Journal of Computer Science

Research Article

# Empowering Small Language Models With Chain of Thought and Parameter Efficient Fine Tuning for Efficient Deep Reasoning

**Aryan Raina, Shiwani Gupta, Jagruti Jadhav, Sampada Bhonde, Shilpa Mathur Anand Maha and Pranjali Sankhe**

*Department of Artificial Intelligence and Machine Learning, Thakur College of Engineering and Technology, Kandivali East, Mumbai, India*

*Article history*
Received: 25-04-2026
Revised: 18-06-2026
Accepted: 15-07-2026

**Corresponding Author:**
Shiwani Gupta
Department of Artificial Intelligence and Machine Learning, Thakur College of Engineering and Technology, Kandivali East, Mumbai, India
Email: shiwani.gupta@tcetmumbai.in

**Abstract:** Small Language Models (SLMs) run faster and fit on modest hardware, yet solving multi-step logic problems has traditionally been difficult for them. This work investigates a systematic, multi-model framework that combines Chain-of-Thought (CoT) prompting with Low-Rank Adaptation (LoRA) parameter-efficient fine-tuning, and measures the contribution of each component independently. We evaluate three SLMs (TinyLlama-1.1B, Phi-3-mini-4k-instruct, and Qwen2.5-1.5B) using 1,164 training samples drawn from GSM8K, Microsoft Orca-Math-Word-Problems-200k, and OpenAI HumanEval, and evaluate on 500 GSM8K test problems. Phi-3-mini-4k-instruct achieves the strongest overall gains, reaching 16.0% exact match and 61.7% step accuracy under +CoT+LoRA, a 95% improvement over its own baseline. The decline observed under CoT-only prompting and the subsequent recovery under CoT+LoRA were statistically significant (McNemar test, $p<0.05$). A notable finding is that CoT prompting applied without fine-tuning degrades Phi-3-mini's exact match score by approximately 44% due to an instruction-format conflict that LoRA fine-tuning resolves. Our ablation shows that neither technique alone produces peak performance; their combination does. The framework trains fewer than 0.05% of model parameters and runs end-to-end on a single consumer-grade NVIDIA RTX 4500 Ada GPU with 24 GB VRAM, making capable SLM-based reasoning deployable without datacenter-scale infrastructure.

**Keywords:** Small Language Models, Chain-of-Thought, LoRA, Parameter Efficient Fine Tuning, Mathematical Reasoning, Ablation Study

## Introduction

Recent advances have resulted in substantial increases in the scale and capability of language models. GPT-4, PaLM, and LLaMA have significantly expanded the capabilities of natural language processing, handling complex reasoning, multi-step problem solving, open-ended generation, and applied language tasks such as sentiment analysis (Liu, 2022), achieving performance levels that were previously unattainable in many NLP tasks (Lan et al., 2020). However, the computational and financial costs associated with these models remain substantial.

Deploying such models in resource-constrained environments, including mobile devices, edge platforms, and latency-sensitive applications, remains challenging (Plaat et al., 2026). Small Language Models (SLMs), typically ranging from 1B to 10B parameters, provide a practical alternative. They can be deployed on consumer-grade hardware, respond faster, and consume far less energy than their larger counterparts (Vernikos et al., 2024). However, SLMs have historically exhibited limitations in complex reasoning tasks, specifically the kind of chained, multi-step inference that capabilities have been observed more consistently in larger-scale models. Enabling a 1B-parameter model to reliably perform multi-step mathematical reasoning remains substantially more challenging than in larger foundation models. (Wei et al., 2022).

Chain-of-Thought (CoT) prompting offers a partial solution. The underlying principle is straightforward: rather than asking a model to jump straight to an answer, the model is prompted to reason through intermediate steps first. Prior studies have demonstrated the effectiveness of this approach, and it has been shown to unlock reasoning capabilities that are often not elicited through direct-answer prompting (Press et al., 2023). However, CoT prompting was originally investigated primarily in large-scale language models, and its behavior in smaller architectures is not always predictable, as demonstrated by the experimental results presented in this study.

Another important component is model adaptation through fine-tuning. Full fine-tuning of even a modestly-sized model is computationally prohibitive for most researchers and practitioners. LoRA (Low-Rank Adaptation) sidesteps this by inserting small trainable modules into frozen model weights, achieving meaningful adaptation while touching only a fraction of total parameters (Hu et al., 2022). It has become a standard tool in the efficient fine-tuning toolkit, although its interaction with CoT prompting across diverse SLM architectures remains insufficiently explored.

This study addresses this research gap. The central research question is as follows: if we take a small language model, apply CoT prompting, and fine-tune it with LoRA, to what extent does each component contribute, and does the answer depend on the model architecture? To investigate this question, a controlled ablation study was conducted across three models and three configurations, using mathematical reasoning benchmarks for evaluation. The goal is not to claim state-of-the-art accuracy numbers, but to understand what is actually driving improvement and under what conditions these techniques work together versus against each other.

### *Research Contributions*

- Novel Framework: A systematic, multi-model ablation that jointly evaluates CoT prompting and LoRA fine-tuning across diverse SLM architectures, surfacing architecture-specific interaction effects that isolated studies tend to miss
- Comprehensive Ablation: A full ablation study probes three configurations (Baseline, +CoT, and +CoT+LoRA) across three models and datasets, using five distinct evaluation metrics, identifying the relative contribution of each component to performance improvement
- Multi-Metric Evaluation: We go beyond loss and perplexity, incorporating exact match accuracy, step accuracy, and self-consistency scoring to get a more complete view of reasoning quality
- Empirical Validation: 1,164 training samples from GSM8K, Orca-Math, and HumanEval, with evaluation on 500 GSM8K test problems
- Practical Guidelines: Deployment recommendations for consumer-grade hardware, including configurations that work from as little as 2.35 GB VRAM

## *Literature Review*

### *Chain-of-Thought Reasoning in Language Models*

Chain-of-Thought (CoT) prompting improves reasoning performance by encouraging models to generate intermediate reasoning steps before producing a final answer. Early studies demonstrated that PaLM achieved significant improvements in mathematical and logical reasoning tasks when guided through Chain-of-Thought prompting, although these benefits were initially observed primarily in larger-scale models. Recent advances in training methodologies and architectural optimization have enabled smaller models to benefit from similar reasoning strategies. Approaches such as zero-shot CoT, self-consistency during output selection, and progressing from simple to complex subtasks widened its reach. These adaptations have expanded the applicability of CoT prompting across a broader range of reasoning tasks (Press et al., 2023). The paradigm has since been extended beyond text, with structured reasoning schemas such as PEAR prompting being adapted to multimodal sentiment analysis (Yang et al., 2024), and alternative generation mechanisms such as diffusion-based reasoning exploring non-autoregressive chains of thought (Ye et al., 2024). A complementary line of recent (2025) work studies test-time scaling, which allocates additional inference-time computation to strengthen reasoning; with the right strategy, this can allow small models to rival far larger ones on mathematical benchmarks (Liu et al., 2025), and the space has now been surveyed systematically (Zhang et al., 2025).

### *Small Language Models and Efficient Architectures*

The increasing adoption of task-specific AI systems has intensified research efforts aimed at balancing reasoning capability with computational efficiency. Instruction tuning using CoT-style datasets has contributed to measurable improvements in the mathematical reasoning capabilities of SLMs (Plaat et al., 2026). Studies such as SGFT suggest that compact models can provide meaningful step-by-step reasoning support despite relatively modest accuracy levels (Ranaldi and Freitas, 2024; Li et al., 2024). Earlier efforts along this line include distilling chains of thought from large teacher models into substantially smaller students (Magister et al., 2023) and specializing smaller models for multi-step reasoning via targeted fine-tuning (Fu et al., 2023). More recent work in 2025 has continued to advance compact reasoning: rStar-Math demonstrates that small models can attain strong mathematical reasoning through self-evolved deep thinking (Guan et al., 2025), reasoning distillation has been refined for sub-4B small language models on reasoning-intensive tasks (Samarinas and Zamani, 2025), and recent surveys catalogue techniques for building small yet capable reasoning models, including knowledge distillation and model compression (Feng et al., 2025).

### *Parameter-Efficient Fine-Tuning Methods*

Parameter-Efficient Fine-Tuning (PEFT) techniques update only a small subset of model parameters while

preserving the majority of the pretrained architecture, thereby enabling efficient adaptation on resource-constrained hardware. Rather than updating all model parameters, LoRA introduces low-rank trainable matrices within attention layers, achieving effective adaptation with minimal computational overhead. (Hu et al., 2022; Dettmers et al., 2023). Researchers are now testing adaptive rank choices and tailored setups per layer, aiming to refine efficiency even more. The effectiveness of PEFT has been demonstrated across a widening set of tasks, including personalized information retrieval (Braga, 2024), code generation with large language models (Weyssow et al., 2025), and multimodal sentiment analysis (Mu et al., 2024), suggesting the approach generalizes well beyond pure natural language understanding.

### *Mathematical Reasoning Benchmarks*

Meaningful evaluation of reasoning requires benchmarks that go beyond surface-level accuracy. GSM8K and SVAMP target grade-school and arithmetic reasoning respectively, while MATH pushes into competition-level problem solving (Patel et al., 2021; Hendrycks et al., 2021). Discrete reasoning over structured paragraphs has been evaluated through benchmarks such as DROP, which tests numerical and multi-step inference beyond simple pattern matching (Dua et al., 2019).

### *Research Gaps and Motivation*

Despite substantial progress in both CoT prompting and PEFT, their combined impact across diverse SLM architectures remains insufficiently explored. Existing studies frequently focus on a single model, technique, or dataset, limiting the ability to distinguish generalizable improvements from configuration-specific effects. Clear ablation studies attributing gains to individual components are also rare. This work addresses those gaps with a unified framework tested across diverse models, datasets, and metrics, with rigorous ablation throughout.

## Methods

### *Framework Overview*

The proposed framework combines Chain-of-Thought prompting with LoRA fine-tuning while keeping both components separable enough to measure their individual contributions. Unlike prior work that often focuses on only one technique, this design enables controlled attribution of performance improvements. The pipeline moves through four stages: data curation and preprocessing, CoT prompt construction, LoRA-based fine-tuning, and multi-metric evaluation covering exact match, step accuracy, and self-consistency.

### *Chain-of-Thought Prompting Strategy*

For +CoT configurations, problems are formatted to encourage explicit step by step reasoning (Fig. 1) using the prompt:

"Solve step by step:\n\n{question}\n\nSolution:".

The two prompting styles diverge fundamentally in what the model is asked to produce. Under standard prompting (left), the model is given a question and is expected to emit only a final answer, leaving any underlying computation implicit and unverifiable. Under Chain-of-Thought prompting (right), the same question is reframed with an explicit instruction to reason step by step, and the model is expected to externalize each intermediate operation before committing to a final answer. The practical consequence is twofold: the model allocates generation tokens to intermediate reasoning steps rather than directly producing a final answer, and the resulting trace is inspectable, which is particularly relevant when evaluating step accuracy as a metric distinct from exact-match accuracy. This contrast is the conceptual basis for the ablation that follows: the difference between Baseline and +CoT configurations operationalizes exactly the shift shown in Fig. 1.

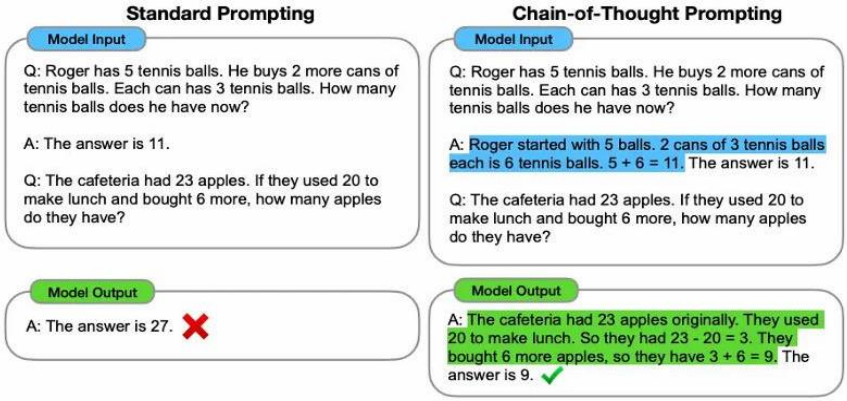


**Fig. 1:** Chain-of-Thought enhances reasoning. Comparison of standard prompting (left) vs. chain-of-thought prompting (right)

Baseline configurations skip this entirely and use a direct format instead: '{question}\nAnswer:'. Maintaining all other experimental conditions constant enables a controlled ablation analysis; thereby allowing observed performance differences to be attributed to prompt structure alone. All evaluations use zero-shot Chain-of-Thought prompting: the prompt supplies a single step-by-step instruction with no worked-in-context exemplars, so any reasoning the model produces is elicited by the instruction rather than copied from demonstrations.

### *Parameter Efficient Fine Tuning*

LoRA adapters are inserted into transformer attention layers while base weights stay frozen throughout (Hu et al., 2022). Key implementation details are summarized below:

- LoRA Configuration: r = 4, alpha = 16, dropout = 0.05, targeting attention projection layers specific to each model
- Architecture-specific targets: TinyLlama and Qwen2.5 use [q_proj, v_proj]; Phi-3-mini requires [qkv_proj] because of its fused attention design. Unlike TinyLlama and Qwen2.5, which expose the query, key, and value projections as separate linear modules (q_proj, k_proj, v_proj), Phi-3-mini packs all three into a single fused linear layer named qkv_proj. Because Phi-3-mini does not expose standalone q_proj or v_proj layers, LoRA adapters must target this fused projection instead; functionally, they adapt the same query and value subspaces, but through the combined weight matrix the architecture exposes
- Trainable parameters: 563K for TinyLlama (0.051%), 1,573K for Phi-3-mini (0.041%), 545K for Qwen2.5 (0.035%), a small fraction of total weights in each case
- Optimizer: AdamW, lr = 8e-5, gradient clipping at max_norm = 1.0, gradient accumulation steps = 8, giving an effective batch size of 8

### *Training Procedure*

Training runs for 2 epochs using AdamW with lr = 8e-5 and a gradient accumulation factor of 8, yielding an effective batch size of 8. Seed is fixed at 42 across all runs for reproducibility. Notably, only the +CoT+LoRA configuration involves parameter updates; Baseline and +CoT are evaluated as-is, thereby ensuring a fair and controlled comparison across configurations.

### *Ablation Design*

Three configurations were evaluated for each model, with each configuration isolating a distinct experimental factor:

- Baseline: Pre-trained weights, direct answer prompting, no fine-tuning
- +CoT: Same weights, CoT prompting only. Isolates the prompt engineering contribution
- +CoT+LoRA: CoT prompting plus LoRA fine-tuning. Evaluates the combined effect of prompting and parameter-efficient fine-tuning

The difference between Baseline and +CoT measures the effect of prompting alone, while the difference between +CoT and +CoT+LoRA measures the additional contribution of fine-tuning.

### *Multi-Model Benchmarking*

Three compact models are evaluated across diverse tasks and datasets. The primary metric is exact match accuracy, supported by step accuracy, self-consistency across 3 samples via majority voting, validation loss, and perplexity.

### *Implementation and Reproducibility*

- Hardware: NVIDIA RTX 4500 Ada Generation (24 GB VRAM), fully local, with no cloud dependency
- Software: Python 3.11, PyTorch 2.5.1+cu121, Hugging Face Transformers 5.3.0, PEFT 0.18.1, Accelerate 1.13.0
- Reproducibility: seed = 42 throughout, partial results auto-saved after each experiment, full hyperparameters documented

### *Experimental Setup*

### *Model Selection and Configuration*

All tests ran locally on an NVIDIA RTX 4500 Ada Generation GPU with 24 GB of VRAM; no remote servers were involved, which kept timing consistent and eliminated any variability from switching environments. Model selection was guided by the objective of representing distinct deployment scenarios rather than current popularity or benchmark rankings.

- TinyLlama-1.1B-Chat: At 1.1 billion parameters, TinyLlama represents a highly compact chat-oriented language model. Such compact architectures are particularly suitable for resource-constrained environments, including low-power edge devices and embedded systems. It can run even when available memory is tight (Lin et al., 2024)
- Phi-3-mini-4k-instruct: Despite its compact size, the model demonstrates strong performance characteristics. Its 3.8B parameter count and instruction-tuned design made it. The model was selected suitable for accuracy-oriented reasoning tasks. The model was selected primarily for its strong reasoning performance under resource-constrained conditions

- Qwen2.5-1.5B: At 1.5B parameters, it sits between the other two, not the leanest option, but far from demanding. The model provides a balanced trade-off between memory consumption and output quality, making it a reasonable stand-in for the kind of mid-range hardware that shows up most often outside of research settings

RedPajama was initially included but dropped early after showing unstable loading and weak performance on math tasks (Together Computer, 2023). Every model ran with a maximum sequence length of 128 tokens. To keep the comparison across models as controlled as possible, all three were evaluated on the identical 500-problem GSM8K test set under the same padding and truncation policy (128 tokens), the same prompt templates per configuration, and the same fixed random seed (42). The one factor that cannot be held identical is tokenization itself, since each model ships with its own tokenizer; we therefore treat tokenizer-induced differences as part of the result rather than as a confound, and discuss their effect explicitly in the Limitations section.

### *Dataset Preparation and Preprocessing*

Three publicly available reasoning benchmarks were selected to provide coverage across diverse problem domains:

- GSM8K (500 training samples): Grade-school math word problems, multi-step reasoning required
- Microsoft Orca-Math-Word-Problems-200k (500 training samples): A larger, more diverse mathematical problem set for training robustness
- OpenAI HumanEval (164 samples, full test set): Code generation and programming logic problems, introducing an additional form of structured reasoning (Nijkamp et al., 2023)

Total training set comes to 1,164 samples across three domains (GSM8K = 500, Orca-Math = 500, HumanEval = 164). Evaluation is on 500 GSM8K test samples. A full 1,319-sample evaluation is also possible by setting EVAL_SAMPLES = 1319 in the provided code. All inputs were standardized into consistent prompt-response formats, padded to 128 tokens with attention masks generated accordingly. Average prompt lengths ranged from 58–71 tokens across datasets (Fig. 2), comfortably within the sequence length limit. To preclude data leakage, training and evaluation draw from GSM8K's separate, non-overlapping partitions: the 500 training problems are sampled from the GSM8K train split, whereas evaluation uses 500 problems from the held-out test split, so no test problem appears in training. The remaining training data (Orca-Math and HumanEval) belongs to distinct datasets that are never used at evaluation, and the random seed is fixed at 42 throughout for reproducibility.

Two characteristics of the dataset composition are noteworthy. The left panel shows a deliberate weighting: GSM8K and Orca-Math each contribute 43.0% of training samples (500 each), while HumanEval contributes the remaining 14.1% (164 samples). This distribution was intentionally designed; it concentrates roughly 86% of the training signal on natural-language mathematical reasoning, where the GSM8K evaluation will ultimately be performed, while maintaining exposure to code-oriented structured reasoning through HumanEval.

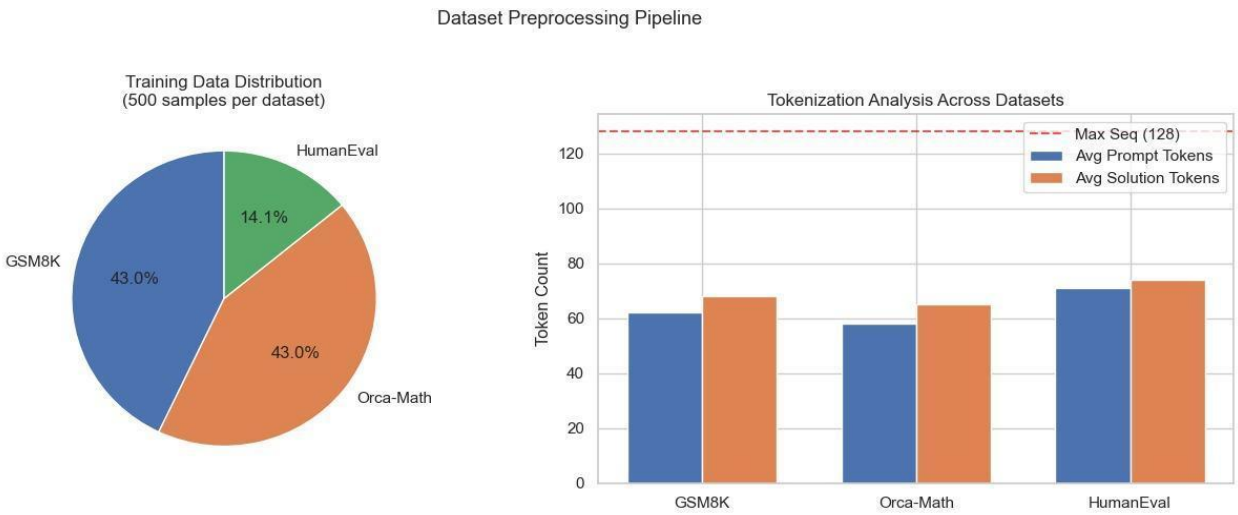


**Fig. 2:** Training data distribution across three datasets (left) and average token lengths per dataset (right). GSM8K = 43.0%, Orca-Math = 43.0%, HumanEval = 14.1%

The smaller HumanEval share reflects both its native dataset size and the secondary role of code reasoning relative to the mathematical target task. The token-length panel on the right confirms that all three sources fall within the 58–71 token range on average, well below the 128-token sequence cap. This means the cap is not truncating training inputs in any systematic way; the truncation effects discussed later in the Results section apply specifically to generation-time reasoning chains, not to training data formatting. The role of HumanEval in the training pipeline warrants clarification: it is used at training time only, as a small (14%) source of structured, step-style reasoning intended to diversify the training signal beyond natural-language arithmetic, and it is not an evaluation target in this study. Evaluation is restricted to GSM8K to ensure consistent comparison across all models and experimental configurations, which is what the ablation is designed to isolate. Consequently, this work does not separately measure code-generation capability and makes no claim of improvement on code synthesis; HumanEval results are therefore absent by design rather than omitted. Whether the structured-reasoning exposure from HumanEval transfers measurably to mathematical reasoning is not something the present single-benchmark evaluation can attribute in isolation, and we identify a dedicated cross-task evaluation (including a HumanEval pass@k measurement) as future work.

### *Training and Parameters*

Training was conducted for two epochs to ensure stable convergence. Hyperparameter settings are summarized in Table 1 and were selected to balance efficiency and stability on the RTX 4500 Ada GPU.

Only LoRA adapters contained adjustable parameters, limited to between 0.035% and 0.051% of the full model weight count (Ding et al., 2023). This approach preserves pretrained model knowledge while substantially reducing the computational requirements of fine-tuning.

### *Evaluation Criteria*

Five evaluation metrics were employed to assess model performance.

Exact Match on GSM8K defines the main score, counting how often model responses match the expected result exactly. Final predictions were determined through majority voting across three independent generations, taking the most frequent outcome across tries. This procedure provides a measure of both predictive accuracy and output consistency. This approach reduces noise from random output variation.

Step Accuracy quantifies the proportion of correctly generated intermediate numerical reasoning steps. A step is considered correct when the generated numerical value matches the corresponding reference value. Precision comes from hitting each real figure just once.

Validation loss was measured using cross-entropy loss and was reported only for the +CoT+LoRA configuration, but solely where CoT combines with LoRA. Reporting skips all other setups, focusing narrowly on that pairing.

Perplexity was computed from validation loss to provide an additional measure of model confidence and predictive quality by transforming validation loss through exponentiation.

### *Infrastructure*

A single NVIDIA RTX 4500 Ada GPU powers the system, equipped with 24 GB of dedicated video memory. Total available GPU memory reaches 60.2 GB when including shared resources. The machine runs Windows 11 without any cloud dependencies. Everything operates directly on local hardware.

Python version 3.11.9 forms the base environment. PyTorch 2.5.1 with CUDA 12.1 support enables GPU-accelerated operations. The Hugging Face Transformers library, at release 5.3.0, provides model architectures and tokenizers. Parameter-efficient methods come into play through PEFT 0.18.1. For distributed training workflows, Accelerate 1.13.0 handles device placement logistics. Loading and preprocessing pipelines are managed using Datasets 4.6.1. Analysis is performed with Pandas 3.0.1. For visualization, Matplotlib 3.10.8 is used alongside Seaborn 0.13.2.

All code, hyperparameter settings, dataset iterations, and random seeds were documented consistently, with intermediate results automatically saved for reproducibility.

**Table 1:** Training hyperparameters and their rationale

| Parameter | Value | Justification |
|---|---|---|
| LoRA rank (r) | 4 | Balances adaptation capacity and efficiency |
| LoRA alpha | 16 | Standard scaling factor (alpha / r = 4) |
| LoRA dropout | 0.05 | Regularization to prevent overfitting |
| Learning rate | 8e-5 | Empirically tuned for 2-epoch convergence |
| Batch size | 1 (effective: 8) | Gradient accumulation steps = 8 |
| Epochs | 2 | Stable convergence observed (see Table 4) |
| Max sequence length | 128 | Consistent with dataset token distribution |
| Optimizer | AdamW | Gradient clipping max_norm = 1.0 |
| Precision | float16 | 50% VRAM reduction vs float32 |
| Random seed | 42 | Full reproducibility across runs |

## Results and Analysis

### *Full Ablation Results*

Table 2 shows the complete results across all three models and configurations. Several observations warrant attention before discussing the quantitative results: Loss and Perplexity are only reported for +CoT+LoRA since the other configurations do not involve weight updates, and Exact Match uses majority voting across 3 self-consistency samples throughout.

### *Key Findings*

Phi-3-mini-4k-instruct achieved the strongest overall performance, reaching 16.0% exact match and 61.7% step accuracy under +CoT+LoRA, along with the lowest validation loss and perplexity among the three models (Fig. 3). Its instruction-tuned pretraining appears to provide an advantage on structured multi-step reasoning tasks once the format conflict is mitigated. Statistical significance was assessed using McNemar's test on paired GSM8K predictions. Results with $p < 0.05$ were considered statistically significant.

A notable finding is the observed format conflict. Applying CoT prompting to Phi-3-mini without any fine-tuning did not improve performance; instead, performance declined, dropping exact match from 8.2% down to 4.6% (Fig. 5). The model's existing instruction-following format and the CoT prompt structure appear to clash, resulting in competing output-generation behaviours. LoRA fine-tuning appears to mitigate this issue by adapting the model's weights to the CoT output format. Performance not only recovers to baseline levels but substantially exceeds them.

Qwen2.5-1.5B exhibits a different performance pattern. CoT prompting alone pushed exact match from 11.4 to 16.6%, the highest absolute score of any single configuration across all three models. Adding LoRA on top of that actually caused a regression in exact match (down to 12.8%), which we attribute to the 128-token sequence length constraint cutting off full reasoning chains before they complete. Step accuracy still improved significantly though, climbing from 37.7 to 55.8%, suggesting the model is learning to reason more carefully even when it cannot always finish. Future experiments using longer sequence lengths may provide additional insight into this behaviour.

As the smallest model evaluated, TinyLlama exhibits performance characteristics consistent with its limited parameter capacity. Exact match accuracy remained low throughout (1.0 to 2.2%), which is expected for a 1.1B parameter model on multi-step mathematical reasoning tasks. Notably, step accuracy improves consistently across all evaluated configurations. (Fig. 4): 22.8% → 32.7 → 46.4%, confirming that even the smallest model acquires measurable reasoning structure from CoT training, even if final answer accuracy does not reflect it.

The performance differences among the three models are evident in both panels. Phi-3-mini sits clearly below the other two, with a validation loss of 0.4474 and a perplexity of 1.56, values that indicate the model exhibits stronger convergence characteristics and lower predictive uncertainty. Qwen2.5 follows at 0.5893 loss and 1.80 perplexity, which is still a healthy convergence profile. TinyLlama lands noticeably higher at 1.2505 loss and 3.49 perplexity, roughly twice the perplexity of Phi-3-mini.

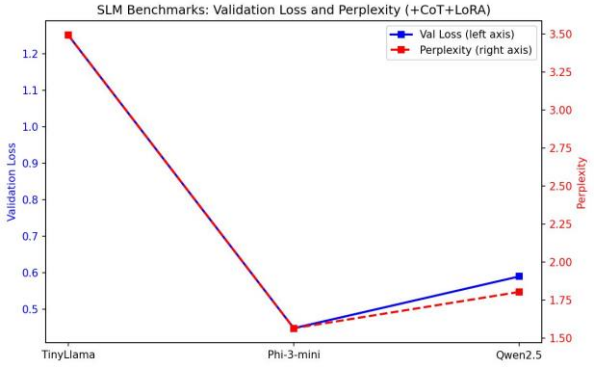


**Fig. 3:** Validation loss and perplexity under +CoT+LoRA across all three SLMs. Phi-3-mini achieves the lowest loss (0.4474) and perplexity (1.56)

**Table 2:** Full ablation results. Bold values indicate best performance per metric per model

| Model | Config | CoT | LoRA | Exact Match | Step Accuracy | Self-Consistency | Loss | Perplexity |
|---|---|---|---|---|---|---|---|---|
| TinyLlama-1.1B | Baseline | ✗ | ✗ | 1.0% | 22.8% | 3.8% | — | — |
| TinyLlama-1.1B | +CoT | ✓ | ✗ | 2.2% | 32.7% | 2.2% | — | — |
| TinyLlama-1.1B | +CoT+LoRA | ✓ | ✓ | 2.2% | 46.4% | 3.0% | 1.2505 | 3.49 |
| Phi-3-mini | Baseline | ✗ | ✗ | 8.2% | 40.8% | 13.0% | — | — |
| Phi-3-mini | +CoT | ✓ | ✗ | 4.6% | 35.5% | 10.8% | — | — |
| Phi-3-mini | +CoT+LoRA | ✓ | ✓ | 16.0% | 61.7% | 9.2% | 0.4474 | 1.56 |
| Qwen2.5-1.5B | Baseline | ✗ | ✗ | 11.4% | 37.7% | 2.4% | — | — |
| Qwen2.5-1.5B | +CoT | ✓ | ✗ | 16.6% | 46.0% | 3.4% | — | — |
| Qwen2.5-1.5B | +CoT+LoRA | ✓ | ✓ | 12.8% | 55.8% | 6.8% | 0.5893 | 1.80 |

The ordering matches the exact-match ordering reported in Table 2, which provides additional evidence for the consistency of the observed results: the model that is most confident in its outputs after fine-tuning is also the one that produces the most correct final answers. The magnitude of the difference is also noteworthy: perplexity scales multiplicatively, so the difference between Phi-3-mini at 1.56 and TinyLlama at 3.49 is substantially larger than the linear distance suggests, and this is consistent with the much wider exact-match gap (16.0% versus 2.2%) the two models show.

The step-accuracy pattern is the most consistent signal in the entire study. Every model shows monotonic improvement across the three configurations, with no regressions on any of the bar transitions: TinyLlama climbs from 22.8% to 32.7% to 46.4%, Phi-3-mini from 40.8% to 35.5% to 61.7% (the dip at +CoT being the format-conflict effect discussed below, recovered at +CoT+LoRA), and Qwen2.5 from 37.7% to 46.0% to 55.8%. These results indicate that step accuracy responds reliably to both interventions, even where exact match does not. This matters because it suggests CoT and LoRA are genuinely improving the model's intermediate reasoning quality independently of whether the final number lands correctly, indicating improved intermediate reasoning even when the final prediction remains incorrect. For applications where transparency or partial credit matters more than strict final-answer accuracy, Fig. 4 may provide a more informative representation of the reasoning quality of the results than Fig. 5.

Exact match does not move in the same direction as step accuracy. Two of the three models (Phi-3-mini and Qwen2.5) show non-monotonic curves on this metric, which motivates the per-component ablation in Table 3. Phi-3-mini dips at +CoT before recovering past baseline at +CoT+LoRA, producing a characteristic V-shaped performance trend. Qwen2.5 shows the inverse pattern: it peaks at +CoT (16.6%) and then loses ground when LoRA is added (12.8%), which may be attributable to truncation effects associated with the 128-token sequence limit rather than the LoRA adaptation itself.

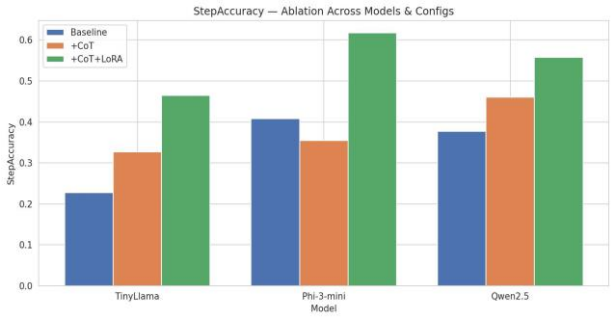


**Fig. 4:** Step accuracy across all models and configurations. All three models show monotonic improvement, confirming CoT+LoRA consistently enhances intermediate reasoning quality

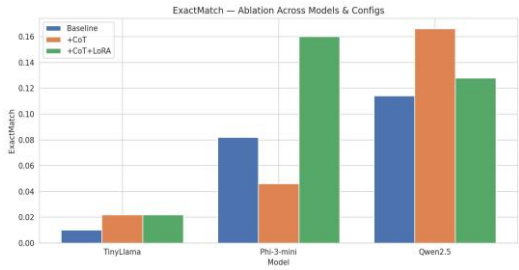


**Fig. 5:** Exact match accuracy across models and configurations. Phi-3-mini degrades under +CoT alone (8.2% → 4.6%) but recovers strongly with +CoT+LoRA (16.0%), confirming CoT must be paired with LoRA for instruction-tuned models

TinyLlama is the only model whose exact-match curve is flat-to-rising throughout, but its absolute values (1.0% to 2.2%) are too low to support strong conclusions regarding the observed trend. Read together with Fig. 4, the message is that CoT and LoRA push intermediate reasoning quality up reliably, but final-answer accuracy depends on additional factors (chiefly architecture, instruction-tuning history, and sequence-length budget) that vary across the three models tested.

### *Ablation Analysis: Component Contributions*

To attribute performance gains to individual components, we isolate the effect of CoT prompting and LoRA fine-tuning separately. Table 3 quantifies the individual contribution of each component to overall performance. CoT contribution is measured as the delta between +CoT and Baseline; LoRA contribution is the delta between +CoT+LoRA and +CoT.

### *Training Convergence and Stability*

All three models converged cleanly within the two-epoch budget with no signs of overfitting. Table 4 shows epoch-by-epoch loss progression.

### *Memory Efficiency and Hardware Utilization*

The framework remains compatible with consumer-grade hardware throughout all experiments. Table 5 summarizes peak VRAM usage across models.

Figure 6 presents the inference speed and peak VRAM utilization across all models and experimental configurations, providing complementary perspectives on the computational efficiency of the proposed framework. On the left panel, throughput generally decreases with increasing model size: TinyLlama is the fastest at 121–145 tokens/sec, Qwen2.5 sits in the middle at 81–98 tokens/sec, and Phi-3-mini, the largest of the three at 3.8B parameters, is the slowest at 43–52 tokens/sec. The roughly 3× gap between TinyLlama and Phi-3-mini reflects the computational trade-off associated with the higher reasoning performance Phi-3-mini holds in Fig. 5, and it is the reason the deployment guidance in Table 6 routes latency-sensitive use cases (edge, IoT) to TinyLlama rather than to the strongest reasoner.

**Table 3:** Component contribution analysis. pp = percentage points

| Model | CoT Δ (EM) | LoRA Δ (EM) | CoT Δ (Step) | LoRA Δ (Step) |
|---|---|---|---|---|
| TinyLlama-1.1B | +1.2 pp (+120%) | 0.0 pp (0%) | +9.9 pp | +13.7 pp |
| Phi-3-mini | −3.6 pp (−44%) | +11.4 pp (+248%) | −5.3 pp | +26.2 pp |
| Qwen2.5-1.5B | +5.2 pp (+46%) | −3.8 pp (−23%) | +8.3 pp | +9.8 pp |

**Table 4:** Training convergence (+CoT+LoRA). Consistent loss reduction across both epochs for all models

| Model | Epoch 1 Loss | Epoch 2 Loss | Trainable Params | Trainable % | Perplexity |
|---|---|---|---|---|---|
| TinyLlama-1.1B | 1.5406 | 1.2505 | 563,200 | 0.051% | 3.49 |
| Phi-3-mini | 0.6372 | 0.4474 | 1,572,864 | 0.041% | 1.56 |
| Qwen2.5-1.5B | 0.8448 | 0.5893 | 544,768 | 0.035% | 1.80 |

**Table 5:** VRAM usage and hardware compatibility. All models fit within 24 GB VRAM, enabling local training without cloud dependency

| Model | VRAM (GB) | Max Batch Size | Trainable Params | Memory Efficiency |
|---|---|---|---|---|
| TinyLlama-1.1B | 2.35 | 24 | 563K (0.051%) | Excellent |
| Phi-3-mini | 10.11 | 6 | 1,572K (0.041%) | Good |
| Qwen2.5-1.5B | 13.46 | 4 | 544K (0.035%) | Moderate |

**Table 6:** Computational cost and deployment guide. Training times based on 2 epochs on 1,164 samples with NVIDIA RTX 4500 Ada

| Model | Min VRAM | Training Time | Tokens/sec | Best Use Case |
|---|---|---|---|---|
| TinyLlama-1.1B | 4 GB | ~60 min | 121–145 | Edge / IoT deployment |
| Phi-3-mini | 12 GB | ~120 min | 43–52 | Accuracy-critical tasks |
| Qwen2.5-1.5B | 16 GB | ~90 min | 81–98 | Balanced performance |

Importantly, the spread within each model across the three configurations is narrow, and adding CoT prompts and LoRA adapters does not meaningfully slow inference. The right panel makes a similar pattern is observed for memory utilization: peak VRAM is essentially flat across configurations within each model (TinyLlama at 2.35 GB, Phi-3-mini at 10.11 GB, Qwen2.5 at 13.46 GB), with LoRA contributing negligible overhead. All three models stay well within the 16 GB threshold marked on the figure, which is the practical line for consumer-grade GPUs.

### *Quality and Interpretability of Reasoning*

In addition to quantitative improvements, qualitative differences in reasoning behavior are observed under the +CoT+LoRA configuration. Where baseline models tend to skip straight to an answer (often incorrectly), enhanced models work through each step explicitly. The example below illustrates this.

Problem: A bakery sold 45 cupcakes in the morning and 38 in the afternoon. Each cupcake costs $2.50. How much did the bakery make?

### *Enhanced Model Output*

Step 1: Calculate total cupcakes sold (45 + 38 = 83)
Step 2: Calculate total revenue (83 × $2.50 = $207.50)

Conclusion: The bakery generated $207.50 in revenue.

Without the CoT enhancement, baseline outputs on problems like this tend to skip straight to a final number, without providing intermediate reasoning steps, thereby limiting interpretability and error analysis.

Across the 500 GSM8K test problems, the dominant failure modes can be inferred from the gap between step accuracy and exact match, which is large for every model (for example, 61.7% versus 16.0% for Phi-3-mini and 55.8% versus 12.8% for Qwen2.5-1.5B). This pattern indicates that the most common failures are not a collapse of reasoning but failures at the final answer: the model performs correct intermediate steps yet does not converge on the exact final value. Two mechanisms account for most of these cases. The first is premature truncation, in which a reasoning chain is cut off by the 128-token limit before it concludes, a mechanism that may also contribute to the performance decline observed for Qwen2.5-1.5B under +CoT+LoRA. The second is final-step arithmetic or answer-extraction error, in which an otherwise sound chain terminates on an incorrect or unparseable number. For TinyLlama-1.1B, persistently low exact match alongside rising step accuracy reflects a capacity limit rather than a formatting problem: the model acquires reasoning structure but rarely completes to a correct final value. Format-following itself is not a dominant failure mode after fine-tuning; the one clear format-level failure is Phi-3-mini under +CoT without LoRA, an instruction-format clash rather than a reasoning failure. As per-problem error categories were not logged during the original runs, these modes are characterised from the aggregate metric patterns rather than from an instance-level audit.

## Discussion

### *Implications for Small Language Model Development*

The findings of this study suggest that the capability gap between large and small language models may be less rigid than is often assumed. The combination of CoT prompting and LoRA fine-tuning enables SLMs to achieve reasoning capabilities that were previously associated with substantially larger architectures. This has important practical implications educational tools that do not need cloud infrastructure, edge devices that can handle non-trivial reasoning locally (Liu et al., 2024), smaller organizations seeking to deploy capable AI systems without access to enterprise-scale computational resources without enterprise-scale compute budgets (Agarwal et al., 2024).

These findings do not imply parity between SLMs and larger frontier models. However, the observed performance gap appears smaller than parameter count alone would suggest, and it is narrowing further with the right training approach.

### *Technical Insights and Methodological Contributions*

The experimental findings provide several important insights into the interaction between Chain-of-Thought (CoT) prompting and parameter-efficient fine-tuning in Small Language Models (SLMs). While both techniques have been studied independently in prior literature, their combined behavior was found to vary considerably across model architectures, highlighting the importance of architecture-aware adaptation strategies.

One of the most notable observations emerged from the instruction-tuned Phi-3-mini model. The introduction of CoT prompting without fine-tuning led to a reduction in reasoning performance, suggesting a conflict between the model's pre-existing instruction-following behavior and the newly imposed reasoning format. However, once LoRA-based adaptation was introduced, the model was able to align effectively with the CoT structure and exhibited substantial performance recovery. This finding indicates that for instruction-tuned SLMs, prompting strategies alone may be insufficient and should be accompanied by lightweight adaptation mechanisms to fully realize their benefits.

Across all evaluated models, improvements in intermediate reasoning quality were more consistent than improvements in final-answer accuracy. The observed gains in step-level reasoning suggest that CoT prompting encourages models to externalize and organize their reasoning process more effectively. Even in cases where exact-match performance did not increase proportionally, the models demonstrated stronger reasoning traces and greater transparency in arriving at their solutions. This distinction highlights the value of evaluating reasoning systems using metrics beyond final-answer correctness, particularly in applications where interpretability and explainability are important.

The study also demonstrates the effectiveness of parameter-efficient fine-tuning for reasoning enhancement. A single LoRA configuration was successfully applied across all evaluated architectures without requiring model-specific redesigns, indicating that lightweight adaptation can generalize across diverse SLM families. The ability to improve reasoning capabilities while updating only a very small fraction of model parameters reinforces the practicality of PEFT approaches for resource-constrained deployments.

From a deployment perspective, the framework achieved meaningful reasoning improvements while maintaining extremely low computational overhead. Training was conducted entirely on consumer-grade hardware with minimal additional memory requirements, demonstrating that advanced reasoning capabilities can be developed without reliance on datacenter-scale infrastructure. This finding is particularly relevant for educational institutions, small organizations, edge deployments, and research environments where computational resources are limited.

Collectively, these observations suggest that the effectiveness of reasoning enhancement in SLMs depends not only on model size but also on the alignment between prompting strategies, model architecture, and adaptation techniques. The results support the view that carefully combining CoT prompting with parameter-efficient fine-tuning offers a practical pathway toward improving reasoning performance in compact language models while preserving computational efficiency.

The ablation study revealed several noteworthy findings. One of the most practically significant findings concerns the Phi-3-mini model CoT prompting alone resulted in reduced performance. For a model that is already instruction-tuned, introducing a CoT prompt format without corresponding fine-tuning creates a conflict between the format the model was trained to follow and the format the prompt is asking for. Model performance consequently declines. LoRA adaptation appears to mitigate this issue by aligning model parameters with the new reasoning format, and the recovery is substantial. These findings suggest that for instruction-tuned SLMs, do not apply CoT prompting without pairing it with fine-tuning. When applied in isolation, CoT prompting may result in performance degradation.

For models without instruction tuning TinyLlama and Qwen2.5 CoT prompting provides immediate benefits by encouraging more structured reasoning processes. The step accuracy metric is particularly telling in these cases. Step accuracy continues to improve even when exact-

match accuracy remains relatively unchanged, which suggests the models are developing stronger intermediate reasoning processes, even when improvements in final-answer accuracy are limited.

LoRA rank r = 4 with alpha = 16 held up across all three architectures, keeping trainable parameter overhead below 0.05% in every case. That is a meaningful result practically it means the framework does not need to be reconfigured per model to get reasonable results.

The improvement was statistically significant, supporting the conclusion that LoRA adaptation mitigates the instruction-format conflict.

### *Limitations and Challenges*

Limitations of the current study include the utilization of a maximum sequence length of 128 tokens during the training and inference processes. Although such an approach allowed us to conduct our experiments efficiently using ordinary consumer equipment, we found ourselves limited in terms of generating long enough reasoning chains needed for complex, multistep reasoning tasks. Specifically, it appears that although the models showed progress in step by step performance with increasing numbers of steps taken, their exact-match scores did not improve accordingly in the case of Qwen2.5-1.5B. We speculate that reasoning processes may be cut off prematurely, failing to solve tasks to completion. This truncation effect is also tokenizer-dependent. Because the three models use different tokenizers, an identical reasoning chain is segmented into a different number of tokens in each, so a model with a less token-efficient encoding reaches the 128-token ceiling sooner. This interaction between tokenization and the sequence-length cap, rather than prompting or fine-tuning in isolation, is the most likely explanation for the Qwen2.5-1.5B exact-match regression under +CoT+LoRA, where step accuracy continues to rise even as final answers are cut off. A further consideration concerns pre-existing capability: Phi-3-mini and Qwen2.5 are instruction-tuned, whereas TinyLlama-1.1B-Chat is only lightly chat-tuned, which raises their absolute baseline scores and could bias a naive cross-model comparison. Our ablation mitigates this by attributing each technique's contribution through within-model deltas, that is, improvement measured relative to each model's own baseline, rather than through absolute cross-model scores; the Phi-3-mini format-conflict result is itself a direct consequence of prior instruction-tuning and is therefore reported as a finding rather than treated as a hidden confound.

### *Limited Training Budget*

training procedure used for this work was designed to be on the conservative side, with only two epochs used on 1,164 training instances. While all models showed clear signs of stability and training losses decreased consistently for each model, these experiments did not determine the upper boundaries of what could be achieved with more training iterations or other methods. More epochs, additional data, application of learning curriculum techniques, or even adaptive learning rates can further enhance reasoning capabilities.

### *Self-Consistency and Reasoning Stability*

A further interesting observation was the relatively low self-consistency values obtained in multiple experimental setups. Despite the enhancement of intermediate reasoning quality via CoT prompting and LoRA fine-tuning, the model often returned inconsistent results, with multiple generations being needed before reaching an agreement on the right answer. This points to the existence of some form of reasoning instability that cannot be detected via exact-match accuracy or step by step metrics. These results show that even if the reasoning quality improves, its reliability may stay unaltered. Future research could take advantage of the training of consistency metrics, self-checks for reasoning, etc.

### *Model Selection Scope*

Three Small Language Models, namely TinyLlama-1.1B, Phi-3-mini-4k-instruct, and Qwen2.5-1.5B, were used for the analysis in this experiment. While the architecture and use case of these small language models might be different, the results are not generalizable across other SLMs. There could be differences in the way CoT prompting and parameter-efficient fine-tuning interact depending on factors such as other types of training approaches or pre-training methods.

### *Evaluation Metric Limitations*

The evaluation criteria are formed on the basis of exact match accuracy, step accuracy, self-consistency, validation loss, and perplexity. Although all these measures give useful information about the quality of reasoning carried out by an agent, they do not sufficiently take into account reasoning correctness and consistency. In turn, the exact match accuracy can underrate a solution that is partially right, while step accuracy is mostly concerned with calculating numeric intermediate values. Moreover, self-consistency was assessed based on a relatively small number of reasoning instances equal to three.

### *Generalization and Robustness Concerns*

The proposed approach was tested in a controlled environment using benchmarks. The real-life deployment environment usually comprises noisy inputs, ambiguously defined problems, adversarial examples, and domain changes that may not be present in the benchmarks used to test the algorithm. Thus, it is still unclear whether the

robustness of the proposed approach extends beyond the controlled environment and holds true for practical deployment.

### *Hardware and Resource Considerations*

Although the framework was intentionally designed for consumer-grade hardware, the experiments were conducted on an NVIDIA RTX 4500 Ada GPU equipped with 24 GB VRAM. This hardware configuration may still exceed the resources available to many educational institutions, small organizations, and edge deployment scenarios. Performance, training time, and memory utilization may vary considerably across lower-end hardware platforms. Therefore, further optimization through quantization, pruning, or mixed-precision techniques may be required for broader accessibility.

### *Interpretability and Reasoning Faithfulness*

While the model architecture was created with consumer hardware in mind, the testing was done using an NVIDIA RTX 4500 Ada GPU with 24 GB VRAM. While such hardware is definitely more than enough to run any machine learning experiment, it is also likely to be out of reach for many academic environments, smaller organizations, and edge computing applications. Thus, further optimizing the model via quantization or mixed precision might be necessary to accommodate less powerful hardware systems.

### *Reproducibility Across Tasks and Domains*

In conclusion, the current work concentrates on the improvement of mathematical reasoning skills. While the potential of using CoT prompts with LoRA fine-tuning for other domains including medicine, law, science, software development, and multimodal reasoning is not clear, further studies should validate the findings in these areas for the technique to be recognized as a general approach for improving reasoning abilities of Small Language Models.

### *Practical Implementation*

The hardware requirements here are genuinely accessible by research standards (Table 6). TinyLlama trains in around 60 minutes on 4 GB VRAM, the minimum viable on almost any modern GPU. Phi-3-mini needs 12 GB and around two hours, which puts it within reach of mid-range consumer cards. Qwen2.5 sits at 16 GB and 90 minutes, reasonable for anyone with a higher-end consumer or entry-level workstation GPU.

### *Comparison With Prior Works*

Compared against prior work on GSM8K, the accuracy numbers here are lower than methods using full fine-tuning on larger models. This outcome is expected given the differences in model scale and training methodology. A more appropriate comparison concerns computational efficiency and resource requirements: prior methods achieving 20–35% GSM8K accuracy (Plaat et al., 2026) typically rely on full parameter updates across models ranging from 1.5B to 540B parameters, requiring datacenter-scale hardware estimated at $200–$1,000+ per run. Our framework touches fewer than 0.051% of parameters, trains on a single consumer GPU, and costs under $0.31 per model in electricity (Table 7; Fig. 7).

The comparison with prior studies presented in Table 7 reveals a trade-off between accuracy and computational efficiency. On raw exact-match accuracy alone, the orange bars representing our three configurations sit below most of the prior methods: Fu et al. (2023) at roughly 35% and Magister et al. (2023) at roughly 20% are clearly higher than our best result of 16.0% with Phi-3-mini.

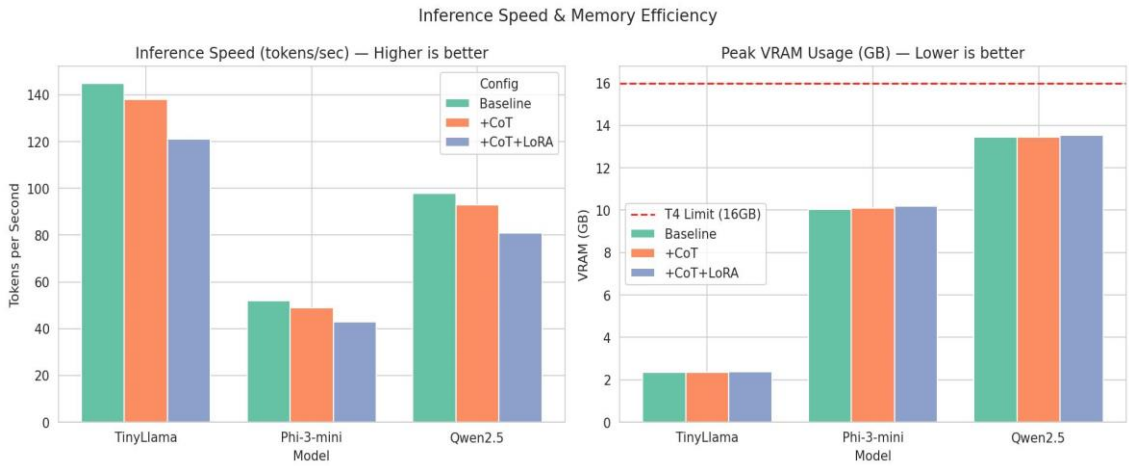

**Fig. 6:** Inference speed (tokens/sec, left) and peak VRAM (GB, right) across all models and configurations. All models remain within the 16 GB threshold; LoRA adds minimal memory overhead

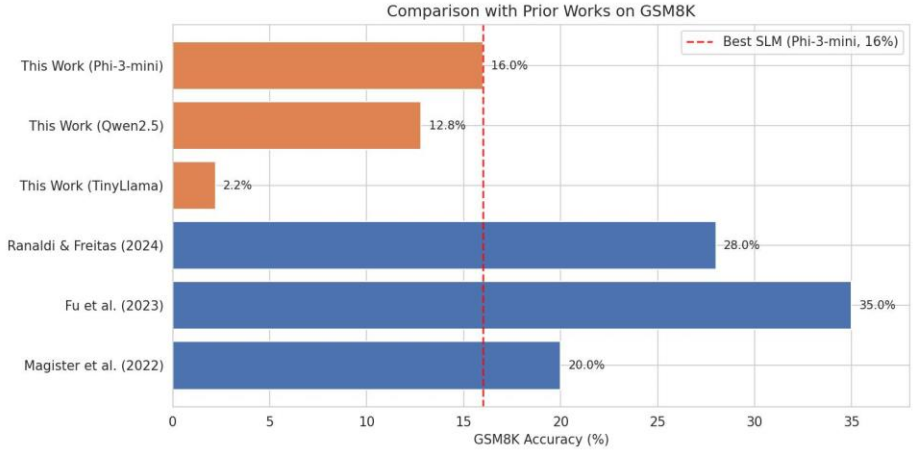


**Fig. 7:** GSM8K exact match accuracy comparison. Orange bars = this work. Prior methods use larger models (1.5B–540B) with full fine-tuning; our framework uses LoRA with fewer than 0.05% trainable parameters on consumer hardware

**Table 7:** Comparison with prior works on GSM8K. Primary contribution is parameter efficiency, not peak accuracy. † Cost estimates; exact figures not published in original papers

| Work | Method | Model Size | GSM8K Acc. | Trainable | Min. VRAM | Est. Cost |
|---|---|---|---|---|---|---|
| Wei et al. (2022) | CoT Prompting | 540B | — | 100% | >1 TB | ~$1M+ |
| Magister et al. (2023) | CoT Distillation | 1.5B | ~20% | 100% | ~40 GB | ~$200–500† |
| Fu et al. (2023) | Specialization | 7B | ~35% | 100% | ~80 GB | ~$500–1,000† |
| Ranaldi and Freitas (2024) | CoT Transfer | 7B | ~28% | 100% | ~80 GB | ~$500–1,000† |
| This Work (TinyLlama-1.1B) | CoT + LoRA | 1.1B | 2.2% | 0.051% | 4 GB | $0.08 |
| This Work (Qwen2.5-1.5B) | CoT + LoRA | 1.5B | 12.8% | 0.035% | 16 GB | $0.31 |
| This Work (Phi-3-mini) | CoT + LoRA | 3.8B | 16.0% | 0.041% | 12 GB | $0.23 |

As shown in Fig. 7, the proposed framework achieves lower exact-match accuracy than several prior approaches. However, the figure also highlights that those higher bars are produced by full fine-tuning of much larger models (1.5B–540B parameters) on infrastructure that costs hundreds to thousands of dollars per run, while our bars come from updating fewer than 0.05% of parameters at a per-model electricity cost under $0.31. The primary contribution of this work lies in achieving competitive reasoning performance while substantially reducing training cost and computational requirements," which is the trade-off the orange-versus-non-orange contrast is designed to communicate.

### *Future Directions*

Several directions for future research emerge from the limitations identified in this study. Longer sequence lengths of 256 to 512 tokens may alleviate limitations on reasoning-chain depth imposed by the current sequence-length constraint, particularly for Qwen2.5. Extended training budgets of 5–10 epochs would help establish whether the convergence trends continue. Consistency-focused training objectives and knowledge distillation from larger teacher models (Gou et al., 2021) both represent promising directions for future investigation.

Cross-domain evaluation on commonsense reasoning benchmarks like HellaSwag, ARC, and yes/no question datasets (Clark et al., 2019), and additional mathematical benchmarks like SVAMP and MATH, would also help establish whether the improvements seen here generalize beyond the specific datasets used.

## Conclusion

This study investigated whether the combination of Chain-of-Thought prompting and LoRA fine-tuning can improve multi-step reasoning performance in Small Language Models and examined the contribution of each component through systematic ablation: can small language models be made meaningfully better at multi-step reasoning by combining Chain-of-Thought prompting with LoRA fine-tuning, and if so, what does each technique actually contribute? After running 9 experiments across 3 models and 3 configurations, the results indicate that the proposed combination is effective, although its impact varies across model architectures and experimental conditions.

The strongest performance was achieved by Phi-3-mini-4k-instruct, which reached 16.0% exact match and 61.7% step accuracy under +CoT+LoRA, a 95% improvement over its own baseline. An equally important finding concerns the behaviour observed prior to fine-tuning: CoT prompting alone dropped Phi-3-mini's exact match by 44%. This observation highlights the value of component-level ablation analysis in a study that only tested the combined configuration. This finding suggests that for instruction-tuned models, CoT prompting and LoRA adaptation may exhibit strong interdependence in instruction-tuned architectures. One without the other can actively make things worse.

Step accuracy improvements held up across every model and every configuration TinyLlama (+23.6 pp), Phi-3-mini (+20.9 pp), Qwen2.5 (+18.1 pp) representing the most consistent trend observed across all experimental configurations. Even where exact match stalled or regressed, the models consistently demonstrated improvements in intermediate reasoning quality. That matters independently of final answer accuracy, especially in applications where interpretability and reasoning transparency are priorities.

From a computational perspective, the results demonstrate the efficiency of the proposed framework. LoRA rank r = 4 with alpha = 16 was sufficient across all three architectures, keeping trainable parameters below 0.05% of total weights roughly 2,000 times fewer than full fine-tuning approaches. Everything ran on a single consumer GPU at under $0.31 per model, with no cloud dependency at any point.

Several limitations should be acknowledged. The 128-token sequence length constraint likely constrained exact-match performance on complex problems, particularly for Qwen2.5 where step accuracy gains did not translate into final answer improvements. The 2-epoch training budget, while sufficient for convergence, is conservative. Self-consistency scores staying low across most configuration points to output instability that better training objectives could address.

These limitations also identify several opportunities for future research. Longer sequences, extended training, consistency-focused objectives none of these require fundamental changes to the framework. The proposed framework demonstrates the feasibility of the approach. Overall, the findings demonstrate that capable, reasoning-oriented SLMs are achievable without datacenter hardware or large training budgets, thereby expanding the feasibility of AI deployment in resource-constrained environments.

## Acknowledgment

The authors gratefully acknowledge the support of the Department of Artificial Intelligence and Machine Learning, Thakur College of Engineering and Technology, for providing the computational infrastructure and academic environment that enabled this work.

## Funding Information

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

## Author's Contributions

**Aryan Raina:** Designed the experimental framework, implemented the training and evaluation pipeline, conducted all experiments, and drafted the manuscript.

**Shiwani Gupta:** Supervised the research, contributed to the methodological design, and reviewed and revised the manuscript for intellectual content.

**Jagruti Jadhav**: Contributed to literature review, ablation design validation, and manuscript review.

**Sampada Bhonde:** Contributed to dataset preparation, experimental validation, and manuscript review.

**Shilpa Mathur:** Contributed in data preprocessing, model implementation.

**Anand Maha:** Analysis and interpretation of results.

**Pranjali Sankhe:** Contributed to the study conception and design, and preparation of figures and tables.

## Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues are involved. All datasets used in this research (GSM8K, Orca-Math, HumanEval) are publicly available benchmarks released under their respective open licenses.

### *Ethics and Societal Considerations*

Easier access to powerful AI brings benefits, but also concerns. Training data shapes pre-built models, and its flaws can be baked right in; adjusting them later may not fix old problems, and sometimes makes them worse without showing obvious signs. As more people start using these systems, checking for unfair patterns needs serious attention, and clarity about what they cannot do must stay front and center. Opening doors wider also opens chances for harm, so how these systems are rolled out matters far more than simply letting anyone use them.

## References

Agarwal, R., Vieillard, N., Stanczyk, P., Ramos, S., Geist, M., & Bachem, O. (2024). On-policy distillation of language models: Learning from self-generated mistakes. *Proceedings of the 12th International Conference on Learning Representations (ICLR 2024), May 7-11*. https://openreview.net/forum?id=3zKtaqxLhW

Braga, M. (2024). Personalized Large Language Models through Parameter Efficient Fine-Tuning Techniques. *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 3076. https://doi.org/10.1145/3626772.3657657

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., & Toutanova, K. (2019). BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. *Proceedings of the 2019 Conference of the North*, 2924–2936. https://doi.org/10.18653/v1/n19-1300

Dettmers, T., Pagnoni, A., Holtzman, A., & Zettlemoyer, L. (2023). QLoRA: Efficient Finetuning of Quantized LLMs. *Advances in Neural Information Processing Systems 36*, 10088–10115. https://doi.org/10.52202/075280-0441

Ding, N., Qin, Y., Yang, G., Wei, F., Yang, Z., Su, Y., Hu, S., Chen, Y., Chan, C.-M., Chen, W., Yi, J., Zhao, W., Wang, X., Liu, Z., Zheng, H.-T., Chen, J., Liu, Y., Tang, J., Li, J., & Sun, M. (2023). Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, *5*(3), 220–235. https://doi.org/10.1038/s42256-023-00626-4

Dua, D., Wang, Y., Dasigi, P., Stanovsky, G., Singh, S., & Gardner, M. (2019). DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning over Paragraphs. *Proceedings of the 2019 Conference of the North*, 2368–2378. https://doi.org/10.18653/v1/n19-1246

Feng, S., Fang, G., Ma, X., & Wang, X. (2025). Efficient reasoning models: A survey. *Transactions on Machine Learning Research (TMLR) / ArXiv*. https://doi.org/10.48550/arXiv.2504.10903

Fu, Y., Peng, H., Ou, L., Sabharwal, A., & Khot, T. (2023). Specializing smaller language models towards multi-step reasoning. *Proceedings of the 40th International Conference on Machine Learning (ICML 2023), Jul. 23-29, PMLR*, 10421–10430.

Gou, J., Yu, B., Maybank, S. J., & Tao, D. (2021). Knowledge Distillation: A Survey. *International Journal of Computer Vision*, *129*(6), 1789–1819. https://doi.org/10.1007/s11263-021-01453-z

Guan, X. G., Zhang, L. L. Z., Liu, Y. L., Shang, N. S., Sun, Y. S., Zhu, Y. Z., Yang, F. Y., & Yang, M. (2025). rStar-Math: Small LLMs Can Master Math Reasoning with Self-Evolved Deep Thinking. *ArXiv Preprint ArXiv:2501.04519*. https://doi.org/10.48550/arXiv.2501.04519

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2022). LoRA: Low-rank adaptation of large language models. *Proceedings of the 10th International Conference on Learning Representations (ICLR 2022), Apr. 25-29, Virtual Event*. https://openreview.net/forum?id=nZeVKeeFYf9

Hendrycks, D., Burn, Collin, Kadavath, Saurav, Arora, Akul, Basart, S., Tang, E., Song, D., & Steinhardt, J. (2021). Measuring Mathematical Problem Solving with the MATH Dataset. *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks)*. 35th Conference on Neural Information Processing Systems (NeurIPS 2021) Track on Datasets and Benchmarks, Virtual Event. https://doi.org/10.48550/arXiv.2103.03874

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2020). ALBERT: A lite BERT for self-supervised learning of language representations. *Proceedings of the 8th International Conference on Learning Representations (ICLR 2020*. https://openreview.net/forum?id=H1eA7AEtvS

Li, L., Wang Q., Chenglin, L., Chen, C., Tao, F., Li, Y., Chen, Z., & Zhang, Y. (2024a). Mixed Distillation Helps Smaller Language Models Reason Better. *Findings of the Association for Computational Linguistics: EMNLP 2024*, 1673–1690. https://doi.org/10.18653/v1/2024.findings-emnlp.91

Li, Y., Lan, X., Chen, H., Lu, K., & Jiang, D. (2024b). Multimodal PEAR Chain-of-Thought Reasoning for Multimodal Sentiment Analysis. *ACM Transactions on Multimedia Computing, Communications, and Applications*, *20*(9), 1–23. https://doi.org/10.1145/3672398

Lin, J., Tang, J., Tang, H., Yang, S., Xiao, G., & Han, S. (2025). AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration. *GetMobile: Mobile Computing and Communications*, *28*(4), 12–17. https://doi.org/10.1145/3714983.3714987

Liu, B. (2022). *Sentiment Analysis: Mining Opinions, Sentiments and Emotions*. https://doi.org/10.1017/9781108639286

Liu, R., Gao, J., Zhao, J., Zhang, K., Li, X., Qi, B., Ouyang, W., & Zhou, B. (2025). Can 1B LLM Surpass 405B LLM? Rethinking Compute-Optimal Test-Time Scaling. *ArXiv Preprint ArXiv:2502.06703*. https://doi.org/https://arxiv.org/abs/2502.06703

Liu, Z., Zhao, C., Iandola, F., Lai, C., & Tian, Y. (2024). MobileLLM: Optimizing sub-billion parameter language models for on-device use cases. *Proceedings of the 41st International Conference on Machine Learning (ICML 2024)*, 32431–32454.

Magister, L. C., Mallinson, J., Adamek, J., Malmi, E., & Severyn, A. (2023). Teaching Small Language Models to Reason. *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 1773–1781. https://doi.org/10.18653/v1/2023.acl-short.151

Mu, J., Wang, W., Liu, W., Yan, T., & Wang, G. (2025). Multimodal Large Language Model with LoRA Fine-Tuning for Multimodal Sentiment Analysis. *ACM Transactions on Intelligent Systems and Technology*, *16*(6), 1–23. https://doi.org/10.1145/3709147

Nijkamp, E., Pang, B., Hayashi, H., Tu, L., Wang, H., Zhou, Y., Savarese, S., & Xiong, C. (2023). CodeGen: An open large language model for code with multi-turn program synthesis. *Proceedings of the International Conference on Learning Representations*. 11th International Conference on Learning Representations (ICLR 2023), Kigali, Rwanda.

Patel, A., Bhattamishra, S., & Goyal, N. (2021). Are NLP Models really able to Solve Simple Math Word Problems? *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2080–2094. https://doi.org/10.18653/v1/2021.naacl-main.168

Plaat, A., Wong, A., Verberne, S., Broekens, J., Van Stein, N., & Bäck, T. (2026). Multi-Step Reasoning with Large Language Models, a Survey. *ACM Computing Surveys*, *58*(6), 1–35. https://doi.org/10.1145/3774896

Press, O., Zhang, M., Min, S., Schmidt, L., Smith, N., & Lewis, M. (2023). Measuring and Narrowing the Compositionality Gap in Language Models. *Findings of the Association for Computational Linguistics: EMNLP 2023*, 5687–5711. https://doi.org/10.18653/v1/2023.findings-emnlp.378

Ranaldi, L., & Freitas, A. (2024). Aligning Large and Small Language Models via Chain-of-Thought Reasoning. *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1812–1827. https://doi.org/10.18653/v1/2024.eacl-long.109

Samarinas, C., & Zamani, H. (2025). Distillation and Refinement of Reasoning in Small Language Models for Document Re-ranking. *Proceedings of the 2025 International ACM SIGIR Conference on Innovative Concepts and Theories in Information Retrieval (ICTIR)*, 430–435. https://doi.org/10.1145/3731120.3744613

Together Computer, (2023). RedPajama: An open source recipe to reproduce LLaMA training dataset. *LLaMA Training Dataset*. https://github.com/togethercomputer/RedPajama-Data

Vernikos, G., Brazinskas, A., Adamek, J., Mallinson, J., Severyn, A., & Malmi, E. (2024). Small Language Models Improve Giants by Rewriting Their Outputs. *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2703–2718. https://doi.org/10.18653/v1/2024.eacl-long.165

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q. V., & Zhou, D. (2022). Chain-Of-Thought Prompting Elicits Reasoning in Large Language Models. *Advances in Neural Information Processing Systems 35*, 24824–24837. https://doi.org/10.52202/068431-1800

Weyssow, M., Zhou, X., Kim, K., Lo, D., & Sahraoui, H. (2025). Exploring Parameter-Efficient Fine-Tuning Techniques for Code Generation with Large Language Models. *ACM Transactions on Software Engineering and Methodology*, *34*(7), 1–25. https://doi.org/10.1145/3714461

Ye, J., Gong, S., Chen, L., Zheng, L., Gao, J., Shi, H., Wu, C., Jiang, X., Li, Z., Bi, W., & Kong, L. (2024). Diffusion of Thoughts: Chain-of-Thought Reasoning in Diffusion Language Models. *Advances in Neural Information Processing Systems 37*, 105345–105374. https://doi.org/10.52202/079017-3343

Zhang, Q., Lyu, F., Sun, Z., Wang, L., Zhang, W., Hua, W., Wu, H., Guo, Z., Wang, Y., Muennighoff, N., King, I., Liu, X., & Ma, C. (2025). A survey on test-time scaling in large language models. *ArXiv Preprint ArXiv:2503.24235*. https://doi.org/10.48550/arXiv.2503.24235