From Small to Mighty: Advancing Math Reasoning Capabilities in Small Language Models

Sai Paresh Karyekar [*1] **Sharika Menon Rajeev** [*1] **Alyan Khan** [*1]

Abstract

Large Language Models (LLMs) excel in various tasks but struggle with mathematical reasoning. This challenge is often overcome by increasing model size. However, these larger models demand higher computational resources, making them unsuitable for low-resource environments like mobile devices or standard laptops. Small Language Models (SLMs), such as the T5-small transformer model (60M parameters), are efficient but they do perform well on reasoning-intensive tasks.

This project explores techniques to enhance mathematical reasoning in SLMs through methods like Low-Rank Adaptation (LoRA), LoRA with Chain-of-Thought (CoT) prompting, and full fine-tuning. Additionally, we propose a method to stack additional layers to a fine-tuned LoRA model. We compare our methods to the quantized and fine-tuned T5-base model (220M parameters).

***Keywords*—** Small Language Models, LoRA, Mathematical Reasoning, Chain-of-Thoughts

1. Introduction

Large Language Models (LLMs) have achieved remarkable results in summarization, translation, and text generation tasks. However, they often fall short in mathematical reasoning, where tasks demand pattern recognition and logical computation. Mathematical reasoning enables systems to tackle tasks that require logic, pattern recognition, and problem-solving. These capabilities are essential for numerous real-world applications, such as automated theorem proving, scientific simulations, financial modeling, and algorithmic trading. Beyond these specialized domains, mathematical reasoning also plays a pivotal role in enhancing a language model's general comprehension and ability to handle complex tasks like code generation, data analysis, and process optimization.

Large Language models (LLMs) like GPT-4 and PaLM have made significant strides in reasoning tasks, primarily due to their vast number of parameters and extensive training datasets. However, their resource requirements makes them unsuitable for deployment in resource-constrained environments such as mobile devices or edge computing. The computational overhead—comprising memory consumption, processing power, and energy demands causes issues with scalability and cost-effectiveness. As AI becomes increasingly integrated into everyday technologies, ensuring that models can function efficiently on modest hardware is crucial.

Small Language Models having significantly fewer parameters (0.1x) provide a lighter alternative, which can run efficiently on standard hardware. However, their reduced size and limited training data severely impair their ability to handle even general reasoning tasks. Addressing this performance gap is crucial to making the transformer model perform advanced tasks.

This project investigates methods to enhance reasoning performance in SLMs, focusing on the T5-small model (Raffel et al., 2020). We explore fine-tuning strategies, including Low-Rank Adaptation (LoRA) (Hu et al., 2021), LoRA integrated with Chain-of-Thought (CoT) prompting (Wei et al., 2023) (Brown et al., 2020) and full fine-tuning (Lv et al., 2024). We propose a novel approach called Stacked LoRA, which involves adding another layer of trained low-rank matrices to a previously fine-tuned LoRA model. For comparison, we fine-tune the T5-base model and use QLoRA (Dettmers et al., 2023) to quantize it, testing whether large-to-small compression outperforms direct fine-tuning of SLMs.

2. Methodology

Adapting large pre-trained models for specific tasks is resource-intensive, especially as model sizes grow. Parameter-Efficient Fine-Tuning (PEFT) addresses this by optimizing a small subset of parameters while keeping most weights frozen (Mangrulkar et al., 2022). By integrating lightweight, task-specific modules, PEFT enables efficient adaptation and reduces storage needs.

Building on PEFT, we explored fine-tuning strategies for the T5-small model using the GSM8k dataset, a benchmark collection of 8,000 grade-school math problems designed to test the reasoning abilities of language models. The dataset consists of problems requiring arithmetic, algebra, geometry, and logical reasoning, often combining multiple concepts within a single question. The GSM8k dataset is divided into two primary variants to assess and improve reasoning performance. The Main dataset provides direct problem-solution pairs, where the model is trained to produce the final answer based solely on the input question. In contrast, the Socratic dataset incorporates step-by-step reasoning, guiding the model through intermediate steps before arriving at the final answer.

2.1. LoRA

Low-Rank Adaptation (LoRA) is a PEFT method designed to optimize task-specific fine-tuning by using low-rank matrices. Rather than updating all model parameters, LoRA freezes the pre-trained weights and introduces small, trainable matrices A and B in each

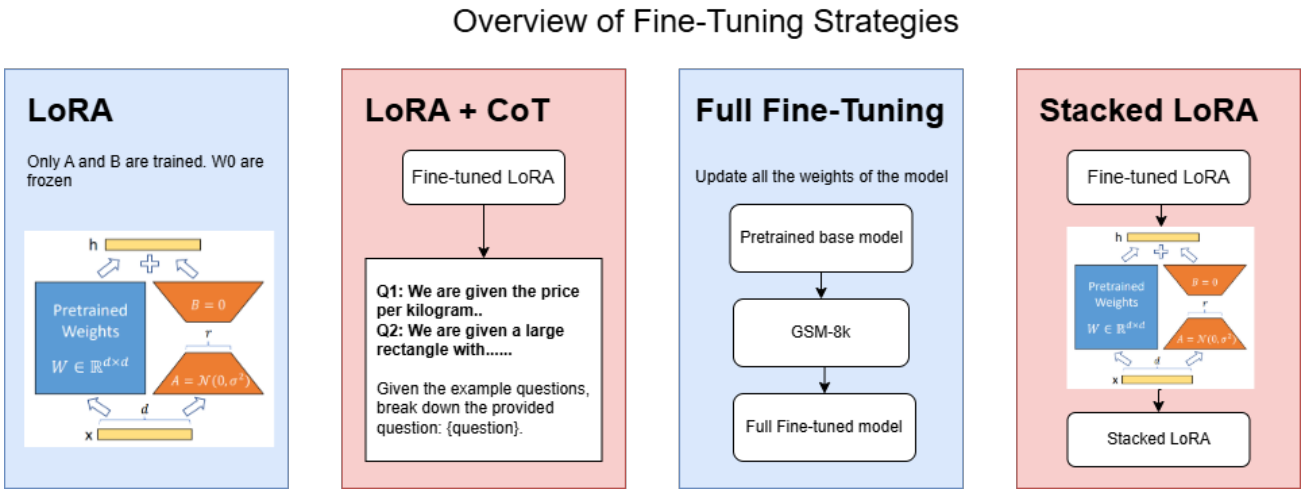

Figure 1. Overview of our methodologies

Transformer layer. The parameter decomposition is expressed as:

$$W = W_0 + \Delta W, \quad \text{where} \quad \Delta W = A \cdot B$$

Here, A and B are low-rank matrices, with ranks much smaller than the dimensions of W_0. This significantly reduces the number of parameters to be trained while preserving model performance.

LoRA's benefits include reduced GPU memory usage, enabling fine-tuning on standard hardware without additional inference latency (Hu et al., 2021). The low-rank matrices are merged into the frozen weights during deployment, ensuring no extra computational cost at inference. This modular approach allows task-specific adaptations without duplicating the base model, optimizing both storage and deployment.

2.2. LoRA with Chain-of-Thought

Chain of Thought (CoT) is a reasoning technique in language models that aims to enhance their ability to solve complex, multi-step problems by generating intermediate reasoning steps. The central idea of CoT is to guide the model to generate intermediate reasoning steps that lead to the final answer, mimicking the way humans solve problems by breaking them into smaller, logical sub-problems. This is particularly helpful in tasks such as mathematical reasoning, logical deductions, and multi-step problem solving. LoRA (Low-Rank Adaptation) combined with Chain of Thought (CoT) integrates efficient fine-tuning with structured reasoning to enhance the problem-solving capabilities of language models. When combined, LoRA fine-tunes the model using CoT prompts, teaching it to generate intermediate reasoning steps during training.

In our project, we evaluated three specific breakdown strategies for LoRA with Chain-of-Thought (CoT) prompting to enhance the mathematical reasoning capabilities of T5-small. We used the Question with Breakdown strategy, which involved presenting the model with a question that included a structured, step-by-step breakdown, guiding it through the reasoning process and helping it arrive at the correct answer. We also tested the Question without Breakdown strategy, where the model had to solve the problem without any intermediate steps, assessing its innate problem-solving ability. Lastly, we implemented the Python Code-Based Prompt(Lincoln Murr & Gao, 2023) strategy, which provided the model with prompts in the form of Python code to guide its reasoning process.

2.3. Full Fine-Tuning

A full fine-tuning approach was applied to enhance the mathematical reasoning capabilities of the T5-small transformer model. In this methodology, the entire set of trainable parameters of the pretrained T5-small model was updated during fine-tuning to adapt the model comprehensively to mathematical reasoning tasks. Unlike Low-Rank Adaptation (LoRA), which introduces efficient parameter updates by freezing the majority of the pretrained model's parameters and applying low-rank adjustments via lightweight adapters, full fine-tuning modifies all parameters of the model.

This approach offers greater flexibility in capturing complex patterns and interactions in the data but comes at the cost of increased computational overhead and memory usage.

2.4. QLoRA

QLoRA builds on LoRA by quantizing model weights to 4-bit precision, achieving performance similar to 16-bit fine-tuning while drastically reducing memory usage. It introduces three innovations: 4-bit NormalFloat (NF4) for optimal quantization of normally distributed weights, double quantization to minimize memory by quantizing constants, and paged optimizers that utilize NVIDIA unified memory to handle memory spikes during training.

For our experiment, we fine-tuned the T5-base model using QLoRA. This approach quantizes the model weights to 4-bit precision, significantly reducing memory usage. Compared to full precision fine-tuning, QLoRA reduces the model size by approximately 75%. Despite the reduced size, the model maintains performance comparable to 16-bit fine-tuning, with computational resources minimized by dequantizing weights during forward and backward passes and computing gradients only for the LoRA parameters.

2.5. Stacked LoRA

A progressive fine-tuning strategy utilizing Low-Rank Adaptation (LoRA) was employed to enhance the mathematical reasoning

capabilities of the T5-small transformer model. The process began with fine-tuning the pre-trained T5-small model on a mathematics-specific dataset GSM8K to adapt its parameters to mathematical reasoning tasks. This step aimed to specialize the model in understanding and solving problems within the domain of mathematics.

Following this, the model underwent an additional fine-tuning stage using a coding dataset Code-Feedback-Filtered-Instruction. This approach was motivated by previous research (Xinlu Zhang, 2024), which demonstrated that incorporating coding datasets can significantly improve the mathematical reasoning performance of language models by enhancing their ability to understand logical structures and dependencies. LoRA adapters were applied in both stages, with the second stage stacking an additional adapter on top of the first. The parameter decomposition is expressed as

$$W = W_0 + \sum_{i=1}^{2} \Delta W_i, \quad \text{where} \quad \Delta W_i = A_i \cdot B_i$$

This stacking of LoRA adapters enabled efficient parameter updates without modifying the core architecture of the pre-trained model, thus facilitating knowledge transfer between the mathematical and coding domains. The two-stage fine-tuning process was designed to systematically build on the initial specialization of the model in mathematics, further enhancing its reasoning capabilities through the integration of coding-related knowledge.

Below is a brief overview of the metrics that were used to determine the performance of the models.

- **Accuracy:** This metric evaluates the percentage of correct answers or predictions made by the model in classification or reasoning tasks.
- **ROUGE-1:** Rouge-1 measures the overlap of individual words between a generated text and a reference text. It evaluates how many words in the model's output appear in the human reference text, providing a way to assess content matching.
- **Perplexity:** Perplexity quantifies a model's confidence in predicting text sequences. It measures how "confused" the model is when making predictions - lower perplexity indicates higher certainty and better model performance. When a model has low perplexity, it demonstrates better understanding of language patterns and more accurate predictions of word sequences.
- **Semantic Similarity:** Semantic similarity evaluates how closely related two pieces of text are in terms of their meaning and context. It measures the conceptual and contextual relationships between texts, making it valuable for understanding how well a model captures the underlying meaning of language.

3. Results

We assess our fine-tuned model's performance based on accuracy, semantic similarity, perplexity, and ROUGE-1 scores. The goal of these experiments was to analyze the trade-offs between computational efficiency and task performance while maintaining strong language understanding capabilities.

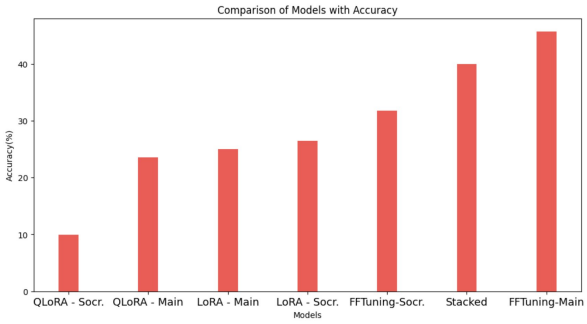

Figure 2. Comparison of accuracy of the models implemented

The LoRA implementation on the main dataset achieved an accuracy of 25%, while the Socratic version slightly outperformed it with an accuracy of 26%. Despite the relatively modest accuracy scores, both variants demonstrated strong semantic similarity, with scores of 79.70% for the main dataset and 80.42% for the Socratic dataset. This indicates that the model was able to retain its pre-trained language understanding capabilities while adapting effectively to the mathematical reasoning task.

In contrast, QLoRA, which quantizes the model weights to 4-bit precision, showed a slight dip in accuracy compared to standard LoRA. It achieved 23% accuracy on the main dataset and 10% on the Socratic version. However, QLoRA excelled in efficiency metrics. The model demonstrated a reduction in perplexity from 3.2633 for LoRA to 2.4158 for QLoRA, highlighting its enhanced computational efficiency. Additionally, QLoRA maintained high semantic similarity (79.99%), comparable to LoRA, and achieved slightly better ROUGE-1 scores (0.5156) compared to LoRA (0.5084). These results demonstrate that QLoRA can offer substantial memory savings while still performing competitively in terms of task adaptation.

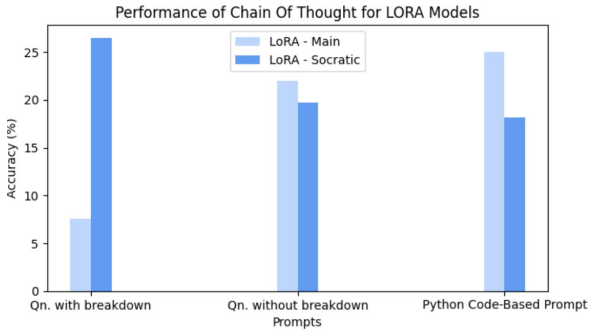

Figure 3. Accuracy for different CoT methods implemented on the fine-tuned LoRA model

To further explore the potential of LoRA, we evaluated its performance when combined with Chain-of-Thought (CoT) prompting strategies on the GSM8k main and Socratic datasets. Three CoT approaches were tested: questions with breakdown, questions without breakdown, and Python code-based prompts. For questions with breakdown, LoRA-Socratic significantly outperformed LoRA-Main, achieving an accuracy close to 25%, while LoRA-Main remained below 10%. For questions without breakdown, both configurations performed similarly, with accuracies just under 25%. The Python code-based prompting method achieved the

Table 1. Comparison of models and their Performance Metrics

Model	Training Params	Time (min)	ROUGE-1	Perplexity	Semantic Similarity (%)
LoRA - Main	294k	10	0.5084	3.2633	79.70
QLoRA - Main	884k	36	0.5156	2.4158	79.99
LoRA - Socratic	294k	10	0.4678	3.2706	80.42
QLoRA - Socratic	884k	37	0.5262	2.3754	75.96
Full Fine-Tuning - Main	60M	14	0.4136	2.2708	62.81
Full Fine-Tuning - Socratic	60M	14	0.4778	2.2353	62.50
Stacked LoRA	588k	15	0.3546	2.2587	61.38

highest accuracy on the main dataset, reaching 25% and matching the standard LoRA implementation. However, in the Socratic dataset, its performance was slightly lower, achieving accuracy of 18. 2%. Despite the reduced accuracy on the Socratic dataset, this method proved effective in guiding the model through structured, step-by-step problem solving processes.

We evaluated the Full Fine-Tuning approach using both the Main and Socratic strategies to assess their performance. For ROUGE-1, Full Fine-Tuning achieved scores of 0.4136 for Main and 0.4778 for Socratic, indicating a slight advantage for the Socratic approach in generating text that closely matches references. Perplexity scores were 2.2708 for Main and 2.2353 for Socratic, showing minimal differences in predictive accuracy between the two strategies. These results demonstrate that while Full Fine-Tuning enhanced model performance, the Socratic approach provided marginal benefits in ROUGE-1 and Perplexity.

Finally, for the Stacked LoRA model, the ROGUE-1 score of 0.35 was the lowest among all the models, indicating minimal utilization of the coding dataset it was retrained on. Despite this, the model achieved a perplexity score of 2.25 while maintaining a reasonable semantic similarity score of 61.38%, although this was also the lowest compared to the other model.

4. Evaluation

LoRA focuses on fine-tuning only a subset of the model's parameters by adding trainable low-rank matrices, which makes it computationally efficient while allowing for effective adaptation to specific tasks. This approach helps to preserve the existing capabilities of the pre-trained language model while updating it for new tasks. The high semantic similarity scores indicate that LoRA retained the language model's capacity for understanding and generating coherent text, even if it did not achieve high task-specific accuracy.

QLoRA quantizes the model's weights to 4-bit precision, which reduces memory requirements and increases computational efficiency. This leads to a lower perplexity score, showing that the predictive capability of the model improved in terms of processing speed and memory usage. However, quantizing the weights can degrade performance, particularly in complex reasoning tasks, as seen in the accuracy drop for the Socratic dataset. The high semantic similarity and competitive ROUGE-1 scores imply that QLoRA still effectively processes language data.

Chain-of-Thought (CoT) prompting provides a structured approach that helps the model break down problems step-by-step. This is especially beneficial for tasks requiring reasoning and logic, which aligns with the improvements observed in the "questions with breakdown" strategy. It helps guide the model through its thought process and enhances its ability to handle complex reasoning, explaining why this approach led to higher accuracy on the Socratic dataset.

Full Fine-Tuning involves updating all parameters in the model, which is computationally intensive. While it did show some benefits in terms of generating responses that are closer to human-like references (as indicated by ROUGE-1 scores), these improvements were not substantial enough to justify its use over more efficient methods like LoRA or QLoRA. The comparable perplexity scores suggest that while Full Fine-Tuning can refine language understanding, it does not significantly outpace more efficient approaches for this task.

Stacked LoRA's low ROUGE-1 score suggests minimal use of the coding dataset, likely due to the stacking approach diluting task-specific adaptation. Its perplexity score of 2.25 indicates improved predictive capability, but the semantic similarity score of 61.38% —the lowest among the models—implies a trade-off between adapting to new tasks and retaining the pre-trained model's general language understanding.

5. Conclusion

In this project, we explored methods to improve the mathematical reasoning abilities of Small Language Models (SLMs), focusing on the T5-small architecture. Our work emphasized fine-tuning techniques such as Low-Rank Adaptation (LoRA) and Chain-of-Thought (CoT) prompting. Among these, Stacked LoRA emerged as a promising approach that offers a good balance between computational efficiency and performance. It delivered results comparable to full fine-tuning but with significantly lower resource requirements. Additionally, CoT prompting, especially when incorporating Python code for step-by-step reasoning, enhanced both the interpretability and accuracy of the models. For larger models, such as T5-base, quantization to 8-bit precision preserved strong semantic similarity while improving computational efficiency.

6. Contributions

Each member contributed equally to different aspects of the project. Sai took charge of implementing the LoRA and QLoRA models, while Sharika focused on developing prompt-based Chain of Thought methods. Alyan concentrated on full fine-tuning and integrating stacked LoRA. Together, we reviewed and analyzed the results, ensuring we had a clear and thorough understanding of the outcomes.

7. Acknowledgments

We would like to express our gratitude to Prof. Amirali Aghazadeh for his timely guidance and support throughout the project. We also extend our thanks to Georgia Tech's PACE-ICE cluster system for providing the necessary computational resources.

References

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. Qlora: Efficient finetuning of quantized llms, 2023. URL https://arxiv.org/abs/2305.14314.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Lincoln Murr, M. G. and Gao, D. Testing llms on code generation with varying levels of prompt specificity, 2023. URL https://arxiv.org/pdf/2311.07599.

Lv, K., Yang, Y., Liu, T., Gao, Q., Guo, Q., and Qiu, X. Full parameter fine-tuning for large language models with limited resources, 2024. URL https://arxiv.org/abs/2306.09782.

Mangrulkar, S., Gugger, S., Debut, L., Belkada, Y., Paul, S., and Bossan, B. Peft: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft, 2022.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

Xinlu Zhang, Zhiyu Zoey Chen, X. Y. X. Y. L. C. Unveiling the impact of coding data instruction fine-tuning on large language models reasoning, 2024. URL https://arxiv.org/html/2405.20535v1.