Improving Reasoning of Small Reasoning Models with Masked Critique Fine-tuning

Akalbir Singh Chadha
University of Southern California
Los Angeles
California
akalbirs@usc.edu

Chandresh Mallick
University of Southern California
Los Angeles
California
cmallick@usc.edu

Abstract

Small reasoning models (SRMs) often struggle with tasks requiring complex multi-step reasoning, lagging behind their larger counterparts. We present a method to enhance the reasoning capabilities of SRMs via masked critique fine-tuning. In our approach, a powerful teacher model generates detailed reasoning solutions and accompanying critiques for a curated set of problems (the LIMO dataset, expanded to 2,351 examples). We then mask selected tokens in the teacher's critiques—forcing the student model to infer the missing pieces—and fine-tune a 1.5B-parameter SRM (DeepScaleR-1.5B) on this data. Training on these partially masked critiques encourages the small model to generalize its reasoning steps rather than memorize solutions. Empirically, our fine-tuned model significantly outperforms baseline counterparts on reasoning benchmarks MATH 500 and AMC. These results demonstrate that masked critique fine-tuning effectively imparts advanced reasoning skills to a 1.5B model, yielding substantial performance gains over conventional fine-tuning and narrowing the gap between small and large-scale models

1 Introduction

Recent advancements in Large Language Models (LLMs) have demonstrated remarkable improvements in reasoning tasks. Models such as GPT-4o, Claude 3, and Gemini 2.0 have set new benchmarks in reasoning abilities. However, a new category of models known as Reasoning Language Models (RLMs) has emerged, including OpenAI's o1 and o3, Claude Thinking, and DeepSeek-R1, which generate intermediate reasoning steps before producing final answers. This self-reflective reasoning process allows them to self-correct and achieve superior performance.

Despite these impressive results, most RLMs are computationally intensive, rendering them impractical for real-world applications where resources are limited. This challenge motivates the development of Small Reasoning Models (SRMs) that combine robust reasoning capabilities with computational efficiency. Prior research has explored methods such as Chain of Thought (CoT) prompting and fine-tuning on reasoning datasets, Monte Carlo Tree Search (MCTS), and Process Reward Models (PRMs[1]) to enhance reasoning abilities (Zhang et al., 2024; Zeng et al., 2024; Zhao et al., 2024; Wei et al., 2023). Notably, the DeepSeek-R1 report indicates that reinforcement learning naturally encourages the emergence of coherent reasoning chains, thereby bolstering overall model performance.

Our methodology leverages critique-based fine-tuning on a high-quality dataset—where intermediate reasoning chains are refined using masked token supervision—to enable a distilled small reasoning model to internalize robust reasoning patterns. We use critique fine-tuning instead of supervised fine-tuning because Wang et al. (2025b) showed that critique finetuning leads to better generalization than supervised finetuning. In this study, we show such an approach can match or even surpass baseline small language models in complex reasoning tasks, offering an efficient yet powerful alternative for real-world applications.

1.1 Research Question

Can a small reasoning model, distilled from a larger reasoning model and fine-tuned on carefully selected data, outperform baseline small language models in reasoning tasks while maintaining computational efficiency?

2 Related Work

Large Language Models (LLMs) have revolutionized natural language processing by demonstrating impressive reasoning and problem-solving capabilities. Models such as GPT-4o,

[1] https://opensource-o1.github.io/

Claude 3.5 Sonnet, and Gemini 2.0 not only generate fluent text but also perform complex inference tasks across diverse domains (Brown et al., 2020). However, the standard paradigm of next-token prediction does not inherently encourage structured, step-by-step reasoning. This limitation has prompted research into explicit methods that improve logical inference and transparency in these models (Bommasani et al., 2021). In this survey, we review several key approaches that enhance reasoning in LLMs, including chain-of-thought prompting, tree-based search techniques, process supervision, and reinforcement learning strategies, as well as methods for adapting these techniques to smaller models.

2.1 Chain-of-Thought (CoT) Prompting and Self-Consistency

Chain-of-Thought (CoT) prompting encourages models to generate intermediate reasoning steps before arriving at a final answer. This approach not only improves performance on complex tasks such as mathematical problem solving and logical inference but also provides interpretability to the model's reasoning process (Wei et al., 2022). Research has shown that when a model is prompted to "think step-by-step," it is more likely to produce a coherent chain of reasoning that leads to a correct answer. Furthermore, variants such as Zero-Shot CoT demonstrate that even without fine-tuning, models can generalize this strategy effectively (Kojima et al., 2022). The Self-Consistency method builds on CoT by sampling multiple reasoning paths and selecting the answer that appears most frequently, thereby reducing errors caused by occasional lapses in logical coherence (Wang et al., 2022).

2.2 Tree-of-Thoughts (ToT) and Monte Carlo Tree Search (MCTS)

While CoT produces a single linear reasoning path, Tree-of-Thoughts (ToT) allows models to explore multiple branches of reasoning concurrently, akin to human deliberation (Yao et al., 2023). This method supports the exploration of alternative solution strategies before converging on a final answer, which is particularly useful in problems with multiple valid reasoning paths. Complementarily, Monte Carlo Tree Search (MCTS), a technique originally developed for game-playing AI (Silver et al., 2016), has inspired search-based methods in the context of LLM reasoning. These methods dynamically evaluate and select the most promising reasoning trajectories, allowing models to backtrack from dead-end paths and increase overall reasoning accuracy—especially in complex domains such as mathematical proofs and logic puzzles.

2.3 Supervised Fine-Tuning on CoT data

The most direct method, where the LLM is fine-tuned using a standard language modeling objective (like cross-entropy loss) on a dataset containing questions paired with detailed CoT reasoning chains and final answers. The model learns to replicate the structure and content of the reasoning steps provided in the training data (Luong et al., 2024). While effective for imparting basic CoT capabilities, its success hinges on the availability of high-quality, annotated CoT data, which can be resource-intensive to create (Wang et al., 2025a). Studies also suggest SFT might primarily teach the model to replicate the structural patterns of reasoning rather than deeply absorbing the semantic content (Lobo et al., 2025).

To improve SFT efficiency and potentially the quality of generated CoTs by focusing on the most critical reasoning steps. It uses perplexity—a measure of model uncertainty—to identify essential steps within a CoT. A step is deemed critical if its removal significantly increases the model's perplexity on the remaining sequence. The SFT dataset is then constructed using examples that retain these critical steps, while less important steps might be removed or merged to maintain logical flow. This encourages the model to learn more concise yet effective reasoning paths (Cui et al., 2025).

To foster more structured and consistent reasoning, this method introduces special, learnable "planning tokens" at the beginning of each reasoning step during SFT. These tokens act as high-level plans or indicators of the operation type for the subsequent step. Various methods exist for assigning these tokens, including rule-based approaches (Arithmetic), clustering step embeddings (K-Means), or learning discrete representations via VQ-VAE (SQ-VAE). The embeddings for these planning tokens are trained alongside the model (compatible with full FT or PEFT like LoRA), adding minimal parameter overhead (<0.001%). By training the model to predict the planning token before the detailed step, it aims to guide the generation hierarchically, improving logical flow and consistency, with reported accuracy gains on math and QA benchmarks (Wang et al., 2024a).

Addressing the issue of variability and instability in generated CoTs, SCoT incorporates an explicit strategic planning phase into the SFT process. For each training instance, the model might be trained to first generate a high-level problem-solving strategy (termed strategic knowledge) before generating the detailed CoT steps and answer. This strategy should ideally be correct, comprehensive, and relatively simple to follow. By grounding the step-by-step reasoning in an explicit plan, SCoT aims to produce more reliable and logically sound CoT paths (Wang et al., 2024b).

The technique by Yu et al. focuses on optimizing the SFT process by tailoring the training data difficulty to the specific LLM being trained. It first grades the difficulty of potential training questions based on the target LLM's own reasoning performance (e.g., assessing CoT correctness). Then, questions are sampled for the SFT dataset according to a desired difficulty distribution, ensuring the model trains on problems appropriate for its current capabilities. This adaptive approach aims to maximize the learning signal, particularly when using high-quality CoT data distilled from more capable models, potentially reducing data generation costs and enhancing SFT efficiency (Yu et al., 2025a).

2.4 Reinforcement Learning for Reasoning in Small Language Models

Recent work has explored reinforcement learning (RL) to boost the reasoning and generalization of small language models, contrasting outcome-based reward schemes (rewarding final answers) with process-based feedback (rewarding intermediate reasoning quality), and comparing value-free policy optimization to value-based methods. For example, OpenAI's o1 series demonstrated that a small model can achieve more accurate, "System-2" chain-of-thought reasoning by allocating extra inference steps – a strategy known as test-time compute scaling – trading higher computation for improved problem-solving performance[2]. A Hugging Face survey further highlights two paradigms for using this extra compute: self-refinement (the model iteratively corrects its own reasoning process) versus search with a verifier (the model generates multiple solutions and a learned reward model selects the best outcome)[3].

Guan et al. pursue the process-oriented approach in rStar-Math, which uses Monte Carlo Tree Search guided by a small process reward model to generate and vet step-by-step solution trajectories. By fine-tuning on millions of verified reasoning paths and iteratively co-evolving the policy and reward models, this method enabled a 7B model to reach state-of-the-art math performance (e.g. 90% accuracy on MATH, surpassing some GPT-4-based solvers) without any distillation from larger models.

In contrast, Shao et al. adopt an outcome-based reward in DeepSeekMath 7B, pairing extensive math pre-training with a novel RL fine-tuning algorithm called Group Relative Policy Optimization (GRPO). GRPO is a value-model-free variant of PPO that forgoes a learned value critic and instead computes advantages by comparing group outcomes, which improved the model's mathematical reasoning while greatly reducing memory overhead. Using GRPO to optimize final-answer correctness, DeepSeekMath achieved 51.7% on the MATH benchmark without external tools, approaching the performance of much larger proprietary models.

To push outcome-based RL further, Yu et al. introduce DAPO, a scalable open-source RL system that attains competition-level math reasoning results by addressing key RL instabilities. DAPO augments PPO with several techniques – notably a widened clipping range to prevent entropy collapse, token-level credit assignment for long chain-of-thoughts, and a mild length penalty to curb overly long solutions – which together stabilize training (reducing reward noise) and improve sample efficiency. This approach, using a value-free policy gradient similar to GRPO, achieved state-of-the-art accuracy on challenging math contests (50% on AIME) while using only half the training steps of prior methods.

In parallel,Yuan et al. argue for value-model-based methods: their VAPO framework incorporates an explicit learned value function and algorithmic innovations (like length-adaptive GAE for varying solution lengths) to overcome value estimation bias and sparse rewards in long reasoning episodes. With a more reliable critic, VAPO attains a new state-of-the-art (60.4% on AIME), outperforming the best value-free approaches by over 10 points, all while converging stably in as few as 5k steps and with no training divergence.

Finally, Fatemi et al. target the verbosity problem: they observe that unconstrained outcome-based RL fine-tuning often leads models to generate excessively long-winded solutions with

[2] https://huggingface.co/blog/Kseniase/testtimecompute

[3] https://huggingface.co/spaces/HuggingFaceH4/blogpost-scaling-test-time-compute

only marginal gains. To counter this, they propose a two-phase RL regimen—first focusing purely on accuracy and exploration, then introducing a conciseness reward—that dramatically shortens reasoning chains without sacrificing correctness

2.5 Test-Time Compute Scaling for Reasoning

Test-time compute scaling lets a model "think longer or try more paths" at inference, boosting accuracy without changing weights—particularly valuable for small language models (SLMs). We summarise three broad approaches that will complement later sections on prompting, RL, and data-efficient training. Best-of-N sampling or majority-vote self-consistency markedly improves SLMs: a 3B model with an optimal chooser can rival models 100× larger on math questions[4] . Adding a lightweight verifier (process reward model) to rerank the N samples yields further gains and underpins the OpenAI o1 system[5]. Gains plateau for already-strong LMs but remain large for weaker reasoners.

Step-wise search treats reasoning as tree exploration. Beam search with verifier pruning and its diverse extension (DVTS) beat plain sampling on hard tasks, with the benefit largest below 7B[4]. r-star-Math pushes the idea furthest: a 7B policy guided by MCTS + value network attains 90% on MATH and top-20% AIME performance—surpassing much larger closed models—simply by expanding more test-time trajectories (Guan et al., 2025).

Rather than parallel sampling, budget-forcing lets one run deeper reasoning for only the hardest questions. Fine-tuning on just 1k examples, s1 learns a special end-of-thought token; suppressing or triggering it scales reasoning length on demand and yields a smooth accuracy-vs-tokens curve, matching o1-preview at far lower training cost (Muennighoff et al., 2025). Complementary RL work trains models to stop early when continued thinking no longer pays off, cutting average inference tokens while maintaining near-max accuracy (Arora and Zanette, 2025). Study of reasoning models' CoT shows that the reasoning chains are all over the place and reasoning models are prone to abrupt thought switching without a proper justification. To promote efficient reasoning (Wang et al., 2025c) proposed to penalize thought-switching tokens like "alternatively".

[4] https://huggingface.co/spaces/HuggingFaceH4/blogpost-scaling-test-time-compute

[5] https://openai.com/index/learning-to-reason-with-llms

2.6 Data-efficient fine-tuning

Researchers, along with domain experts, generated a small dataset containing difficult questions and optimally structured CoT with verification involving humans & LLMs. This process yielded a small dataset with only 817 samples. Language models trained on this dataset had significant in-domain and out-of-domain performance increases (Ye et al., 2025). Thereby challenging the belief in the need of good very huge datasets for an increase in reasoning. Another approach automated this process. Where a small dataset was extracted from a larger one based on rewards seen during RFT. Later, this small dataset was used to fine-tune a language model (Li et al., 2025).

2.7 Reasoning in the latent space

Yang et al. show that transformer hidden states naturally retrieve intermediate "bridge" entities, revealing latent two-hop reasoning already present in larger models. Goyal et al. insert learnable <pause> tokens that give a 1B parameter model extra internal compute, raising SQuAD exact-match by18% without changing model size.

Wang et al. introduced lightweight planning tokens as discrete latent variables before each reasoning step, boosting math and multi-hop QA with only 0.001% extra parameters. Deng et al.first distil explicit chain-of-thought (CoT) into hidden-state dynamics so a student reasons entirely latently, then show a curriculum that gradually removes CoT tokens, letting GPT-2-small solve 9-digit multiplication and Mistral-7B reach 50% on GSM8K with no visible rationale (Deng et al., 2024).

Yu et al. further distills the outputs of complex multi-step algorithms into a single forward pass, retaining most of the reasoning gains with lower inference cost.

Hao et al. proposed CoCoNut, where continuous latent "tokens" (hidden states fed back into the model) enable breadth-first search in latent space and outperform textual CoT on logical puzzles.

Chowdhury and Caragea design a recurrent-depth transformer whose core block is iterated at test-time; a 3.5B model matches 50B performance when given enough cycles, trading context length for latent computation.

3 Methodology

3.1 Training Data Collection and Curation

The proposed approach involves curating a high-quality dataset of reasoning chains and critiques, followed by a targeted fine-tuning strategy to enhance a small base model's reasoning capabilities. First, we select the base reasoning dataset LIMO[6]. Experiments from the respective papers show that this dataset is of high quality for the chosen base model architecture.

For each question in this dataset, we use a pre-trained small base model (e.g., a 1.5B decoder-only transformer) to generate both a CoT and the final answer. We sample multiple (and different) outputs for a single question (from the base model). These outputs are then critiqued by a state-of-the-art teacher model like Gemini-2.5-pro-exp. The teacher model identifies errors in the reasoning steps (e.g., incorrect assumptions, missing logic) and validates the correctness of the final answer. The teacher model not only generates critique but also rewrites the base model's reasoning chains based on the critique. We also manually review the teacher critiques to ensure high data quality.

The critiques are structured as natural language feedback. To prevent the base model from memorizing specific feedback, we mask some tokens in the critique text, ensuring the model learns generalizable reasoning patterns rather than verbatim corrections. For this, we will simply skip loss computation for masked tokens.

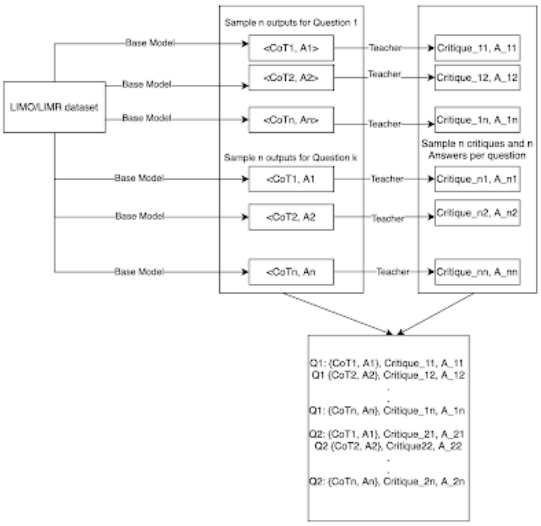
Figure 1 represents the pipeline for generating training data. The base model generates multiple reasoning chains and answers for a given question. These outputs are then reviewed by a teacher model, which provides critiques identifying errors in reasoning. The final dataset consists of questions, reasoning chains, answers, and corresponding critiques along with re-written reasoning chains and final answer, which are later used for fine-tuning.

3.2 Fine-Tuning Strategy

The base model is then fine-tuned on the modified dataset, where the input is a triplet of (user question, base model's CoT and base model's answer), and the target output is the masked critique and rewritten reasoning chains. During training (as seen in Figure 2), the model learns

[6] https://huggingface.co/datasets/GAIR/LIMO

Figure 1: Data Generation Workflow



to predict the original critique and rewritten reasoning chains, effectively internalizing the teacher's feedback about valid and invalid reasoning paths. Crucially, at inference time, the fine-tuned model does not generate critiques but instead produces the answer (As seen in Figure 4). The act of learning to critique during training could reshape its internal representations, suppressing poor reasoning patterns and promoting logically correct CoT generation. This is analogous to a student who, after practicing error detection under expert guidance, inherently avoids mistakes when solving problems independently. By masking parts of the critique, the model is forced to infer missing feedback tokens, which strengthens its ability to generalize and self-correct reasoning steps without explicit supervision.

Figure 2: Training Process (CFT with Masked Tokens)

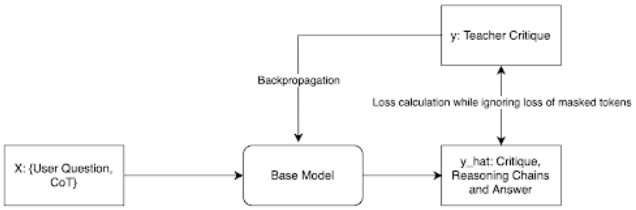


Figure 2 outlines the fine-tuning process, where a base model generates critique and reasoning chains based on critique along with the answer given user question and CoT. The teacher model provides a high-quality critique, which is used as a training target. The base model's out-

put is compared to the teacher critique, and loss is calculated while ignoring loss calculation of masked tokens. Backpropagation updates the base model using this loss, reinforcing improved reasoning patterns while reducing the degree of overfitting.

Figure 3: Sample Input and Output

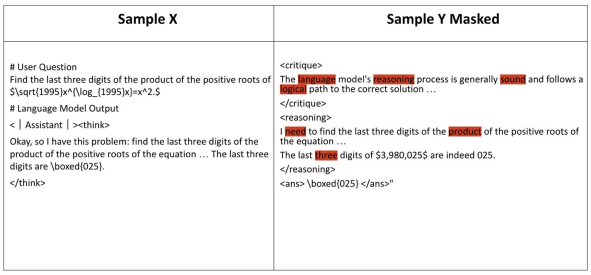


Figure 3 shows sample input X and sample output Y, where tokens are masked.

Figure 4: Flow During Inference Time

Figure 4 represents the inference process, where the fine-tuned model receives a user question as input and produces an answer along with the CoT. Unlike the training phase, the model does not generate a critique but instead internalizes the reasoning learned during fine-tuning, leading to more accurate responses.

3.3 Evaluation Methodology

Our evaluation assesses the efficacy of Masked Critique Fine-Tuning (Blurred Thoughts SFT) in improving the reasoning capabilities of Small Reasoning Models (SRMs). We utilize a rigorous evaluation setup leveraging the LIMO evaluation pipeline, which automates inference, parses boxed answers (`\boxed{}`), and computes exact-match accuracy.

Evaluation Datasets:

1. AIME (2024): A challenging benchmark containing high school-level competition problems emphasizing multi-step reasoning.
2. AMC (2023): Comprising algebra, geometry, and number theory questions commonly used in national mathematics competitions.
3. MATH 500: A curated selection of 500 problems from the broader MATH dataset designed to provide a comprehensive evaluation across varying difficulty levels.
4. Olympiad Bench: Advanced reasoning problems included specifically for assessing high-level, creative mathematical reasoning.

Evaluation Metrics:

- Pass@1 Accuracy: Measures how frequently the model's top answer matches the ground truth. The accuracy is computed using exact matches extracted via boxed answers.

3.4 Data Generation

The teacher model used during data generation is Gemini-2.5-pro-exp, and the LIMO dataset is expanded from 817 samples to 2351 samples. We sampled 3 different outputs for a question. All in all, we should have had 2,451 samples, but due to issues mentioned in the limitations section.

3.5 Fine-Tuning Setup

Our fine-tuning employs Masked Critique Fine-Tuning (Blurred Thoughts SFT), specifically designed to enhance reasoning abilities by training models to internalize critiques within structured reasoning processes. The approach involves:

- Full fine-tuning (no LoRA/adapters).
- Structured input format with <think>, <critique>, <reasoning>, and <ans> tags.
- Partial masking of critique tokens during training, promoting reasoning generalization rather than rote memorization.

Fine-Tuning Details:

- Model: DeepScaleR-1.5B
- Dataset: Expanded LIMO dataset (2351 samples), auto-formatted for critique-aware reasoning.

Training Parameters:

- Epochs: 3
- Effective Batch Size: 8 (Accumulation)
- Max Token Length: 16394
- Learning Rate: 5e-5 (Cosine scheduler)
- Optimizer: Adam8bit (bitsandbytes)
- Precision: bfloat16

- Hardware: A40 GPUs with full-precision fine-tuning and gradient checkpointing enabled.

4 Results

4.1 Baseline Performance Setup

Baseline evaluations were performed on the following SRMs without any quantization or fine-tuning to establish raw performance metrics:

- DeepScaleR-1.5B-Preview
- DeepSeek-R1-Distill-Qwen-1.5B

These initial evaluations as seen in Table 1 enable a clear measurement of the impact resulting from subsequent fine-tuning and optimizations.

4.2 Fine-tuning

Post fine-tuning, the DeepScaleR-1.5B model exhibited significant improvements over baseline models across 2 out of 4 evaluated datasets, demonstrating enhanced multi-step reasoning and critique-aware capabilities.

Baseline vs. Fine-Tuned Pass@1 Accuracy: As seen in Table 2

These results clearly illustrate the efficacy of Masked Critique Fine-Tuning in boosting reasoning performance, especially on datasets requiring deeper logical chains.

4.3 Ablation Study: Masking Threshold Sensitivity

To understand the sensitivity of our fine-tuning approach to masking thresholds within critique sections, we experimented with different masking levels:

- Thresholds Tested: 0.20 (20% masked), 0.15 (15% masked)

Observations:

- Similar convergence behavior across thresholds, with final training loss consistently around 0.3.
- No significant difference in benchmark performance across AIME, AMC, MATH 500, and Olympiad Bench datasets between thresholds.

Insights:

- Due to the dense and detailed nature of critique spans, masking up to 20% still retains sufficient structure for effective learning.
- The model effectively internalized the structure and purpose of critiques, indicating learning beyond mere token memorization.

This ablation reinforces the robustness of Masked Critique Fine-Tuning in encouraging generalizable reasoning capabilities.

5 Limitations

Despite the promising outcomes, several limitations affected this study. Infrastructure instability, particularly downtime of the CARC system used for training and evaluation, occasionally disrupted experimentation timelines. Hardware memory limitations necessitated small batch sizes, potentially constraining convergence speed and limiting the scope of hyperparameter exploration. Data quality issues were also encountered, as some generated training samples exhibited formatting inconsistencies or reasoning errors that required manual intervention. Additionally, the Gemini-2.5 teacher model occasionally produced degenerate or low-quality outputs during critique generation, impacting the quality of certain training samples.

6 Conclusion

This research underscores the potential of Masked Critique Fine-Tuning in augmenting the reasoning abilities of small language models. By integrating structured reasoning traces and masking critique tokens, we enabled the fine-tuned DeepScaleR-1.5B model to achieve improvements across a range of mathematical reasoning benchmarks. The robustness of the methodology, even under varying masking thresholds, suggests that the model learned to internalize critique structures effectively. Our evaluation validates that SRMs, when fine-tuned using well-designed reasoning techniques, can exhibit significant gains in reasoning performance while maintaining manageable computational requirements.

7 Future Scope

Building on the promising results of this study, several avenues for future research emerge. Future work will explore dynamic or adaptive

Table 1: Baseline Pass@1 Accuracy for Mathematical Reasoning Datasets

Model	AIME	AMC	MATH 500	Olympiad Bench
DeepScale-R1-1.5B Preview	30.00	60.00	60.40	39.70
DeepSeek-R1-Distill-Qwen-1.5B	23.33	52.50	69.60	29.48

Table 2: Baseline and Fine-Tuned Pass@1 Accuracy for Mathematical Reasoning Datasets

Model	AIME	AMC	MATH 500	Olympiad Bench
DeepScale-R1-1.5B Preview (Baseline)	30.00	60.00	60.40	39.70
DeepSeek-R1-Distill-Qwen-1.5B (Baseline)	23.33	52.50	69.60	29.48
DeepScaleR-1.5B (Fine-Tuned)	23.33	**67.50**	**72.40**	39.26

masking strategies to optimize critique integration further. Moreover, combining Masked Critique fine-tuning with Reinforcement Fine-tuning (RFT) can lead to better generalization. RFT will eliminate the need of a teacher model, thereby saving budget spent on data generation. Adding, heuristic rewards depending on output format, reasoning chains format and critique format can be an area of exploration.

References

Daman Arora and Andrea Zanette. 2025. Training language models to reason efficiently.

Rishi Bommasani, Drew Hudson, Ehsan Adeli, Russ Altman, Sanjeev Arora, Sebastian von Arx, Michael S. Bernstein, Jonas Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Jishnu Ray Chowdhury and Cornelia Caragea. 2025. Investigating recurrent transformers with dynamic halt.

Yingqian Cui, Pengfei He, Jingying Zeng, Hui Liu, Xianfeng Tang, Zhenwei Dai, Yan Han, Chen Luo, Jing Huang, Zhen Li, Suhang Wang, Yue Xing, Jiliang Tang, and Qi He. 2025. Stepwise perplexity-guided refinement for efficient chain-of-thought reasoning in large language models.

Yuntian Deng, Yejin Choi, and Stuart Shieber. 2024. From explicit cot to implicit cot: Learning to internalize cot step by step. *arXiv preprint arXiv:2405.14838*.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. 2023. Implicit chain of thought reasoning via knowledge distillation.

Mehdi Fatemi, Banafsheh Rafiee, Mingjie Tang, and Kartik Talamadupula. 2025. Concise reasoning via reinforcement learning.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. 2024. Think before you speak: Training language models with pause tokens.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and

Mao Yang. 2025. rstar-math: Small llms can master math reasoning with self-evolved deep thinking.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space.

Takeshi Kojima, Shixiang Gu, Matthew Reid, and Yutaka Matsuo. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

Xuefeng Li, Haoyang Zou, and Pengfei Liu. 2025. Limr: Less is more for rl scaling.

Elita Lobo, Chirag Agarwal, and Himabindu Lakkaraju. 2025. On the impact of fine-tuning on chain-of-thought reasoning.

Trung Quoc Luong, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. 2024. Reft: Reasoning with reinforced fine-tuning.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Léon Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Rui Wang, Hongru Wang, Boyang Xue, Jianhui Pang, Shudong Liu, Yi Chen, Jiahao Qiu, Derek Fai Wong, Heng Ji, and Kam-Fai Wong. 2025a. Harnessing the reasoning economy: A survey of efficient reasoning for large language models.

Xinyi Wang, Lucas Caccia, Oleksiy Ostapenko, Xingdi Yuan, William Yang Wang, and Alessandro Sordoni. 2024a. Guiding language model reasoning with planning tokens.

Xuezhi Wang et al. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yu Wang, Shiwan Zhao, Zhihu Wang, Heyuan Huang, Ming Fan, Yubo Zhang, Zhixing Wang, Haijun Wang, and Ting Liu. 2024b. Strategic chain-of-thought: Guiding accurate reasoning in llms through strategy elicitation.

Yubo Wang, Xiang Yue, and Wenhu Chen. 2025b. Critique fine-tuning: Learning to critique is more effective than learning to imitate.

Yue Wang, Qiuzhi Liu, Jiahao Xu, Tian Liang, Xingyu Chen, Zhiwei He, Linfeng Song, Dian Yu, Juntao Li, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025c. Thoughts are all over the place: On the underthinking of o1-like llms.

Jason Wei, Maarten Bosma, Douwe Zhao, Kelvin Guu, Aman Yu, Brian Lester, Nikhil Naik, Dian Li, Logan Jones, Kurt Shuster, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. 2024. Do large language models latently perform multi-hop reasoning?

Shunyu Yao, Xi Lin, Yi Li, Denny Zhou, Kam-Fai Wong, and Kai-Wei Chang. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. Limo: Less is more for reasoning.

Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. 2024. Distilling system 2 into system 1.

Qianjin Yu, Keyu Wu, Zihan Chen, Chushu Zhang, Manlin Mei, Lingjun Huang, Fang Tan, Yongsheng Du, Kunlin Liu, and Yurui Zhu. 2025a. Rethinking the generation of high-quality cot data from the perspective of llm-adaptive question difficulty grading.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Weinan Dai, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. 2025b. Dapo: An open-source llm reinforcement learning system at scale.

Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, et al. 2025. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*.

Zhiyuan Zeng, Qinyuan Cheng, Zhangyue Yin, Bo Wang, Shimin Li, Yunhua Zhou, Qipeng Guo, Xuanjing Huang, and Xipeng Qiu. 2024. Scaling of search and learning: A roadmap to reproduce o1 from reinforcement learning perspective.

Yuxiang Zhang, Shangxi Wu, Yuqi Yang, Jiangming Shu, Jinlin Xiao, Chao Kong, and Jitao

Sang. 2024. o1-coder: an o1 replication for coding.

Yu Zhao, Huifeng Yin, Bo Zeng, Hao Wang, Tianqi Shi, Chenyang Lyu, Longyue Wang, Weihua Luo, and Kaifu Zhang. 2024. Marco-o1: Towards open reasoning models for open-ended solutions.