differential impact across model families. As shown in Table 4, open-source models benefit substantially from fine-tuning, with Qwen3-8B achieving the largest absolute improvement (+14.02%), followed by Llama 3.1-8B (+9.47%). These gains indicate that targeted training on Arabic mathematical data effectively enhances both language understanding and numerical reasoning in open-source models.

In contrast, closed-source models exhibit more modest changes after fine-tuning. GPT-3.5-turbo and GPT-4o-mini show moderate improvements, while GPT-4o maintains nearly identical performance before and after fine-tuning, suggesting that its pre-training already captures the necessary reasoning patterns for Arabic MWPs.

Model	Base Accuracy	Fine-tuned Accuracy	Change
GPT-3.5-turbo	58.33	63.43	+5.10
GPT-4o-mini	86.50	89.16	+2.66
GPT-4o	95.50	95.07	−0.43
Llama 3.1-8B	35.48	44.95	+9.47
Qwen3-8B	37.91	51.93	+14.02

Table 4: Accuracy (%) before and after fine-tuning on AGSM8K

5.5 Cross-Lingual Results Overview

Figure 2 provides a visual summary of model accuracy across Arabic (AGSM8K) and English (GSM8K) under various prompting techniques. The x-axis represents the evaluated models under different prompting techniques, while the y-axis shows accuracy scores. The prompting strategies include CoT, zero-shot CoT, and Self-consistency (K = 10) for both Arabic and English datasets.

LLMs demonstrate strong performance on mathematical word problems written in English (GSM8K). Open-source language models perform substantially better in English, achieving accuracies of approximately 52–83%, while their performance in Arabic ranges from about 17–54%.

Closed-source models demonstrate more balanced performance across both languages. Models such as GPT-4 and GPT-4o-mini achieve strong results on Arabic AGSM8K in certain settings, particularly when using Chain-of-Thought or Zero-shot CoT prompting. Additionally, O3-mini and GPT-3.5-turbo show consistently high and stable performance on English GSM8K across all prompting methods.

6. Discussion

6.1 Prompting Technique Effectiveness

The results demonstrated that the self-consistency method contributes to improving the performance of LLMs and are considered one of the most effective techniques for solving mathematical problems. This improvement is attributed to its underlying mechanism, which is based on generating multiple reasoning paths and selecting the most frequently occurring answer, thereby reducing the likelihood of errors caused by relying on a single reasoning path. Performance gains increase as the sample size grows, and this effect is particularly evident in language models with limited capabilities. This suggests that aggregating diverse reasoning paths enhances the stability of predictions and improves outcomes across different levels of model capacity.

6.2 BERTScore Interpretation

Although the models achieve high accuracy in solving mathematical word problems, the corresponding BERTScore values (around 71–72%) remain relatively moderate. This is because BERTScore measures similarity between generated and reference reasoning texts, whereas multiple valid reasoning paths can lead to the same correct answer.

As a result, correct solutions expressed with different wording or structure may receive lower BERTScore values. Therefore, accuracy is a more reliable metric for evaluating task success, while BERTScore serves as a complementary measure of textual similarity. Future work may explore alternative evaluation methods that better capture reasoning validity, such as step-level verification or human evaluation.

6.3 Cross-Lingual Performance Analysis

The results reveal a clear performance gap in solving Arabic mathematical word problems,

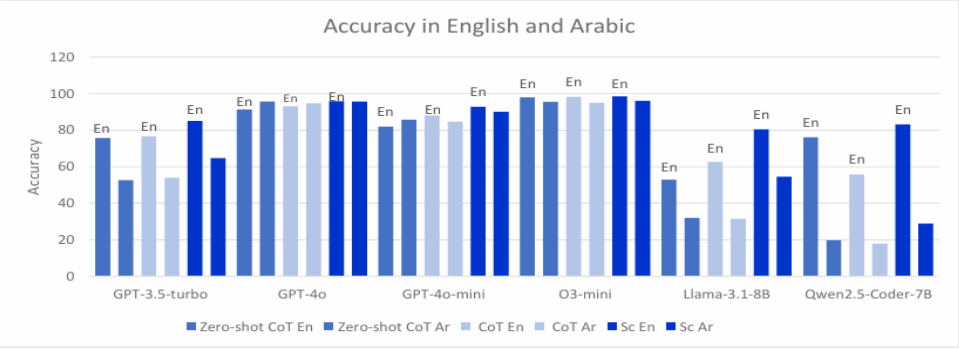

Figure 2: Accuracy of GSM8K and AGSM8K with LLMs

particularly for open-source language models. This gap can be partially attributed to the intrinsic limitations of these models, as well as their insufficient exposure to Arabic during training, where logical reasoning patterns are predominantly learned from English data. Open-source models also suffer from a scarcity of Arabic mathematical datasets, weaker representations of coherent step-by-step reasoning, and limited alignment during training. Furthermore, the morphological and syntactic complexity of the Arabic language introduces additional challenges for effective encoding and semantic understanding of mathematical text. As a result, models often rely on implicit internal translation before solving Arabic problems, which increases the likelihood of errors and leads to weaker performance compared to English. Open-source models generally underperform due to insufficient exposure to Arabic mathematical reasoning during training, highlighting the importance of fine-tuning Arabic-focused models.

In contrast, closed-source OpenAI models achieve stronger performance than open-source counterparts, largely due to their training on broader and higher-quality datasets that include mathematical content in both Arabic and English. This multilingual exposure enhances their ability to comprehend mathematical expressions and generate more accurate solutions across languages.

Nevertheless, English remains the primary language in which logical reasoning and problem-solving capabilities are learned, reflecting the dominance of high-quality mathematical resources in English relative to Arabic.

Performance differences among OpenAI models further reflect variations in model architecture and training objectives: GPT-4o demonstrates strong overall performance due to its exposure to high-quality data, while O3-mini is specifically optimized for logical reasoning tasks.

These factors explain the consistently superior performance of these models in solving mathematical problems in both Arabic and English.

6.4 Dataset Characteristics

The three datasets revealed different aspects of Arabic mathematical reasoning. These datasets differ in the difficulty levels, and this resulted in varying performance of the LLMs. AGSM8K and Qudurat datasets are considered medium level of difficulty, while ArabicMWPs is generally regarded as a simpler dataset. AGSM8K provided comprehensive coverage of grade-school mathematics, Qudurat reflected real academic assessment challenges, and ArabicMWPs offered basic arithmetic baselines. This diversity enables thorough evaluation of model capabilities.

Figure 3 illustrates accuracy across all three datasets with GPT-3.5, GPT-4o-mini, ALLaM, and O3-mini when using zero-shot CoT. ArabicMWPs achieved the highest accuracy.

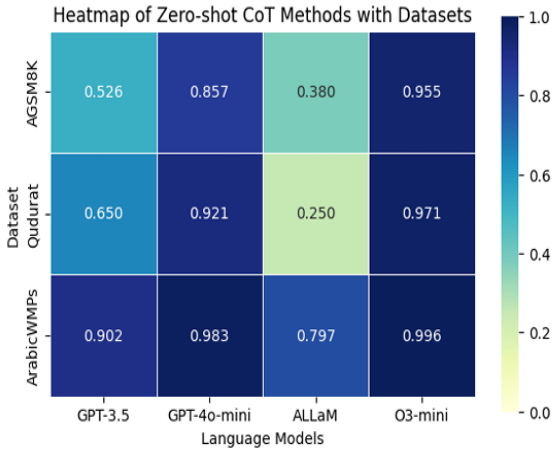

Figure 3: Accuracy of Datasets

7. Limitation and Future Work

Discussion of the limitations is important because it provides a broad understanding of the difficulties encountered in this research and helps in thinking about finding solutions to these problems in the future. There are several limitations that must be considered:

(1) Computational costs associated with extensive experimentation. A significant limitation of this research concerns the costs, especially associated with OpenAI LLMs. Using these models requires payment of costs which is calculated based on the number of tokens. The cost of each LLM is different from the other, in addition to the cost of training the models. In addition, the price of each prompting method differs from one to another due to the length of the prompt and API calls.

(2) Time when creating the new datasets. One of the difficulties in this research was the long time required to translate the data because of the large number of questions translated. In addition, the translation process needed to be accurate, as a

translation error could lead to misunderstanding of the question and resulting in an incorrect answer.

(3) Time when finding the answer. There are some cases that take a long time to find the result, such as self-consistency with high number of samples in open-source LLMs. Moreover, fine tuning LLMs also takes time for open-source models compared with closed-source models.

(4) Resource restrictions on Google Colab necessitated selecting smaller open-source models. Colab allows users to download the LLM in notebook with the proper code to use it. However, resources provided by Colab are not enough for some open-source LLM with big sizes of parameters, for example: 30B or 70B.

(5) Self-consistency results were not reported for the ALLaM model due to the high computational cost and latency associated with generating multiple reasoning paths. Running self-consistency with large sample sizes was not practical within the available resources. Therefore, results based on CoT and Zero-shot CoT prompting techniques are reported, as they still provide meaningful insights into the model's performance.

Future work should explore: (1) Development of larger Arabic mathematical datasets, (2) Investigation of Arabic-specific prompting strategies, (3) fine-tuning more LLMs on the AGSM8K dataset, and (4) Integration with Arabic mathematical education systems.

8. Conclusion

LLMs have successfully performed most NLP tasks, both simple and complex. Solving MWPs requires reasoning thinking to determine the correct answer. Prompt engineering constitutes a fundamental concept that has emerged in recent times with the aim of improving the understanding of LLMs for difficult tasks without the need to train the model. This work represents the first comprehensive evaluation of prompting techniques for Arabic mathematical word problem solving. The introduction of three Arabic datasets and the systematic evaluation of LLMs using three prompting techniques provide valuable insights for the Arabic NLP community.

Key findings include: (1) Self-consistency with GPT-4o achieves state-of-the-art performance (97.65%) on Arabic MWPs, (2) Fine-tuning significantly improves open-source model performance, (3) Substantial performance gaps exist between English and Arabic mathematical reasoning especially in open-source models, and (4) Arabic-specific datasets and evaluation frameworks are essential for progress in Arabic NLP. These results establish important benchmarks for Arabic mathematical reasoning and provide resources for future research in this critical domain.

9. Acknowledgments

This research was funded by the College of Graduate Studies at Kuwait University. The authors gratefully acknowledge their support, which made this research possible.

10. Bibliographical References

Ahn, J., Verma, R., Lou, R., Liu, D., Zhang, R., and Yin, W. (2024). Large language models for mathematical reasoning: Progresses and challenges. arXiv:2402.00157.

Albahli, S. (2025). An advanced natural language processing framework for Arabic named entity recognition: A novel approach to handling morphological richness and nested entities. Applied Sciences, 15(6), 3073.

Alghamdi, R., Liang, Z., and Zhang, X. (2022). ArMath: A dataset for solving Arabic math word problems. In Proceedings of the Thirteenth Language Resources and Evaluation Conference (LREC 2022), pages 351–362.

Al-Khalifa, S., and Al-Khalifa, H. (2024). The Qiyas benchmark: Measuring ChatGPT mathematical and language understanding in Arabic. arXiv:2407.00146.

Allam, A., Ahmed, S., Hamdi, A., and Mohammed, A. (2025). Arabic large language models for medical text generation. In Proceedings of the 4th International Conference on Computer Technologies (ICCTech) (pp. 1–6).

Amatriain, X. (2024). Prompt design and engineering: Introduction and advanced methods. arXiv:2401.14423.

Bari, M. S., Alnumay, Y., Alzahrani, N. A., Alotaibi, N. M., Alyahya, H. A., AlRashed, S. et al. (2024). Allam: Large language models for Arabic and English. arXiv:2407.15390.

Basem, M., Oshallah, I., Hamdi, A., and Mohamed, A. (2025). Few-shot prompting for extractive Quranic QA with instruction-tuned LLMs. In Proceedings of the Intelligent Methods, Systems, and Applications (IMSA) (pp. 24–29).

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P. et al. (2020). Language models are few-shot learners. Advances in Neural Information Processing Systems, 33, 1877–1901.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L. et al. (2021). Training verifiers to solve math word problems. arXiv:2110.14168.

Diao, S., Wang, P., Lin, Y., Pan, R., Liu, X., and Zhang, T. (2023). Active prompting with chain-of-

thought for large language models. arXiv:2302.12246.

Hosseini, M. J., Hajishirzi, H., Etzioni, O., and Kushman, N. (2014). Learning to solve arithmetic word problems with verb categorization. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 523–533).

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. (2022). Large language models are zero-shot reasoners. arXiv:2205.11916.

Kuwait University. Math Test Samples. Available at: https://cem.ku.edu.kw/mathsample

Kwon, S. Y., Bhatia, G., Nagoud, E. M. B., and Abdul-Mageed, M. (2023). ChatGPT for Arabic grammatical error correction. arXiv:2308.04492.

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., and Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. ACM Computing Surveys, 55(9), 1–35.

Lu, P., Qiu, L., Yu, W., Welleck, S., and Chang, K.-W. (2022). A survey of deep learning for mathematical reasoning. arXiv:2212.10535.

Roy, S., and Roth, D. (2016). Solving general arithmetic word problems. arXiv:1608.01413.

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. arXiv:2305.04091.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S. et al. (2022). Self-consistency improves chain of thought reasoning in language models. arXiv:2203.11171.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F. et al. Chain-of-thought prompting elicits reasoning in large language models. arXiv:2201.11903.

World Population Review (2024). Arabic speaking countries. Available at: https://worldpopulationreview.com/country-rankings/arabic-speaking-countries

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., and Narasimhan, K. (2023). Tree of thoughts: Deliberate problem solving with large language models. Advances in Neural Information Processing Systems, 36, 11809–11822.

Zhang, Q., Chen, M., Bukharin, A., Karampatziakis, N., He, P., Cheng, Y. et al. (2023). AdaLoRA: Adaptive budget allocation for parameter-efficient fine-tuning. arXiv:2303.10512.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2019). BERTScore: Evaluating text generation with BERT. arXiv:1904.09675.