Efficient Training of a Small Language Model with Parameter-Efficient Fine-Tuning and Agentic Reasoning for Mathematical Tasks

Maimoona Saifee
Sanchit Pendharkar

January 11, 2026

Abstract

Recent advances in Large Language Models (LLMs) have demonstrated impressive reasoning capabilities, particularly in mathematical and logical tasks. However, the computational and memory requirements of such models limit their accessibility and practical deployment. This project explores the feasibility of training a Small Language Model (SLM) from scratch and adapting it for mathematical reasoning using parameter-efficient fine-tuning techniques. A compact transformer-based language model is pretrained on OpenWebText to acquire general linguistic knowledge and subsequently fine-tuned on the GSM8K dataset using Low-Rank Adaptation (LoRA). In addition, an agentic reasoning layer is introduced at inference time to encourage iterative reasoning behavior without modifying model weights. The results demonstrate that even relatively small models can exhibit structured reasoning behavior when trained and utilized with appropriate architectural and algorithmic strategies.

1 Introduction

Transformer-based language models have become the standard approach for natural language understanding and generation. While large-scale models achieve strong performance across a wide range of tasks, their training and deployment costs pose significant challenges. This has motivated growing interest in Small Language Models (SLMs), which prioritize efficiency, interpretability, and accessibility.

Mathematical reasoning presents a particularly challenging domain for SLMs, as it requires structured multi-step reasoning rather than surface-level pattern matching. This project investigates whether a compact transformer model, trained from scratch and fine-tuned using parameter-efficient methods, can demonstrate basic mathematical reasoning capabilities. The GSM8K dataset, consisting of grade-school-level mathematical word problems with step-by-step solutions, serves as the primary benchmark for evaluating reasoning performance.

2 Limitations of Existing Approaches

Most state-of-the-art mathematical reasoning systems rely on very large pretrained models. While effective, such approaches suffer from several drawbacks:

- High computational and energy costs during pretraining.
- Large memory footprints that restrict deployment on resource-constrained hardware.
- Inefficient fine-tuning procedures that require updating all model parameters.
- Reduced transparency regarding the emergence of reasoning behavior.

These limitations motivate the exploration of smaller, more efficient models combined with targeted adaptation techniques.

3 Proposed Approach

The proposed system follows a three-stage pipeline:

1. **Base Pretraining on OpenWebText:** A decoder-only transformer model is trained from scratch on a subset of OpenWebText using autoregressive next-token prediction. This stage allows the model to learn general linguistic structure and coherence.
2. **LoRA Fine-Tuning on GSM8K:** The pretrained model is adapted to mathematical reasoning tasks using Low-Rank Adaptation. During this stage, the base model weights are frozen and only a small number of additional parameters are trained.
3. **Agentic Reasoning Layer:** At inference time, an external agentic control loop is introduced. This layer enables iterative reasoning through structured prompting without altering the trained model parameters.

This design separates language acquisition, task adaptation, and reasoning control, resulting in an efficient and modular system.

4 Implementation Details

4.1 Model Architecture

The base model is a compact decoder-only transformer with approximately 26 million parameters. The architecture includes:

- Token embeddings with weight tying between input embeddings and output projection.
- Multi-head self-attention layers with causal masking.
- Rotary Positional Embeddings (RoPE) for position encoding.

- Feed-forward networks with GELU activation.
- Residual connections and layer normalization for training stability.

This architecture balances expressiveness with computational efficiency.

Formal Architecture Specification (GPT-2 Style)

Let the input token sequence be $(x_1, x_2, \ldots, x_T)$. The model factorizes the joint probability as:

$$p(x_1, \ldots, x_T) = \prod_{t=1}^{T} p(x_t \mid x_{<t})$$

Each transformer layer computes:

$$h^{(l)} = \text{FFN}(\text{MHA}(h^{(l-1)})) + h^{(l-1)}$$

where MHA denotes masked multi-head self-attention and FFN denotes a position-wise feed-forward network with GELU activation.

Embeddings and Vocabulary (LLaMA Style)

Token embeddings are represented by:

$$E \in \mathbb{R}^{V \times d_{\text{model}}}$$

with vocabulary size:

$$V = 50{,}257$$

Rotary positional embeddings (RoPE) are applied to attention queries and keys:

$$\tilde{q}_t = \text{RoPE}(q_t, t), \qquad \tilde{k}_t = \text{RoPE}(k_t, t)$$

This eliminates the need for absolute positional embeddings and improves extrapolation to longer contexts.

4.2 Pretraining on OpenWebText

Pretraining is performed using next-token prediction with cross-entropy loss. A cosine learning rate schedule with warmup and gradient clipping is employed to ensure stable optimization. Due to computational constraints, pretraining is conducted on a limited subset of OpenWebText, with the goal of achieving basic linguistic coherence rather than full convergence.

Training Objective

$$\mathcal{L} = -\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t})$$

Learning Rate Schedule (GPT-3 Style)

Warmup phase:

$$\eta(t) = \eta_{\max} \cdot \frac{t}{T_w}$$

Cosine decay phase:

$$\eta(t) = \eta_{\min} + \frac{1}{2}(\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\pi \frac{t - T_w}{T_{\text{total}} - T_w}\right)\right)$$

Gradient clipping is applied:

$$\|\nabla_\theta\|_2 \leq \tau$$

Optimizer: AdamW.

4.3 LoRA Fine-Tuning on GSM8K

For task adaptation, Low-Rank Adaptation (LoRA) is applied to selected linear layers within the transformer. The base model parameters remain frozen, and only the low-rank adapter weights are updated. This reduces the number of trainable parameters to a small fraction of the total model size, enabling efficient fine-tuning on consumer-grade hardware.

Each GSM8K example is formatted as:

```
Question: <problem statement>
Answer: <step-by-step solution>
```

The entire sequence is used for autoregressive training, allowing the model to learn both reasoning structure and answer generation.

LoRA Parameterization

Each adapted weight matrix is expressed as:

$$W' = W + AB$$

where:

$$A \in \mathbb{R}^{d \times r}, \qquad B \in \mathbb{R}^{r \times k}, \qquad r \ll \min(d, k)$$

Only matrices A and B are updated during fine-tuning.

4.4 Agentic Reasoning Layer

An agentic reasoning layer is implemented externally at inference time. This layer wraps the trained language model in an iterative reasoning loop consisting of *thought*, *action*, and *observation* steps.

By feeding intermediate outputs back into the model as context, the system simulates multi-step reasoning behavior without modifying model weights or training procedures.

Model Configuration Summary

Parameter	Value
Architecture	GPT-2 style decoder-only Transformer
Embedding	Token + RoPE (LLaMA style)
Vocabulary size	50,257
Number of layers	12
Hidden size	384
Attention heads	6
Feed-forward size	1536
Total parameters	$\approx$ 26M
Optimizer	AdamW
Learning rate schedule	Warmup + cosine decay
Fine-tuning method	LoRA

Table 1: Summary of model hyperparameters.

5 Results and Observations

After LoRA fine-tuning, the model demonstrates improved performance on elementary arithmetic and simple multi-step word problems. The agentic reasoning layer further encourages structured outputs by decomposing problem-solving into iterative steps.

While the model does not achieve state-of-the-art accuracy, it produces partially coherent reasoning chains and correct answers for simpler problems. These results highlight the effectiveness of parameter-efficient fine-tuning and inference-time reasoning control in small-scale models.

6 Future Work

Potential directions for improvement include:

- Increasing model capacity while maintaining efficiency.
- Applying loss masking to focus training on reasoning tokens.
- Incorporating additional mathematical reasoning datasets.
- Merging LoRA adapters into the base model for deployment.
- Quantitative evaluation using exact-match accuracy on GSM8K.

These enhancements could significantly improve reasoning performance without requiring large-scale pretraining.

7 Conclusion

This project demonstrates that Small Language Models can acquire meaningful mathematical reasoning capabilities when trained using a structured and efficient pipeline. By combining pretraining on generic text, parameter-efficient fine-tuning with LoRA, and an external agentic reasoning layer, the system achieves a favorable balance between performance and computational cost. The findings suggest that compact transformer models represent a promising direction for efficient and interpretable reasoning systems.