Revisiting Test-Time Scaling: A Survey and a Diversity Aware Method for Efficient Reasoning

Anonymous ACL submission

Abstract

Test-Time Scaling (TTS) improves the reasoning performance of Large Language Models (LLMs) by allocating additional compute during inference. We conduct a structured survey of TTS methods and categorize them into sampling-based, search-based, and trajectory optimization strategies. We observe that reasoning-optimized models often produce less diverse outputs, which limits TTS effectiveness. To address this, we propose **ADAPT** (**A** **D**iversity **A**ware **P**refix fine-**T**uning), a lightweight method that applies prefix tuning with a diversity focused data strategy. Experiments on mathematical reasoning tasks show that **ADAPT** reaches 80% accuracy using eight times less compute than strong baselines. Our findings highlight the essential role of generative diversity in maximizing TTS effectiveness.

1 Introduction

Large Language Models (LLMs) (OpenAI, 2023; Chowdhery et al., 2022; Touvron et al., 2023) have become central to modern NLP applications such as generation, translation, and question answering. Their success largely stems from transformer-based architectures (Vaswani et al., 2017) and large-scale pretraining (Kaplan et al., 2020; Hoffmann et al., 2022), which endow models with strong fluency and generalization. However, standard autoregressive decoding imposes a fixed inference routine that limits their performance on complex reasoning tasks.

As model sizes grow, the training cost escalates, yet the marginal gains diminish. To mitigate this, Test-Time Scaling (TTS) has emerged as a promising direction: it enhances model performance by allocating more compute during inference, allowing adaptation to input complexity without retraining (OpenAI, 2024a; Snell et al., 2024; Welleck et al., 2024).

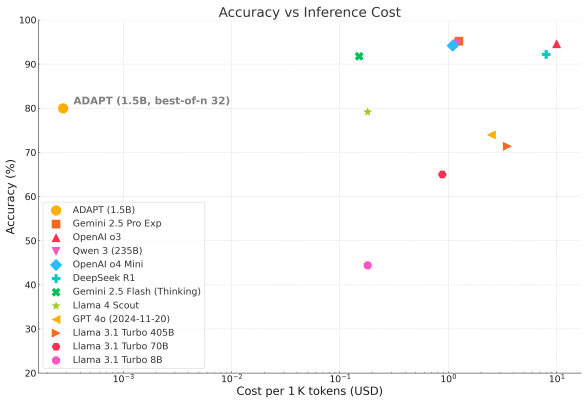

Figure 1: **Accuracy vs. Inference Cost (log scale).** Each point represents a language model.

While TTS has shown effectiveness, its performance is often tied to the model's intrinsic capacity for generation diversity—a factor not yet well understood or explicitly optimized. In particular, models optimized for reasoning, such as distilled variants, tend to exhibit reduced output variance, which may dampen the gains from TTS. This raises an open question: Can diversity-aware fine-tuning improve TTS effectiveness for reasoning models?

To address this, we first conduct a strategy-oriented survey of recent TTS methods, categorizing them into three major families: Sampling (section 3.1), Search (section 3.2), and Trajectory Optimization (section 3.3) and identify diversity as a critical enabler of TTS success. Next, we propose a simple yet effective fine-tuning method, ADAPT (A Diversity Aware Prefix fine-Tuning), which enhances early-stage output diversity via prefix-tuned sampling.

We evaluate ADAPT on a compact reasoning model under Best-of-N sampling. As shown in fig. 1, ADAPT achieves 80% accuracy with eight times fewer samples, outperforming all baseline models in efficiency while retaining strong peak performance.

Contributions. This work makes three key contributions:

- A unified survey of TTS approaches, covering Sampling, Search, and Trajectory Optimization, with a focus on the role of generation diversity.
- The design and evaluation of **ADAPT**, a prefix-tuning method that improves efficiency by increasing diversity at inference.
- A discussion on future directions, including robustness to prompts, synergy between training and inference, hallucination mitigation, safety, and the use of synthetic data for controlled TTS benchmarking.

2 Related Work

Several surveys cover aspects of reasoning, post-training, and test-time scaling. Reasoning-focused overviews include Pan et al. (2025); Xu et al. (2025a), while efficiency-oriented surveys include Feng et al. (2025); Wang et al. (2025b); Sui et al. (2025). Test-time scaling itself is addressed by Zhang et al. (2025); Li (2025), and post-training techniques are reviewed in Kumar et al. (2025b); Tie et al. (2025).

Unlike prior work that focuses primarily on categorization, we go further by connecting our survey insights to a practical hypothesis: that generation diversity moderates TTS effectiveness. We validate this hypothesis through targeted experiments, showing that increasing diversity via prefix tuning leads to more efficient and effective test-time scaling.

3 Test-Time Scaling Survey

In this section, we classify TTS methods into three categories, based on the strategies employed.

3.1 Sampling

Sampling methods draw multiple candidates by adjusting decoding parameters such as temperature. For instance, adjusting the temperature redistributes probabilities to favor rare tokens, encouraging more creative responses. After generating the samples, a verifier grades each response and selects the one with the highest score to be the final response. As shown in Figure 2, the schematic illustrates the basic process of the sampling-based method.

Given the ability to generate diverse responses, numerous previous studies have focused on this

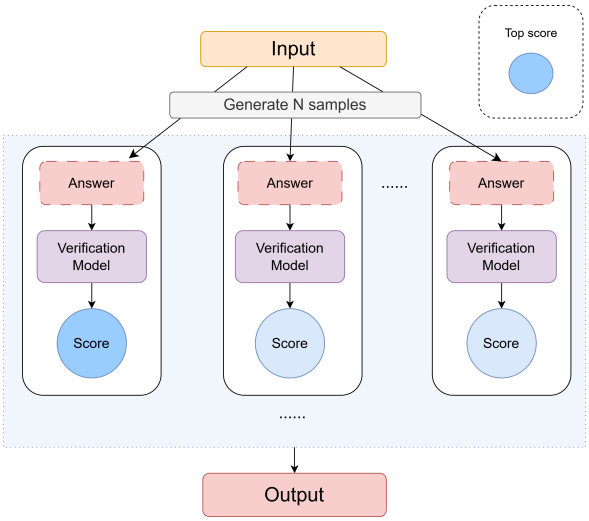

Figure 2: **Sampling-based method.** The model samples N candidates, a verifier scores each, and the highest-scoring answer is returned. Some variants repeat this loop for multiple rounds.

and proposed various strategies to improve performance. For example, Tian et al. (2025) proposed a method that generates answers in multiple rounds. In each round, the model uses the previous answer and the original input as a new input. This approach allows the model to improve its performance without additional training.

In addition, Chow et al. (2024) aim to improve LLM performance under Best-of-N sampling through two approaches. BoN-SFT is a supervised fine-tuning method that maximizes the likelihood of the highest-scoring output among sampled candidates, thereby aligning the model with the Best-of-N policy. In contrast, BoN-RL employs policy-gradient reinforcement learning to directly maximize the expected reward of the selected output under Best-of-N inference.

Furthermore, to address efficiency issues faced by sampling-based strategies, Huang et al. (2025b) proposed a self-calibrated sampling method. They used calibrated confidence scores to enhance the efficiency of answer sampling. The method adjusts sampling through early stopping in Best-of-N and applies confidence-calibrated self-consistency to reduce computation.

The limitations of sampling. Although sampling improves performance, Stroebl et al. (2024) noted that accuracy gains plateau as verifiers become unreliable with more samples. Huang et al. (2025a) found that Best-of-N can suffer from reward hacking, especially in low-diversity models

where sampled responses are redundant. They introduced pessimistic rejection sampling to filter out unreliable outputs, a limitation especially pronounced in distilled models.

3.2 Search

Search-based methods focus on searching for diverse paths. These paths help improve answer quality. Some search in the latent space. This provides an alternative to token-level reasoning. Others adopt the self-improvement method to enhance their performance.

3.2.1 Variations of CoT

The origins of CoT stem from prompt-based methods (Lightman et al., 2024; Ranaldi et al., 2025), such as "Let's think step by step." Several improved CoT variants are shown in fig. 3.

CoT with Self-Consistency (CoT-SC) (Wang et al., 2023) samples multiple reasoning paths. It then selects the one that best meets a consistency criterion. This improves performance but increases computation. Auto-CoT (Zhang et al., 2022) reduces the effect of incorrect demonstrations in the thought process. Monte Carlo Tree Search (MCTS) (Xie et al., 2024) updates the model policy through Direct Preference Optimization (DPO) to manage the trade-off between training and inference.

Similarly, the Tree-of-Thoughts (ToT) Yao et al. (2023) enables exploration of multiple reasoning paths per step and selects the best action. Building on multi-path exploration, Forest-of-Thought (FoT) (Bi et al., 2025) runs multiple reasoning trees in parallel instead of a single one. Each tree independently explores a possible solution path, and dynamically compares, discards, or merges ideas across trees. This process is parallel and selective. It improves diversity, adaptability, and resilience. It also avoids local traps found in single-path methods like CoT and ToT. Graph of Thoughts (GoT) (Besta et al., 2024) represents reasoning as a graph. Nodes are thoughts. Edges are logical links. This structure supports revisiting, merging, and reusing ideas. It enables flexible reasoning and improves coherence and depth in complex tasks.

In addition to structural innovations, Atom of Thoughts (AoT) (Teng et al., 2025) breaks down complex tasks into self-contained sub-questions. This reduces reliance on historical context and improves generalization. On the efficiency side, Chain of Draft (Xu et al., 2025b) constrains the length of the thinking trace, improving both speed and output quality. Finally, the Mixture-of-Agents architecture (Wang et al., 2024a) leverages multiple LLMs with specialized capabilities to collaborate on complex reasoning tasks, further broadening the design space of CoT-style reasoning.

Theory of CoT. While still limited, recent work begins to formalize CoT's reasoning process beyond empirical improvements. One approach explores the expressive power of CoT, as studied in Liu et al. (2023); Merrill and Sabharwal (2024); Li et al. (2024); Feng et al. (2023). Other work focuses on the connection between CoT and in-context learning Huang et al. (2025c), as both improve performance without parameter updates.

3.2.2 Hidden layer search

Hidden layer search offers new opportunities for efficiency and abstraction. Sleep-time compute method (Lin et al., 2025) processes the input context offline. It generates a latent representation during this stage. The model reuses this representation at inference time. This method avoids repeating full context processing for each query.

The Coconut paradigm (Chain of Continuous Thought) (Hao et al., 2024) enables models to reason in a continuous latent space by feeding the last hidden state back into the model instead of decoding it into tokens. This method allows backtracking and enables reducing token usage significantly. Latent-Thought Language Models (LTMs) (Kong et al., 2025) guides decoding through latent thought vectors inferred using variational Bayesian techniques.

Another line of work, CODI (Shen et al., 2025), introduces a framework that compresses CoT reasoning into a continuous latent space via self-distillation. Looped transformer (Saunshi et al., 2025) presents another hidden-layer based strategy by repeatedly applying a smaller transformer module across iterations. This looping mechanism enables smaller models to match the reasoning capabilities of larger ones on complex reasoning tasks.

3.2.3 Self-improvement

TTS often leverages implicit self-improvement through refinement and correction. These approaches draw inspiration from human iterative thinking, enabling models to refine their answers dynamically by leveraging internal feedback signals and reward-guided strategies.

Self-Refine (Madaan et al., 2023) introduces a simple, training-free framework in which LLMs

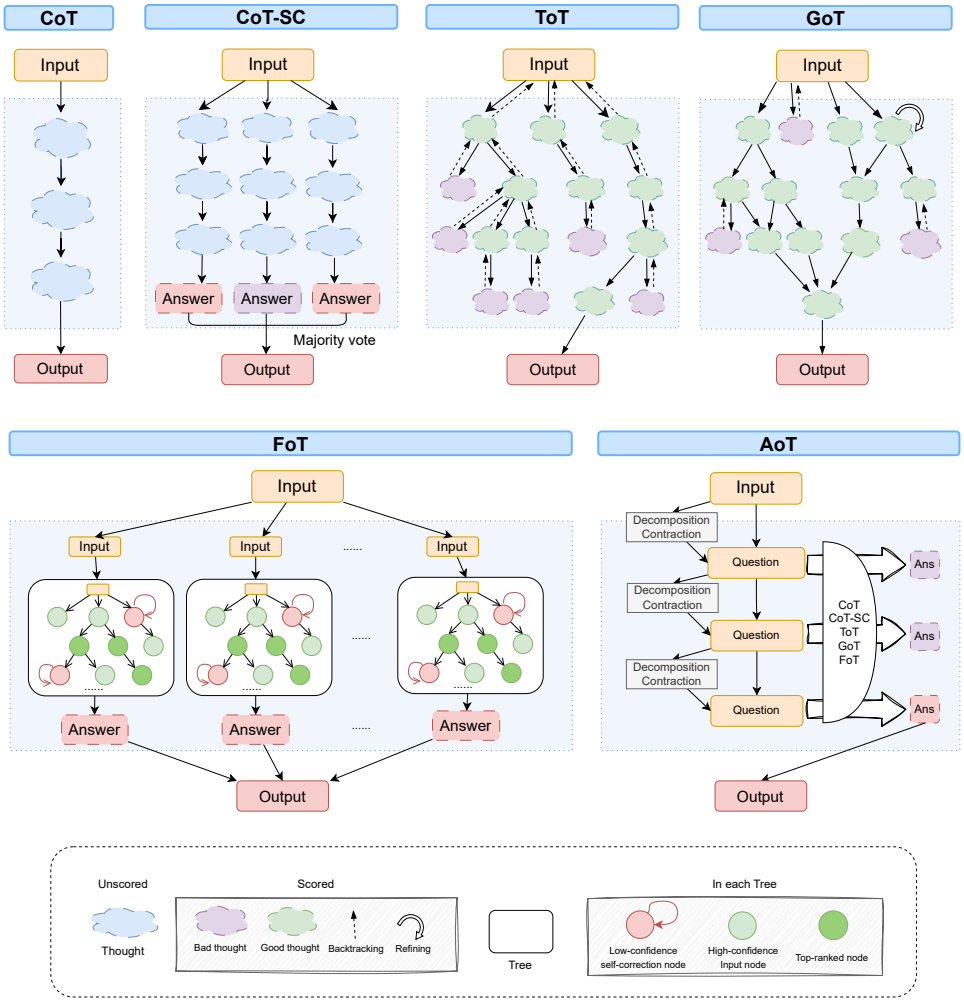

Figure 3: **Search-Based** The schematic illustrates various search-based methods, including Chain of Thought (CoT), Self-Consistency with Chain of Thought (CoT-SC), Tree of Thoughts (ToT), Graph of Thoughts (GoT), Forest of Thoughts (FoT), and Atom of Thoughts (AoT).

generate self-criticism and revise their responses accordingly. Building on this, STaR (Zelikman et al., 2022) bootstraps high-quality rationales that lead to correct answers, using them to fine-tune the model essentially allowing LLMs to "teach themselves" how to reason more effectively.

Expanding further, Training LLMs to Self-Correct via RL (Kumar et al., 2025a) introduce reward-driven strategies. It proposes a reinforcement-based feedback loop to train LLMs to learn how to revise, achieving significant gains in reasoning tasks. Recursive Introspection (RISE) (Qu et al., 2024), on the other hand, formalize multi-turn refinement as a Markov Decision Process (MDP) to iteratively optimize outputs through sequential edits. After RL training, RISE significantly outperforms Self-Refine and parallel sampling, highlighting the benefits of trajectory-aware self-correction. Feedback-based Test-Time Training (FTTT) (Li et al., 2025) takes self-improvement a step further by applying gradient-based updates during inference. Treating reasoning as a local optimization problem, FTTT adapts model weights based on feedback using a learned optimizer (OPTUNE). This method achieves robust performance.

3.3 Trajectory Optimization

Optimization of TTS in LLMs focuses on controlling and refining the reasoning trajectory. This refers to the sequence of intermediate steps generated during inference. This manifests as adjusting trajectory length or structure based on task difficulty, balancing accuracy with computational cost. Unlike static decoding, optimized test-time reasoning introduces adaptive mechanisms to improve outcomes under fixed compute budgets.

This optimization addresses two critical challenges. First, Yang et al. (2025) demonstrated that

longer reasoning trajectories don't uniformly improve results; excessive steps often degrade accuracy through error accumulation. Second, compute allocation at test time can be more effective than scaling model parameters (Snell et al., 2024), suggesting trajectory optimization offers a compute-efficient path to enhanced performance.

3.3.1 RL Paradigms

We categorize current methods into two paradigms: reinforcement learning (RL) and non-RL approaches. RL enables optimal test-time compute scaling by aligning model outputs with rewards. Setlur et al. (2025); Qu et al. (2025) showed that RL with step-wise feedback or meta-reinforcement fine-tuning (MRT) significantly improves sampling efficiency and test-time performance. MRT frames inference as a meta-RL problem, rewarding useful steps toward correct answers to encourage concise reasoning within token limits.

However, recent critiques question RL's necessity and sufficiency. Yue et al. (2025) showed that RL mainly reweights the base model's outputs, boosting performance at low N but lagging at larger N where exploration matters. Their findings suggest RL narrows the reasoning scope, potentially suppressing valuable but infrequent trajectories. In contrast, distillation-based methods inject new knowledge into student models, expanding reasoning capacity across all N values. Cheng et al. (2025) identified reward hacking in RL with process reward models (PRMs) and propose a framework with min-form credit assignment, achieving high accuracy using only 30% of the reasoning steps.

3.3.2 Distillation Paradigms

Distillation-based methods offer a data-driven alternative to RL for trajectory optimization. Rather than relying on reward shaping, distillation directly transfers structured reasoning from large teacher models to smaller student models.

Recent work (Shridhar et al., 2023; Hsieh et al., 2023; Chen et al., 2025b; Ma et al., 2025a) showed that teacher-guided stepwise supervision enables concise, effective reasoning without relying on long inference chains. The teacher explores diverse reasoning, such as multiple paths and tree-structured CoTs. This is distilled into the student to produce accurate answers with shorter trajectories. Distillation improves by adjusting the supervision format (Chen et al., 2025b) or curating datasets that balance trajectory length and informativeness (Yin et al., 2025). Moreover, distilled models often match or surpass larger models within similar compute budgets (Shridhar et al., 2023; Hsieh et al., 2023) and frequently generalize better than RL-trained models, which may overfit to narrow reward signals (Yue et al., 2025; Cheng et al., 2025). Hybrid strategies combining distillation and RL show promise for robust reasoning (Liu et al., 2025), though this remains an open research area.

Several methods optimize trajectory length across paradigms. Z1 (Yu et al., 2025) trains on paired long-short solutions to enable adaptive generation. MRT (Qu et al., 2025) uses episodic rewards to encourage early termination. PURE assigns credit to penalize low-quality steps, reducing verbosity and error-prone reasoning.

3.4 Challenges and Summary

Our preceding survey categorizes TTS approaches into three primary dimensions: parallel scaling, sequential scaling, and computational optimization. A key insight that emerges is the importance of a model's *inherent generative diversity* in determining TTS effectiveness. By increasing test-time sampling, models can explore a broader range of reasoning trajectories, thereby increasing the likelihood of arriving at a correct solution.

This observation, however, raises a critical question for models specifically optimized for reasoning. Prior work on model specialization (section 3.3) suggests that targeted training for specific capabilities may narrow a model's output distribution. Motivated by this, we examine how such specialization affects TTS performance.

We hypothesize that although reasoning-optimized models are skilled at producing accurate responses, this specialization may inadvertently reduce the output diversity needed for effective test-time scaling. This potential trade-off between reasoning precision and generative flexibility serves as the foundation for our experimental investigation.

4 ADAPT: A Diversity Aware Prefix Fine-Tuning Method

Motivated by the hypothesis that generative diversity influences TTS performance, we propose a simple yet effective method to encourage diversity and empirically test whether increased diversity leads to improved TTS results. Our method, **ADAPT** (**A** **D**iversity-**A**ware **P**refix fine-**T**uning),

explicitly promotes generation diversity through an efficient prefix fine-tuning strategy. The key idea is to fine-tune only the initial segments of reasoning trajectories using a carefully curated data mixture, thereby encouraging the model to explore a broader range of initial reasoning paths. This enhanced diversity is expected to enable TTS methods such as best-of-N sampling to identify correct solutions more efficiently, requiring fewer candidates than less diverse models.

4.1 Dataset Curation

The training dataset consists primarily of *diverse* responses, supplemented with a smaller subset of outputs generated by the target model, which may exhibit lower generative diversity. This latter subset is included to mitigate potential catastrophic forgetting and to preserve the model's original capabilities.

In our experiments, the dataset includes 90% responses generated by `Qwen2.5-Math-1.5B` and 10% inference outputs from `DeepSeek-R1-Distill-Qwen-1.5B` (our target model). For the Qwen-derived examples, we employ a custom prompt format designed to encourage varied initial reasoning steps (see appendix A.5 for details), whereas the DeepSeek-generated samples retain their original chat template. Since all training targets are produced by existing models, this fine-tuning process can be viewed as a form of targeted knowledge transfer or self-supervised learning.

4.2 Fine-Tuning Procedure

To improve efficiency and focus on the early stages of reasoning, we truncate all training instances to their first 512 tokens and fine-tune the model using a supervised learning objective. Gradient updates are applied only to the prefix segments, while the remainder of the model remains frozen, preserving most of the pre-trained parameters.

5 Experiments

In this section, we detail the experimental setup designed to evaluate our proposed method, **ADAPT**, and to investigate the interplay between solution diversity, trajectory length, and the performance of TTS for reasoning tasks. We first describe the datasets, evaluation protocols, and baseline models. We then present a comparative analysis, focusing on accuracy and computational efficiency.

5.1 Experimental Setup

Tasks and Datasets. Our experiments are conducted on challenging mathematical reasoning benchmarks, akin to MATH-500 (Lightman et al., 2023). The goal is to assess the models' ability to generate correct multi-step reasoning paths.

Evaluation Protocol. Inspired by the test-time compute setup highlighted by Beeching et al., we employ a Best-of-N sampling strategy for all models. For each problem, we generate N candidate solutions, where $N \in \{2, 4, 8, 16, 32, 64, 128, 256\}$. Unless otherwise specified, solutions are generated with a temperature of 0.8 and a maximum length of 2048 tokens per problem. The final answer is determined by majority voting over the N candidates, and we report the accuracy based on this aggregated answer (`acc_maj`).

Metrics. We evaluate model performance using four metrics. **acc_maj** denotes the final accuracy obtained via majority voting over N sampled outputs. **Improvement** measures the absolute increase in `acc_maj` relative to the baseline performance at $N = 2$. **Gain per generation** quantifies the average accuracy gain when doubling the sample size (e.g., from $N = 2$ to $N = 4$). Finally, **Min N to hit threshold** refers to the smallest sample count N required to reach a target `acc_maj`, such as 80%.

5.2 Models

Three models are compared in our study:

- **Qwen-1.5B**: A pre-trained language model tailored for mathematical reasoning, likely to exhibit higher generative diversity. (`Qwen2.5-Math-1.5B`)
- **DeepSeek-Qwen-1.5B**: A distilled variant optimized for reasoning tasks (DeepSeek-AI, 2025), which may exhibit reduced generative diversity. This model serves as the target for our ADAPT method. (`DeepSeek-R1-Distill-Qwen-1.5B`)
- **ADAPT**: A diversity aware prefix fine-tuning approach applied to **DeepSeek-Qwen-1.5B**, aimed at enhancing its generative diversity while preserving reasoning accuracy.

5.3 Diversity Impact on TTS Performance

We begin by examining how the effectiveness of best-of-N sampling in TTS depends on both solution diversity and trajectory length. We compare a pre-trained model (`Qwen2.5-Math-1.5B`) with

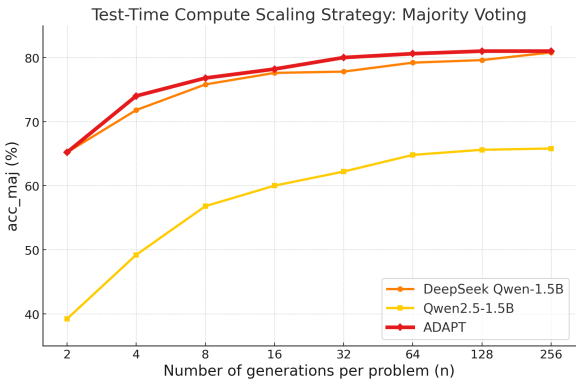

Figure 4: **TTS performance comparison of pre-trained vs. distilled model.**

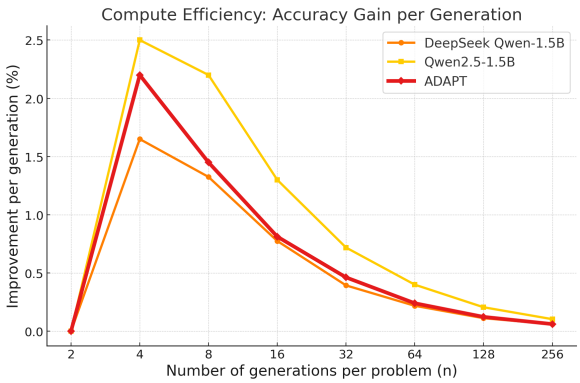

Figure 5: **Marginal gain per generation for across models**

its distilled counterpart optimized for reasoning (`DeepSeek Qwen-1.5B`).

As shown in Figure 4, `DeepSeek Qwen-1.5B` achieves a higher baseline accuracy of 65.2% at $N = 2$, reaching 80.8% at $N = 256$ (+15.6%). In contrast, `Qwen2.5-Math-1.5B` starts lower at 39.2% but exhibits a steeper improvement, reaching 65.8% (+26.6%). These findings suggest that while distillation enhances baseline reasoning performance, it may also suppress generative diversity, limiting the benefits of increased sampling. This observation well motivated our hypothesis: explicitly promoting diversity during fine-tuning can enhance both the efficiency and effectiveness of TTS for reasoning-specialized models.

5.4 Results

Table 1 summarizes the results of **ADAPT** compared to both baselines. Notably, **ADAPT** reaches 80% accuracy with only 32 samples, an 8× improvement over `DeepSeek Qwen-1.5B`. At $N = 16$, it already surpasses the distilled baseline at $N = 256$.

As shown in Table 1, both **ADAPT** and `DeepSeek Qwen-1.5B` achieve strong initial performance (65.2% at $N = 2$), clearly outperforming the pre-trained `Qwen2.5-1.5B` (39.2%). **ADAPT** attains the highest peak accuracy (81.0% at $N = 256$), marginally surpassing `DeepSeek Qwen-1.5B` (80.8%). While `Qwen2.5-1.5B` shows the largest relative improvement (+26.6 pp), its ceiling remains substantially lower.

Beyond peak accuracy, **ADAPT** offers significant advantages in convergence speed and sampling efficiency. It reaches 80% accuracy with just $N = 32$ samples—an 8× improvement over the $N = 256$ needed by `DeepSeek Qwen-1.5B`. With only $N = 16$ samples, **ADAPT** already achieves 78.2%, making it suitable for low-budget inference. In contrast, `Qwen2.5-1.5B` fails to reach the 80% threshold even at $N = 256$.

Figure 5 illustrates the diminishing returns of larger N for all models. For **ADAPT**, most gains occur early ($N = 2$ to $N = 32$), suggesting that it captures the majority of TTS benefits with a relatively small number of generations.

5.5 Discussion

Our results validate the central hypothesis: enhancing output diversity improves both the accuracy and efficiency of reasoning models under Best-of-N sampling. While `DeepSeek Qwen-1.5B` offers a strong baseline, its improvements require large N, consistent with prior observations that distillation, though beneficial for core reasoning, may suppress generative diversity.

ADAPT addresses this limitation via prefix fine-tuning focused on early-stage reasoning. By updating only a small number of prefix parameters and training on a hybrid dataset—composed of model-generated responses and a subset of base model outputs—ADAPT promotes diverse initial reasoning paths while mitigating catastrophic forgetting. The prompt template further steers variation in the early solution space (see appendix A.5).

This strategy yields strong gains even under limited sampling budgets: **ADAPT** exceeds 80% accuracy with only $N = 32$ samples and achieves 78.2% at $N = 16$. These results demonstrate that diversity-aware prefix tuning enables more efficient exploitation of TTS compared to standard fine-tuned models.

In summary, ADAPT delivers faster convergence, stronger low-sample performance, and bet-

Model	Acc. Maj@2	Acc. Maj@256	Acc. Maj@16	Min N for 80% (↓)
Qwen2.5-1.5B	39.2%	65.8%	60.0%	∞
DeepSeek Qwen-1.5B	65.2%	80.8%	77.6%	256
ADAPT (Ours)	**65.2%**	**81.0%**	**78.2%**	**32**

Table 1: **Best-of-N majority voting results.** "Min N" indicates the smallest N required to reach 80% accuracy.

ter overall accuracy under Best-of-N majority voting. Its improvements stem from explicitly optimizing for sample-efficient diversity, making it a compelling approach for resource-constrained inference.

6 Future Directions

In this section, we outline several future directions in the TTS regime. As a new research area, TTS still faces many unexplored challenges and developed applications. A detailed discussion of TTS applications is provided in appendix A. Here, we focus on five key categories: robustness, training, safety, hallucination, and synthetic datasets.

Robustness. TTS might fail under certain conditions. For instance, prompt format significantly affect LLM performance on the same task. Hochlehnert et al. (2025) showed that reasoning accuracy is highly sensitive to prompt design. Therefore, investigating how the content and structure of prompts affect TTS is a promising direction for future research.

Training. Dang et al. (2025) identified that the training of the reasoning models leads to suboptimal performance of TTS. Furthermore, Chen et al. (2025a) suggested that modifications to the pre-training / fine-tuning stages are necessary to optimize TTS performance.

Specifically, they observe that training with cross-entropy loss leads to a decrease in pass@N accuracy. It may result from cross-entropy loss leading to model overconfidence. The finding underscores the need to reconsider the pre-training and fine-tuning strategies in order to achieve better performance and efficiency. For instance, it is important to explore the impact of training strategies, such as backpropagation, on TTS performance.

Safety. Chen et al. found that the thinking process of LLMs may not be entirely transparent. They prompt the model with a question and the corresponding answer, but observe that LLMs rarely acknowledge receiving the hint. Additionally, they note that CoT monitoring may not be sufficiently reliable to capture the true cognitive process of LLMs. Future work should further explore how LLMs perform reasoning to derive answers, in order to establish better control over their behavior.

Hallucination. Despite TTS success on some math and coding tests, OpenAI (2024b) showed that GPT o3 and o4-mini suffer from the hallucination problem. Future work should aim to understand the relationship between RL and SFT methods used to enable LLMs to scale test-time computation, as well as their impact on hallucination.

Synthetic dataset. Synthetic datasets (Goldie et al., 2025; Wang et al., 2025a; Lei et al., 2024) provide precise control over task structure and difficulty. It enables the isolation of specific factors relevant to test-time computation. This reduces confounding effects present in natural data and helps reveal how scaling the inference budget impacts reasoning depth, compositionality, and context integration.

7 Conclusion

In this paper, we present a comprehensive survey of test-time scaling (TTS), categorizing recent approaches into three main strategies: sampling, search, and trajectory optimization. Through this analysis, we hypothesize that *generative diversity* is a key factor influencing TTS performance. Our experiments support this hypothesis, showing that while distillation enhances baseline reasoning accuracy, it can also reduce output diversity, thereby limiting the effectiveness of sampling-based TTS methods. To address this limitation, we introduce **ADAPT**, a diversity aware prefix fine-tuning method designed to enhance output diversity and improve both the efficiency and performance of reasoning-optimized models under TTS.

Limitations

While **ADAPT** demonstrates strong performance under Best-of-N sampling, several limitations re-

main. First, all experiments are conducted on a single reasoning domain—mathematical problem solving. It remains unclear whether similar diversity-induced gains would generalize to broader tasks such as commonsense or multi-hop QA. Second, our evaluations focus on a relatively small model (1.5B parameters); scaling effects and interactions with larger architectures are left for future work.

Third, although ADAPT improves sample efficiency, it does not directly optimize diversity metrics (e.g., self-BLEU, pairwise entropy), and its diversity-enhancing effect is inferred only through indirect accuracy gains. Explicit diversity measurements could provide more rigorous support for the core hypothesis. Finally, we fix the prefix length and data mixture ratio throughout; exploring how these hyperparameters impact diversity and performance may yield further improvements.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program synthesis with large language models. *Preprint*, arXiv:2108.07732.

Edward Beeching, Lewis Tunstall, and Sasha Rush. Scaling test-time compute with open models.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024. Graph of thoughts: Solving elaborate problems with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):17682–17690.

Zhenni Bi, Kai Han, Chuanjian Liu, Yehui Tang, and Yunhe Wang. 2025. Forest-of-thought: Scaling test-time compute for enhancing llm reasoning. *Preprint*, arXiv:2412.09078.

Houda Bouamor, Juan Pino, and Kalika Bali, editors. 2023. *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Singapore.

Feng Chen, Allan Raventos, Nan Cheng, Surya Ganguli, and Shaul Druckmann. 2025a. Rethinking fine-tuning when scaling test-time compute: Limiting confidence improves mathematical reasoning. *Preprint*, arXiv:2502.07154.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Xinghao Chen, Zhijing Sun, Wenjin Guo, Miaoran Zhang, Yanjun Chen, Yirong Sun, Hui Su, Yijie Pan, Dietrich Klakow, Wenjie Li, and Xiaoyu Shen. 2025b. Unveiling the key factors for distilling chain-of-thought reasoning. *Preprint*, arXiv:2502.18001.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner Fabien Roger Vlad Mikulik, Sam Bowman, Jan Leike Jared Kaplan, and 1 others. Reasoning models don't always say what they think.

Jie Cheng, Ruixi Qiao, Lijun Li, Chao Guo, Junle Wang, Gang Xiong, Yisheng Lv, and Fei-Yue Wang. 2025. Stop summation: Min-form credit assignment is all process reward model needs for reasoning. *Preprint*, arXiv:2504.15275.

Yinlam Chow, Guy Tennenholtz, Izzeddin Gur, Vincent Zhuang, Bo Dai, Sridhar Thiagarajan, Craig Boutilier, Rishabh Agarwal, Aviral Kumar, and Aleksandra Faust. 2024. Inference-aware fine-tuning for best-of-n sampling in large language models. *Preprint*, arXiv:2412.15287.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, and 1 others. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv:1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Karan Dalal, Daniel Koceja, Gashon Hussein, Jiarui Xu, Yue Zhao, Youjin Song, Shihao Han, Ka Chun Cheung, Jan Kautz, Carlos Guestrin, Tatsunori Hashimoto, Sanmi Koyejo, Yejin Choi, Yu Sun, and Xiaolong Wang. 2025. One-minute video generation with test-time training. *Preprint*, arXiv:2504.05298.

Xingyu Dang, Christina Baek, Kaiyue Wen, Zico Kolter, and Aditi Raghunathan. 2025. Weight ensembling improves reasoning in language models. *Preprint*, arXiv:2504.10478.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. 2023. Towards revealing the mystery behind chain of thought: A theoretical perspective. *Preprint*, arXiv:2305.15408.

Sicheng Feng, Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2025. Efficient reasoning models: A survey. *Preprint*, arXiv:2504.10903.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. 2024. Omni-math: A universal olympiad level mathematic benchmark for large language models. *Preprint*, arXiv:2410.07985.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. *Preprint*, arXiv:2211.10435.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Anna Goldie, Azalia Mirhoseini, Hao Zhou, Irene Cai, and Christopher D. Manning. 2025. Synthetic data generation & multi-step rl for reasoning & tool use. *Preprint*, arXiv:2504.04736.

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. *Preprint*, arXiv:2412.06769.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *Preprint*, arXiv:2402.14008.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. 2021a. Measuring coding challenge competence with apps. *NeurIPS*.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2021b. Aligning ai with shared human values. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021c. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021d. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Andreas Hochlehnert, Hardik Bhatnagar, Vishaal Udandarao, Samuel Albanie, Ameya Prabhu, and Matthias Bethge. 2025. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. *Preprint*, arXiv:2504.07086.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *Preprint*, arXiv:2305.02301.

Audrey Huang, Adam Block, Qinghua Liu, Nan Jiang, Akshay Krishnamurthy, and Dylan J. Foster. 2025a. Is best-of-n the best of them? coverage, scaling, and optimality in inference-time alignment. *Preprint*, arXiv:2503.21878.

Chengsong Huang, Langlin Huang, Jixuan Leng, Jiacheng Liu, and Jiaxin Huang. 2025b. Efficient test-time scaling via self-calibration. *Preprint*, arXiv:2503.00031.

Jianhao Huang, Zixuan Wang, and Jason D. Lee. 2025c. Transformers learn to implement multi-step gradient descent with chain of thought. *Preprint*, arXiv:2502.21212.

Xiaoke Huang, Juncheng Wu, Hui Liu, Xianfeng Tang, and Yuyin Zhou. 2025d. m1: Unleash the potential of test-time scaling for medical reasoning with large language models. *Preprint*, arXiv:2504.00869.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *Preprint*, arXiv:2403.07974.

Can Jin, Hongwu Peng, Qixin Zhang, Yujin Tang, Dimitris N. Metaxas, and Tong Che. 2025. Two heads are better than one: Test-time scaling of multi-agent collaborative reasoning. *Preprint*, arXiv:2504.09772.

Manuj Kant, Manav Kant, Marzieh Nabi, Preston Carlson, and Megan Ma. 2024. Equitable access to justice: Logical llms show promise. *Preprint*, arXiv:2410.09904.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1152–1157, San Diego, California. Association for Computational Linguistics.

Deqian Kong, Minglu Zhao, Dehong Xu, Bo Pang, Shu Wang, Edouardo Honig, Zhangzhang Si, Chuan Li, Jianwen Xie, Sirui Xie, and Ying Nian Wu. 2025. Scalable language models with posterior inference of latent thought vectors. *Preprint*, arXiv:2502.01567.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. 2025a. Training language models to self-correct via reinforcement learning. In *The Thirteenth International Conference on Learning Representations*.

Komal Kumar, Tajamul Ashraf, Omkar Thawakar, Rao Muhammad Anwer, Hisham Cholakkal, Mubarak Shah, Ming-Hsuan Yang, Phillip H. S. Torr, Fahad Shahbaz Khan, and Salman Khan. 2025b. Llm post-training: A deep dive into reasoning large language models. *Preprint*, arXiv:2502.21321.

Fangyu Lei, Qian Liu, Yiming Huang, Shizhu He, Jun Zhao, and Kang Liu. 2024. S3eval: A synthetic, scalable, systematic evaluation suite for large language models. *Preprint*, arXiv:2310.15147.

Xinzhe Li. 2025. A survey on LLM test-time compute via search: Tasks, LLM profiling, search algorithms, and relevant frameworks. *Transactions on Machine Learning Research*.

Yanyang Li, Michael Lyu, and Liwei Wang. 2025. Learning to reason from feedback at test-time.

Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. 2024. Chain of thought empowers transformers to solve inherently serial problems. *Preprint*, arXiv:2402.12875.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Kevin Lin, Charlie Snell, Yu Wang, Charles Packer, Sarah Wooders, Ion Stoica, and Joseph E. Gonzalez. 2025. Sleep-time compute: Beyond inference scaling at test-time. *Preprint*, arXiv:2504.13171.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.

Bingbin Liu, Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. 2023. Transformers learn shortcuts to automata. *Preprint*, arXiv:2210.10749.

Guanlin Liu, Anand Ramachandran, Tanmay Gangwani, Yan Fu, and Abhinav Sethy. 2025. Knowledge distillation with training wheels. *Preprint*, arXiv:2502.17717.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. 2025. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. `https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-S` Notion Blog.

Junyu Ma, Tianqing Fang, Zhisong Zhang, Hongming Zhang, Haitao Mi, and Dong Yu. 2025a. Recall with reasoning: Chain-of-thought distillation for mamba's long-context memory and extrapolation. *Preprint*, arXiv:2505.03320.

Xiao Ma, Yuhui Tao, Yuhan Zhang, Zexuan Ji, Yizhe Zhang, and Qiang Chen. 2024. Test-time generative augmentation for medical image segmentation. *Preprint*, arXiv:2406.17608.

Yingwei Ma, Yongbin Li, Yihong Dong, Xue Jiang, Rongyu Cao, Jue Chen, Fei Huang, and Binhua Li. 2025b. Thinking longer, not larger: Enhancing software engineering agents via scaling test-time compute. *Preprint*, arXiv:2503.23803.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback.

William Merrill and Ashish Sabharwal. 2024. The expressive power of transformers with chain of thought. *Preprint*, arXiv:2310.07923.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2024. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *Preprint*, arXiv:2410.05229.

NVIDIA, :, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, and 60 others. 2025. Cosmos world foundation model platform for physical ai. *Preprint*, arXiv:2501.03575.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

OpenAI. 2024a. Learning to reason with llms. Accessed: 2025-05-18.

OpenAI. 2024b. Openai o3 and o4-mini system card. Technical report, OpenAI. Accessed: 2025-04-22.

Qianjun Pan, Wenkai Ji, Yuyang Ding, Junsong Li, Shilian Chen, Junyi Wang, Jie Zhou, Qin Chen, Min Zhang, Yulan Wu, and Liang He. 2025. A survey of slow thinking-based reasoning llms using reinforced learning and inference-time scaling law. *Preprint*, arXiv:2505.02665.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Yuxiao Qu, Matthew Y. R. Yang, Amrith Setlur, Lewis Tunstall, Edward Emanuel Beeching, Ruslan Salakhutdinov, and Aviral Kumar. 2025. Optimizing test-time compute via meta reinforcement fine-tuning. *Preprint*, arXiv:2503.07572.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. 2024. Recursive introspection: Teaching language model agents how to self-improve.

Leonardo Ranaldi, Marco Valentino, Alexander Polonsky, and Andrè Freitas. 2025. Improving chain-of-thought reasoning via quasi-symbolic abstractions. *Preprint*, arXiv:2502.12616.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Abulhair Saparov and He He. 2023. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *The Eleventh International Conference on Learning Representations*.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. 2025. Reasoning with latent thoughts: On the power of looped transformers. *Preprint*, arXiv:2502.17416.

Amrith Setlur, Nived Rajaraman, Sergey Levine, and Aviral Kumar. 2025. Scaling test-time compute without verification or rl is suboptimal. *Preprint*, arXiv:2502.12118.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. 2025. Codi: Compressing chain-of-thought into continuous space via self-distillation. *Preprint*, arXiv:2502.21074.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. *Preprint*, arXiv:2212.00193.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *Preprint*, arXiv:2408.03314.

Benedikt Stroebl, Sayash Kapoor, and Arvind Narayanan. 2024. Inference scaling flaws: The limits of llm resampling with imperfect verifiers. *Preprint*, arXiv:2411.17501.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, and Xia Hu. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *Preprint*, arXiv:2503.16419.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Fengwei Teng, Zhaoyang Yu, Quan Shi, Jiayi Zhang, Chenglin Wu, and Yuyu Luo. 2025. Atom of thoughts for markov llm test-time scaling. *Preprint*, arXiv:2502.12018.

Xiaoyu Tian, Sitong Zhao, Haotian Wang, Shuaiting Chen, Yunjie Ji, Yiping Peng, Han Zhao, and Xiangang Li. 2025. Think twice: Enhancing llm reasoning by scaling multi-round test-time thinking. *Preprint*, arXiv:2503.19855.

Guiyao Tie, Zeli Zhao, Dingjie Song, Fuyang Wei, Rong Zhou, Yurou Dai, Wen Yin, Zhejian Yang, Jiangyue Yan, Yao Su, Zhenhan Dai, Yifeng Xie, Yihan Cao, Lichao Sun, Pan Zhou, Lifang He, Hechang Chen, Yu Zhang, Qingsong Wen, and 7 others. 2025. A survey on post-training of large language models. *Preprint*, arXiv:2503.06072.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Jiankang Wang, Jianjun Xu, Xiaorui Wang, Yuxin Wang, Mengting Xing, Shancheng Fang, Zhineng Chen, Hongtao Xie, and Yongdong Zhang. 2025a. A graph-based synthetic data pipeline for scaling high-quality reasoning instructions. *Preprint*, arXiv:2412.08864.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. 2024a. Mixture-of-agents enhances large language model capabilities. *Preprint*, arXiv:2406.04692.

Rui Wang, Hongru Wang, Boyang Xue, Jianhui Pang, Shudong Liu, Yi Chen, Jiahao Qiu, Derek Fai Wong, Heng Ji, and Kam-Fai Wong. 2025b. Harnessing the reasoning economy: A survey of efficient reasoning for large language models. *Preprint*, arXiv:2503.24377.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Preprint*, arXiv:2406.01574.

Sean Welleck, Amanda Bertsch, Matthew Finlayson, Hailey Schoelkopf, Alex Xie, Graham Neubig, Ilia Kulikov, and Zaid Harchaoui. 2024. From decoding to meta-generation: Inference-time algorithms for large language models. *Preprint*, arXiv:2406.16838.

Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P. Lillicrap, Kenji Kawaguchi, and Michael Shieh. 2024. Monte carlo tree search boosts reasoning via iterative preference learning. *Preprint*, arXiv:2405.00451.

Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, Chenyang Shao, Yuwei Yan, Qinglong Yang, Yiwen Song, Sijian Ren, Xinyuan Hu, Yu Li, Jie Feng, Chen Gao, and Yong Li. 2025a. Towards large reasoning models: A survey of reinforced reasoning with large language models. *Preprint*, arXiv:2501.09686.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025b. Chain of draft: Thinking faster by writing less. *Preprint*, arXiv:2502.18600.

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. 2025. Towards thinking-optimal scaling of test-time compute for llm reasoning. *Preprint*, arXiv:2502.18080.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Preprint*, arXiv:2305.10601.

Huifeng Yin, Yu Zhao, Minghao Wu, Xuanfan Ni, Bo Zeng, Hao Wang, Tianqi Shi, Liangying Shao, Chenyang Lyu, Longyue Wang, Weihua Luo, and Kaifu Zhang. 2025. Towards widening the distillation bottleneck for reasoning models. *Preprint*, arXiv:2503.01461.

Zhaojian Yu, Yinghao Wu, Yilun Zhao, Arman Cohan, and Xiao-Ping Zhang. 2025. Z1: Efficient test-time scaling with code. *Preprint*, arXiv:2504.00810.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *Preprint*, arXiv:2504.13837.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. 2025. Robotic control via embodied chain-of-thought reasoning. *Preprint*, arXiv:2407.08693.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Wenyue Hua, Haolun Wu, Zhihan Guo, Yufei Wang, Niklas Muennighoff, Irwin King, Xue Liu, and Chen Ma. 2025. A survey on test-time scaling in large language models: What, how, where, and how well? *Preprint*, arXiv:2503.24235.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models. *Preprint*, arXiv:2210.03493.

Wanjun Zhong, Siyuan Wang, Duyu Tang, Zenan Xu, Daya Guo, Jiahai Wang, Jian Yin, Ming Zhou, and Nan Duan. 2021. Ar-lsat: Investigating analytical reasoning of text. *Preprint*, arXiv:2104.06598.

A Appendix

A.1 Methodology Structure

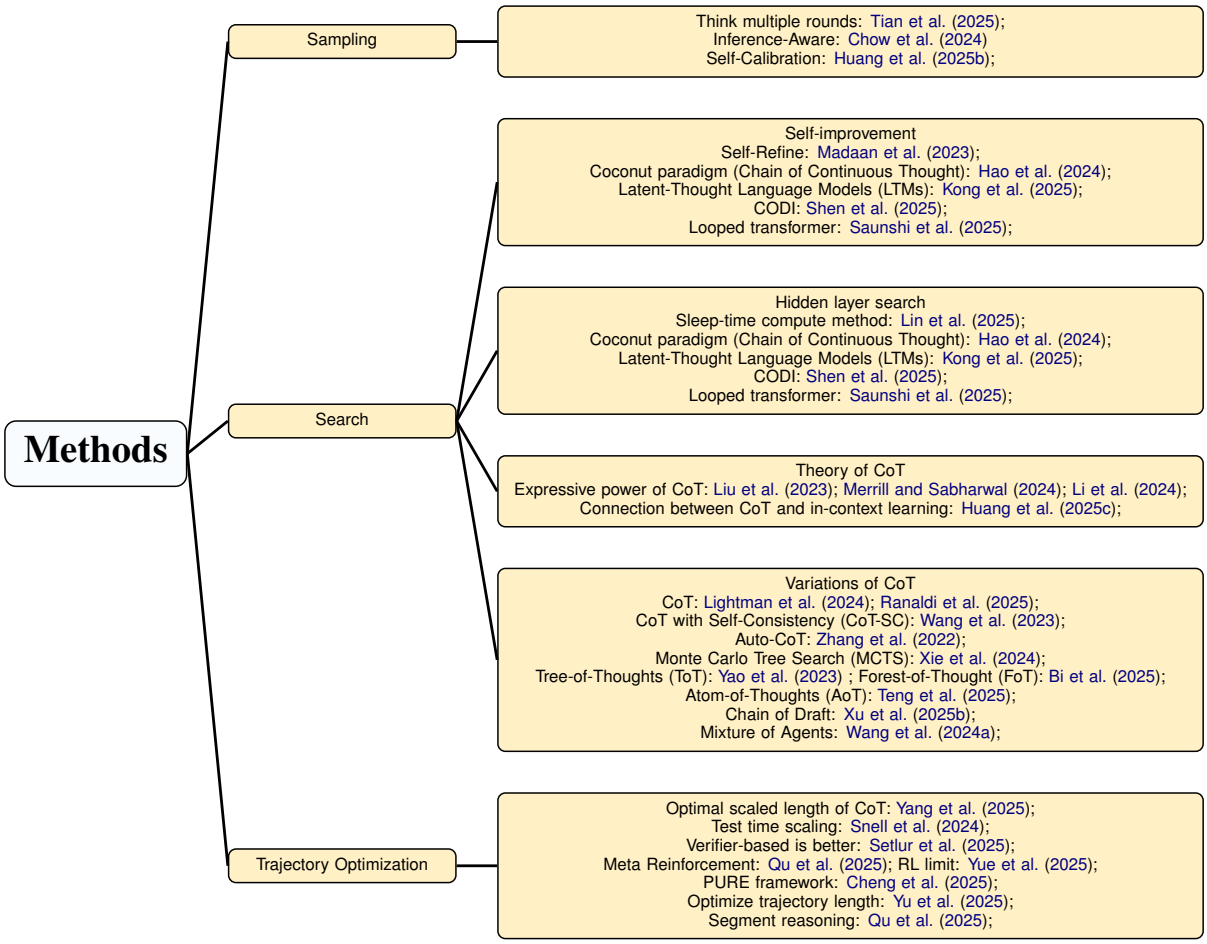

Figure 6: Methodology classification structure.

A.2 Related Work

There are numerous surveys focusing on various aspects of reasoning, post-training, and TTS. Pan et al. (2025); Xu et al. (2025a) survey reasoning models. Efficiency-focused surveys include Feng et al. (2025), Wang et al. (2025b), and Sui et al. (2025). TTS is covered by Zhang et al. (2025) and Li (2025). Post-training techniques are reviewed in Kumar et al. (2025b) and Tie et al. (2025).

Unlike previous work that systematically categorizes TTS methods, our study not only investigates TTS classification but, more importantly, proposes an experimental investigation into the trade-offs between reasoning proficiency and output diversity, inspired by our survey.

A.3 Applications in Real-World Domains

Robotics and Autonomous Systems In robotics, TTS facilitates improved decision-making and adaptability in dynamic environments (Zawalski et al., 2025). World Foundation Models (WFMs) (NVIDIA et al., 2025) simulate physical environments. They support tasks like autonomous driving and robotics. TTS improves their prediction. It also increases adaptability during inference.

Software Engineering and Autonomous Agents In software engineering, TTS enhances reasoning in autonomous agents handling complex development tasks. The SWE-Reasoner framework (Ma et al., 2025b) introduce a unified approach that combines internal and external test-time compute strategies to dynamically allocate computational resources during inference. Internally, it leverages development-contextualized long Chain-of-Thought trajectories to to guide smaller models for multi-step reasoning tasks. Externally, it incorporates reward-guided search and execution-based verification to focus inference-time compute on critical development phases such as fault localization and patch generation. Complementing this, multi-agent collaborative systems such as M1 + CEO ((Jin et al., 2025)) use a central coordination agent to dynamically manage reasoning depth, agent iteration, and communication strategies at test time.

Video Processing and Streaming Analytics TTS has shown promise in video processing Dalal et al. (2025), particularly in scenarios requiring real-time analysis and adaptation. The Test-Time Training (TTT) approach extends TTS to streaming video data. It helps models adapt to temporal changes. It also improves performance on tasks like instance segmentation and panoptic segmentation. TTT outperforms traditional fixed-model baselines by continuously updating the model with incoming video frames, thus enhancing accuracy in dynamic visual environments .

Medical Diagnostics and Clinical Decision Support TTS enhances the diagnostic capabilities by allowing more extensive reasoning during inference. (Huang et al., 2025d) demonstrate that increasing the reasoning token budget at test time significantly improves performance on medical question-answering tasks. However, the study also notes an optimal reasoning token budget, beyond which performance may degrade due to overthinking.

Additionally, TTS contributes to uncertainty estimation in medical image segmentation Ma et al. (2024). By applying test-time augmentation techniques, models can better assess aleatoric uncertainty, leading to more reliable segmentation outputs and reducing overconfident incorrect predictions.

Legal Document Analysis and Compliance Legal document analysis involves processing complex and lengthy texts, where TTS enhances the comprehension and reasoning abilities of AI models Kant et al. (2024). By allocating more computational resources during inference, models can better understand intricate legal language, identify relevant precedents, and ensure compliance with regulations. This capability is particularly valuable in tasks such as contract analysis, legal research, and compliance monitoring.

Takeaways. TTS enhances real-world applications by enabling adaptive, context-aware inference. In robotics, it improves decision-making in dynamic environments. In software engineering, it supports multi-step reasoning and verification in development tasks. For video, it adapts to streaming input for more accurate segmentation. In medicine, it boosts clinical QA performance and improves uncertainty estimation. In legal analysis, it helps models process complex language and ensure compliance.

A.4 Dataset

As shown in Table table 2, we have collected and summarized several common datasets used in previous work for future experimental implementation.

Dataset	Size	Domain	Description	Ref
DeepScaleR-Preview-Dataset	40K+	Math	Problem-answer pairs	Luo et al. (2025)
MATH	12.5K	Math	Competition mathematics problems	Hendrycks et al. (2021d)
GSM8K	8.5K	Math	Grade school math problems	Cobbe et al. (2021)
GSM-hard	1.32K	Math	Question-answer pairs	Gao et al. (2023)
GSM-Symbolic	5K	Math	Question-answer pairs	Mirzadeh et al. (2024)
TheoremQA	800	Math	Question-answer pairs	Bouamor et al. (2023)
SVAMP	1K	Math	Question-answer pairs	Patel et al. (2021)
MAWPS	3.3K	Math	Collection of math word problems	Koncel-Kedziorski et al. (2016)
AQUA-RAT	100K	Math	Grade-school-math problems	Ling et al. (2017)
OmniMATH	4428	Math	Judged by OmniJudge, GPT-4o	Gao et al. (2024)
Olympiad-Bench	8476	Math, Physics	Question-answer pairs	He et al. (2024)
Humaneval	Hundred	Code	Handwritten code	Chen et al. (2021)
APPS	10K	Code	Question-code solution pairs	Hendrycks et al. (2021a)
LiveCodeBench	442	Code	Question-code solution pairs	Jain et al. (2024)
MBPP	1K	Code	Basic algorithmic and functional programming tasks	Austin et al. (2021)
CommonsenseQA	12K	Commonsense	Multiple-choice question-answer pairs	Talmor et al. (2019)
StrategyQA	2.8K	Commonsense	Question-answer pairs	Geva et al. (2021)
ARC	7.8K	Commonsense	Multiple-choice question-answering pairs	Clark et al. (2018)
5-hop ProntoQA	∞	Logic	Question-answer pairs	Saparov and He (2023)
AR-LSAT	2,046	Law	Question-answer pairs	Zhong et al. (2021)
GPQA	Hundreds	Multiple domain	Question-answer pairs	Rein et al. (2024)
MMLU	231K	Multiple domain	Reasoning-focused question-answer pairs	Hendrycks et al. (2021c,b)
MMLU-pro	12K	Multiple domain	Reasoning-focused question-answer pairs	Wang et al. (2024b)

Table 2: Dataset on the recent reasoning methods

A.5 ADAPT Training Details

A.5.1 Custom Prompt Format

```
"{question} Please provide the initial step towards resolving the question. This step may serve as a foundation but might not encompass the entire solution.\n"
```

A.5.2 Training Parameter Setting

Dataset The combined dataset is shuffled and split into 90% for training and 10% for testing, with the test portion further divided evenly into evaluation and held-out sets.

Tokenization Tokenization includes padding to a maximum length of 512 tokens and truncation when necessary.

Training Training is performed for 3 epochs with a learning rate of 5×10^{-6}, per-device batch size of 4, and gradient accumulation steps of 8. We employ `bfloat16` precision, set `max_grad_norm` to 1.0.