Stop Overthinking:
A Survey on Efficient Reasoning for Large Language Models

Anonymous authors
Paper under double-blind review

Abstract

Large Language Models (LLMs) have demonstrated remarkable capabilities in complex tasks. Recent advancements in Large Reasoning Models (LRMs), such as OpenAI o1 and DeepSeek-R1, have further improved performance in System-2 reasoning domains like mathematics and programming by harnessing supervised fine-tuning (SFT) and reinforcement learning (RL) to enhance Chain-of-Thought (CoT) reasoning. However, while longer CoT reasoning sequences improve performance, they also introduce significant computational overhead due to lengthy and redundant outputs, known as the "overthinking phenomenon".

Efficient Reasoning, which seeks to optimize reasoning length and computation during inference of reasoning models while preserving reasoning capabilities, offers practical benefits such as faster processing times, lower energy consumption, and improved responsiveness, especially valuable for reasoning-intensive applications. Despite its potential, efficient reasoning remains in the early stages of research.

In this paper, we provide the first structured survey to systematically investigate and explore the current progress toward achieving efficient reasoning in LLMs. Overall, relying on the inherent mechanism of LLMs, we categorize existing works into several key directions: ***(1) model-based efficient reasoning***, which considers optimizing full-length reasoning models into more concise reasoning models or directly training efficient reasoning models; ***(2) reasoning output-based efficient reasoning***, which aims to dynamically reduce reasoning steps and length during inference; ***(3) input prompts-based efficient reasoning***, which seeks to enhance reasoning efficiency based on input prompt properties such as difficulty or length control. Additionally, we introduce the use of efficient data for training reasoning models, explore the reasoning capabilities of small language models, and discuss evaluation methods and benchmarking.

1 Introduction

Large Language Models (LLMs) have emerged as exceptionally powerful AI tools, demonstrating advanced capabilities in natural language understanding and complex reasoning. Recently, the rise of reasoning-focused LLMs, also referred to as reasoning-capable models or Large Reasoning Models (LRMs) (Xu et al., 2025a) such as OpenAI o1 (OpenAI, 2024) and DeepSeek-R1 (Guo et al., 2025), has significantly improved performance in System-2 reasoning domains (Li et al., 2025d), particularly in challenging mathematics (Cobbe et al., 2021; Hendrycks et al., 2021) and programming tasks (Codeforces, 2025; Chen et al., 2021). Evolving from foundational pretrained models (e.g., LLaMA (Touvron et al., 2023; Grattafiori et al., 2024)) trained with next-token prediction (Devlin et al., 2019), these models typically leverage Chain-of-Thought (CoT) (Wei et al., 2022) reasoning chains to generate explicit, step-by-step reasoning sequences before arriving at a final answer, significantly improving their effectiveness in reasoning-intensive tasks.

Such reasoning abilities in LLMs are typically developed through supervised fine-tuning (SFT) and reinforcement learning (RL), which promote iterative and systematic problem-solving abilities. For instance, DeepSeek-R1 (Guo et al., 2025) undergoes multiple rounds of SFT and RL training, emphasizing structured thinking templates and rule-based reward mechanisms. In particular, the rule-based rewards provides precise

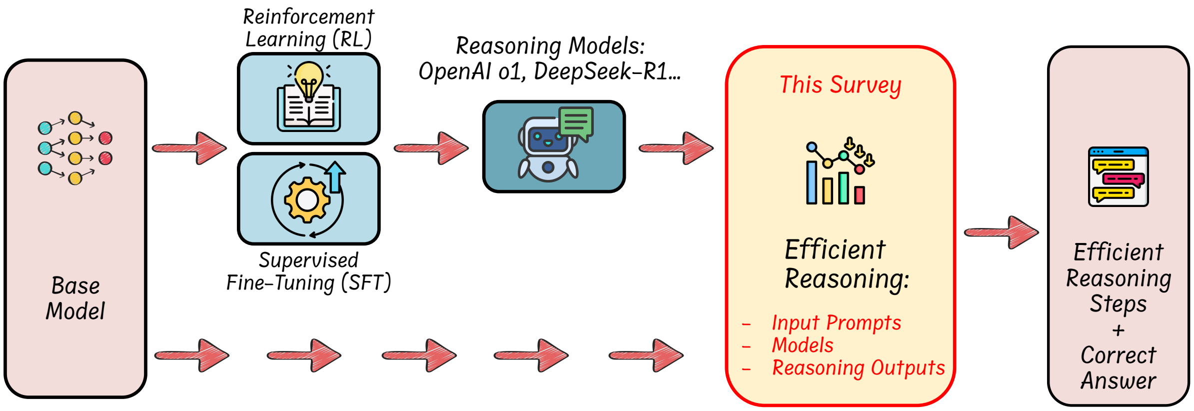

Figure 1: The pipeline of developing efficient reasoning for LLMs. A reasoning model can be trained on the base model using SFT, RL, or a combination of both. While reasoning models demonstrate strong reasoning capabilities, they often suffer from the "overthinking phenomenon", generating unnecessarily lengthy reasoning steps. To improve efficiency, various methods can be applied to reduce redundant steps while maintaining accuracy during post-training based on a reasoning model, or to fine-tune non-reasoning models to incorporate efficient reasoning capabilities. This approach enables the model to answer questions with concise and effective reasoning steps. In this paper, we explore the latest progress in efficient reasoning for LLMs, aiming to provide insights that can guide future research and the development of reasoning-driven applications across various domains.

and explicit feedback signals during training, effectively enhancing the general reasoning capabilities beyond the pretrained LLM.

However, while long CoT reasoning significantly boosts accuracy, step-by-step thinking mechanisms also lead to lengthy output responses, resulting in substantial computational overhead and increased reasoning time. For instance, the "overthinking problem" arises when answering a simple question (Chen et al., 2024c) like, "*what is the answer of 2 plus 3?*" Some reasoning models, especially smaller ones, can generate reasoning sequences spanning thousands of tokens. This verbosity significantly increases both inference costs and latency, limiting the practical application of reasoning models in computation-sensitive real-world scenarios, such as real-time autonomous driving systems, interactive conversational assistants, precision robotic control tasks, and large-scale online search engines.

Efficient reasoning, particularly the reduction of reasoning length and computation during inference of reasoning models as shown in Figure 1, provides direct cost reduction and improved feasibility for real-world deployments. Recently, numerous studies (Luo et al., 2025; Yeo et al., 2025; Han et al., 2024; Ma et al., 2025b; Hao et al., 2024) have explored ways to develop more concise reasoning paths, making efficient reasoning a rapidly evolving research area.

In this paper, we present the first structured survey systematically exploring the progress in efficient reasoning for LLMs. As illustrated in Figure 2, we categorize existing work into three key directions: *(1) Model-based efficient reasoning*, which focuses on optimizing full-length reasoning models into more concise variants or directly training efficient reasoning models. *(2) Reasoning output-based efficient reasoning*, which dynamically reduces reasoning steps and length during inference. *(3) Input prompts-based efficient reasoning*, which enhances reasoning efficiency based on input properties such as difficulty or length control. Unlike model compression techniques such as quantization (Xiao et al., 2023; Frantar et al., 2023; Lin et al., 2024) or KV cache compression (Zhang et al., 2023; Liu et al., 2024b; Shi et al., 2024; Yuan et al., 2024), which focus on reducing model size for lightweight inference, efficient reasoning in LLMs emphasizes *smart and concise reasoning* by optimizing the length of *generated* reasoning sequences and reducing unnecessary thinking steps.

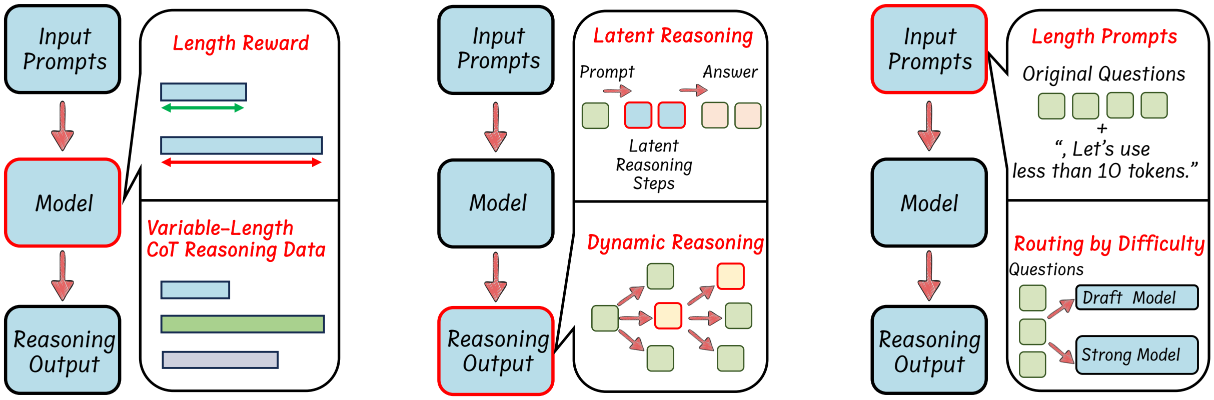

Figure 2: Overview of efficient reasoning methods, which can be summarized as model-oriented (Left: *I*, *II*) and reasoning output-oriented (Middle: *III*, *IV*), and input prompts-oriented (Right: *V*, *VI*) methods. Specifically, (I) Reinforcement Learning with Length Reward Design (Section 3.1); (II) Supervised Fine-Tuning with Variable-Length CoT Data (Section 3.2); (III) Compressing Reasoning Steps into Fewer Latent Representation (Section 4.1); (IV) Dynamic Reasoning Paradigm during Inference (Section 4.2); (V) Prompt-guided Efficient Reasoning (Section 5.1); (VI) Routing Prompts to Optimize Reasoning Efficiency (Section 5.2);

Overall, we provide a summary of the current key approaches to efficient reasoning, organizing them into the following categories:

- Reinforcement Learning with Length-Based Reward Design (Section 3.1)
- Supervised Fine-Tuning with Variable-Length CoT Data (Section 3.2)
- Compressing Reasoning Steps into Fewer Latent Representations (Section 4.1)
- Dynamic Reasoning Paradigms During Inference (Section 4.2)
- Prompt-Guided Efficient Reasoning (Section 5.1)
- Routing Prompts to Optimize Reasoning Efficiency (Section 5.2)

Furthermore, beyond these approaches, we also include efficient LLM techniques towards reducing the computation related to the reasoning models, including efficient data for reasoning model training, and model compression methods for reasoning models. We explore other relevant topics, including:

- Training Reasoning Models with Efficient Data (Section 6.1)
- Reasoning Abilities of Small Language Models and Model Compression (Section 6.2)
- Evaluation and Benchmarking of Efficient Reasoning Models (Section 7)

The overall taxonomy of existing literature related to efficient reasoning for LLMs and efficient techniques related to large reasoning models is provided in Figure 3.

2 Background: Long CoT Reasoning Models and the Overthinking Phenomenon

2.1 Chain-of-Thought (CoT) Reasoning

Chain-of-Thought (CoT) reasoning (Wei et al., 2022) is a key approach that has been purposefully introduced in LLMs to enhance their reasoning capabilities. In this setting, models are typically prompted to generate

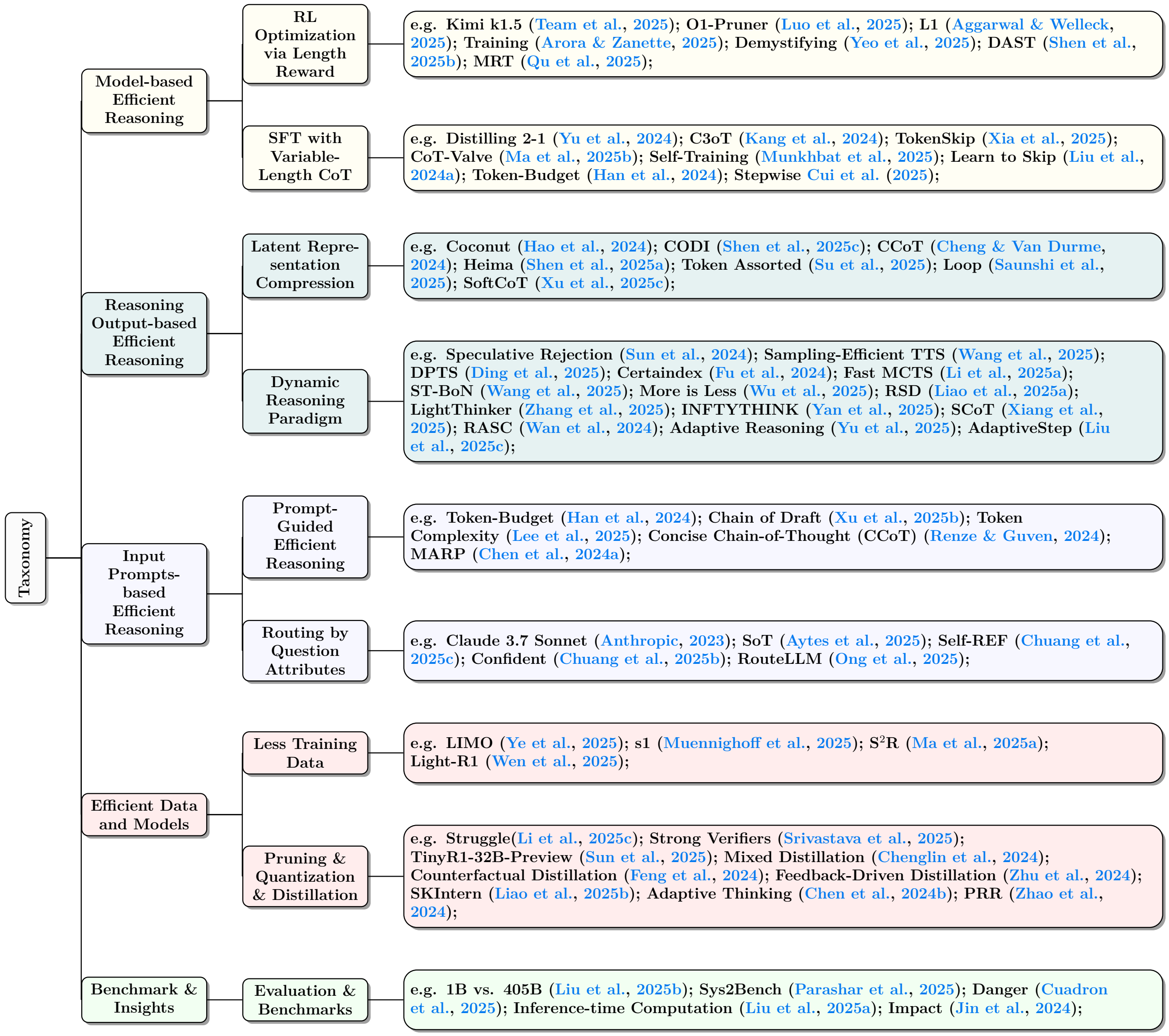

Figure 3: Taxonomy of existing literature on efficient reasoning for LLMs and efficient techniques related to large reasoning models.

a structured reasoning chain before arriving at a final answer. Techniques in this domain have been shown to improve overall accuracy (Wei et al., 2022) since a higher-quality generation context often leads to more consistent and reliable final results. Several notable CoT variants have been developed: Self-Consistency CoT (Wang et al., 2023) replaces the standard greedy decoding approach by sampling diverse reasoning paths and selecting the most consistent answer through marginalization and aggregation. Tree-of-Thought (ToT) prompting (Yao et al., 2023) further structures the reasoning process as a tree with backtracking, significantly improving efficiency in solving parallelizable subtasks. Graph-of-Thoughts (GoT) prompting (Besta et al., 2024) extends this concept by structuring thoughts into a graph, allowing iterative refinement of individual reasoning steps. While many CoT variants exist, they generally involve different prompting techniques to guide the behavior of models, sometimes incorporating controller-like mechanisms to manage thought progression and usage.

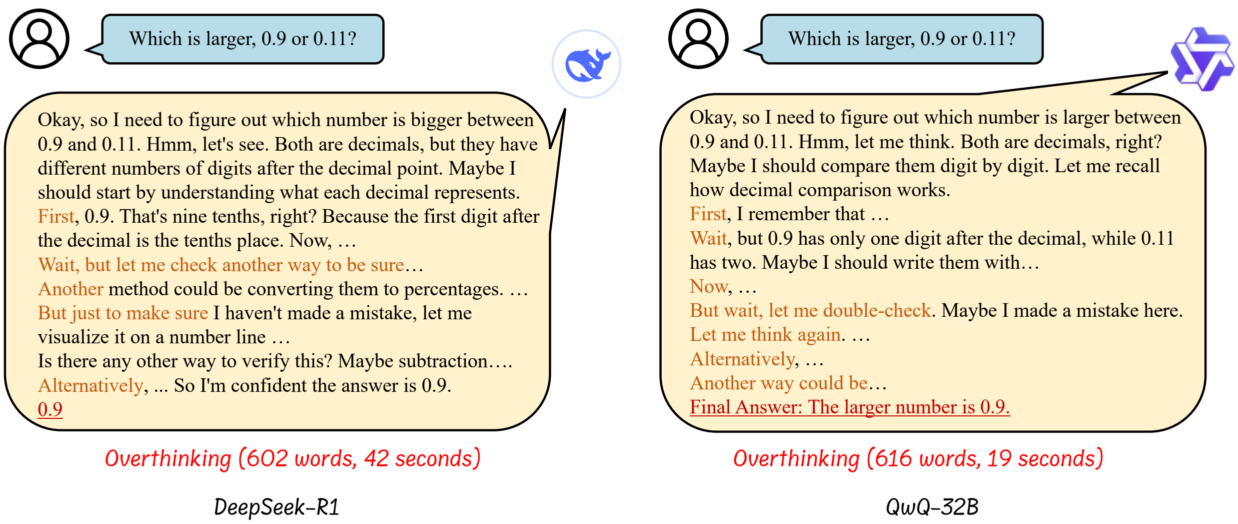

Figure 4: An example of the "overthinking phenomenon": when asked ***"Which is larger, 0.9 or 0.11?"***, the reasoning model takes an unnecessarily long time (e.g., 19 seconds for QwQ-32B (Team) and 42 seconds for DeepSeek-R1 (Guo et al., 2025)) to arrive at the correct answer. This example was tested in March 2025.

2.2 The Mechanism Behind Large Reasoning Models

Multi-step reasoning refers to the ability of LLMs ability to generate structured reasoning steps before committing to a final answer. This capability is particularly beneficial for logic-intensive tasks such as mathematics and programming. More broadly, reasoning-capable models are often favored by human users over their non-reasoning counterparts, as evidenced by rankings in the Chatbot Arena LLM Leaderboard.[1]

Recent reasoning models, such as DeepSeek-R1 (Guo et al., 2025) and OpenAI o1 (Luo et al., 2025), are known or believed to have internalized reasoning behaviors, reducing reliance on explicit test-time augmentations. These models generate detailed CoT reasoning by iteratively producing intermediate steps and refining solutions sequentially until reaching a final answer. Unlike traditional CoT approaches, which rely on prompting, these reasoning models internalize their reasoning capability through extensive training.

The OpenAI o1 model is speculated to employ a tree-based search approach, such as Monte Carlo Tree Search (MCTS) (Kocsis & Szepesvári, 2006; Coulom, 2006), combined with a Process Reward Model (PRM) to explore reasoning paths and determine optimal solutions through guided simulations.[2] DeepSeek-R1, on the other hand, explicitly learns its reasoning capability through supervised fine-tuning and reinforcement learning, with a particular emphasis on rule-based rewards for math and coding tasks. These models are trained to generate reasoning steps in a predefined format before arriving at their final answers.

2.3 The Overthinking Problem in Long CoT Reasoning Models

The "overthinking phenomenon" (Team et al., 2025; Chen et al., 2024c) in long CoT reasoning models refers to situations where LLMs generate excessively detailed or unnecessarily elaborate reasoning steps, ultimately reducing their problem-solving efficiency. In particular, many modern reasoning models, especially those with smaller parameter scales, tend to produce verbose reasoning or redundant intermediate steps, making them unable to provide answers within the user-defined token budget. In worse cases, excessive reasoning steps introduce errors or obscure logical clarity, leading to incorrect answers.

[1]A community-driven evaluation of leading LLMs and AI chatbots: `https://lmarena.ai/?leaderboard`.

[2]There is no official confirmation regarding OpenAI o1's training details and mechanisms. However, sources such as `https://www.interconnects.ai/p/openais-o1-using-search-was-a-psyop` and `https://www.youtube.com/watch?v=6PEJ96k1kiw` discuss these speculations in detail and are recommended for interested readers.

Table 1: Comparison of different length reward-based RL methods. $L(\cdot)$ denotes the way of calculating the prediction length. r_0^c/r_0^w denotes reward (correct/wrong) for $L(\cdot)$=0. r_L^c/r_L^w Reward (correct/wrong) for $L(\cdot) = L_{\max}(\cdot)$ with hyperparameter α. r_e is the exceed length penalty. y_{GT} represents the ground truth answer, y_{pred} is the prediction, and y_{ref} is the reference prediction from reference model of input data x of dataset D. π_{ref} is the policy of reference model and π_θ is the policy of targeted-to-train LLMs.

Method	**RL**	**Length Constraint Reward**	**Data**	**Model**
O1-Pruner (Luo et al., 2025)	PPO	$\mathbb{E}_{x\sim D}\left[\mathbb{E}_{\pi_\theta,\pi_{\mathrm{ref}}}\left[\frac{L(y_{\mathrm{ref}})}{L(y_{\mathrm{pred}})}\right]-1\right]$	GSM8K GaoKao MATH-500	Marco-o1-7B QwQ-32B-Preview
Demystifying (Yeo et al., 2025)	PPO	$\begin{cases} r_0^c + 0.5\times(r_L^c - r_0^c)(1+\cos(\frac{\pi L(y_{\mathrm{pred}})}{L_{\max}}), & \text{if correct,} \\ r_0^c + 0.5\times(r_L^w - r_0^w)(1+\cos(\frac{\pi L(y_{\mathrm{pred}})}{L_{\max}}), & \text{if wrong} \\ r_e, & \text{if } L(y_{\mathrm{pred}}) = L_{\max}, \end{cases}$	MATH-500 AIME-2024 TheoremQA MMLU-Pro-1k	LLaMA-3.1-8B Qwen2.5-7B-Math
L1 (Aggarwal & Welleck, 2025)	GRPO	$\begin{cases} x_{\mathrm{new}} = \mathit{CONCAT}\,(x, \text{“}\textit{Think for N tokens.}\text{”}), \\ r(y, y_{GT}, L(y_{GT})) = \mathbb{I}(y_{\mathrm{pred}} = y_{GT}) - \alpha\cdot\lvert L(y_{GT}) - L(y_{\mathrm{pred}})\rvert \end{cases}$	AMC GPQA LAST MMLU MATH-500 AIME-2024 Olympiad-Bench	DeepSeek-R1-Distill-Qwen-1.5B
DAST (Shen et al., 2025b)	SimPO	Trained with constructed length preference data	MATH-500 AIME-2024	DeepSeek-R1-Distill-Qwen-8B DeepSeek-R1-Distill-Qwen-32B
Training (Arora & Zanette, 2025)	PG	$\mathbb{E}_{x\sim D}\left[\mathbf{1}\{y_{\mathrm{pred}} = y_{\mathrm{GT}}\}(1-\alpha f(L(y_{\mathrm{pred}})))\right]$	GSM8K MATH-500 AIME-2024	DeepSeek-R1-Distill-Qwen-1.5B DeepSeek-R1-Distill-Qwen-7B

Figure 4 illustrates an example of overthinking. Even though the model arrives at the correct answer early in its reasoning process, it continues generating unnecessary intermediate steps, leading to inefficiencies. Given the substantial resource costs associated with LLM inference (e.g., OpenAI o1 costs \$60 per 1M generated tokens), such behavior is highly undesirable. Moreover, the problem becomes even worse if longer reasoning leads to wrong answers. In contrast, efficient reasoning models would use fewer reasoning steps to obtain correct answers while reducing inference costs.

Addressing this challenge is particularly difficult because the pretraining recipes for reasoning-capable models often explicitly encourage generating extended reasoning steps to improve accuracy. For example, DeepSeek-R1-Zero, a more or less a development prototype of DeepSeek-R1, exhibits a direct correlation between increased training duration with longer response lengths and improved benchmark performance (Guo et al., 2025). These trends are often viewed as proxies for successful reasoning training. Consequently, improving inference efficiency requires working against certain pretraining objectives, making it a non-trivial challenge.

This paper aims to systematically summarize various approaches and methodologies toward achieving the challenging yet valuable goal of developing reasoning models with high efficiency and strong reasoning capabilities.

3 Model-based Efficient Reasoning

From the model perspective, these works focus on fine-tuning LLMs to improve their intrinsic ability to reason concisely and efficiently.

3.1 RL with Length Reward Design

Most reasoning models are trained using RL-based methods (e.g., DeepSeek-R1 (Guo et al., 2025), DeepSeek-R1-Zero (Guo et al., 2025), OpenAI o1 (OpenAI, 2024), QwQ-32B-Preview (Team)) which focus on the accuracy reward and format rewards (Guo et al., 2025). To enhance reasoning-length efficiency, some studies propose integrating a length reward into the RL framework, which effectively shortens the reasoning process (as shown in Table 5). In principle, the length reward assigns higher scores to short, correct answers while penalizing lengthy or incorrect ones, thereby optimizing the length of the reasoning path.

Table 2: Comparison of different policy optimization methods in CoT length controls. $\hat{R}_t$ represents the reward model. π_{ref} is the policy of reference model and π_θ is the policy of target-to-train LLMs. γ is a target reward margin term for SimPO and β is a hyperparameter to amply the preference to winning than losing. λ is a clipping-related hyper-parameter with KL-divergence $\mathbb{D}_{KL}[\cdot||\cdot]$. clip$(\cdot)$ is a gradient clipping function with hyperparameter ϵ to control the gradient updating progress of GRPO. The y_w is for winning responses, and y_l is for losing responses of the given input data x_t, where some with G on superscript denote the outputs of different sampled groups.

Method	Optimization Objective
Policy Gradient (PG)	$\mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(y_t\|x_t)\hat{R}_t\right]$
PPO (Schulman et al., 2017)	$\mathbb{E}\left[\min\left(\frac{\pi_\theta(y_t\|x_t)}{\pi_{\theta_{\text{ref}}}(y_t\|x_t)}\hat{R}_t,\ \text{clip}\left(\frac{\pi_\theta(y_t\|x_t)}{\pi_{\theta_{\text{ref}}}(y_t\|x_t)},\ 1-\epsilon,\ 1+\epsilon\right)\hat{R}_t\right)\right]$
SimPO (Meng et al., 2024)	$\mathbb{E}\left[\log\sigma\left(\frac{\beta}{\|y_t^w\|}\log\pi_\theta(y_t^w \mid x_t) - \frac{\beta}{\|y_t^l\|}\log\pi_\theta(y_t^l \mid x_t) - \gamma\right)\right]$
GRPO (Shao et al., 2024)	$\mathbb{E}\left[\min\left(\frac{\pi_\theta(y_t^G\|x_t)}{\pi_{\theta_{\text{ref}}}(y_t^G\|x_t)}\hat{R}_t^G,\ \text{clip}\left(\frac{\pi_\theta(y_t^G\|x_t)}{\pi_{\theta_{\text{r}}}(y_t^G\|x_t)},\ 1-\epsilon,\ 1+\epsilon\right)\hat{R}_t^G\right) - \lambda\mathbb{D}_{KL}[\pi_\theta\|\|\pi_{\text{ref}}]\right]$

The key question is: How to formulate the length reward in RL?

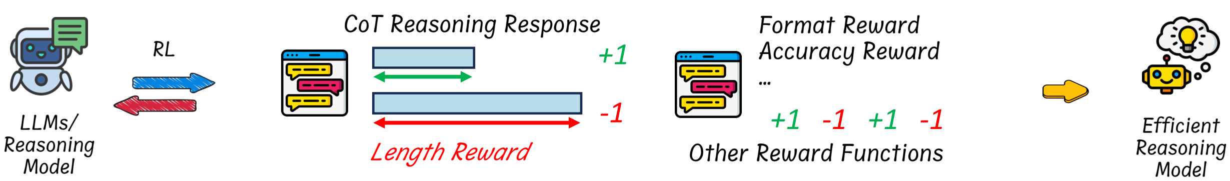

Figure 5: Illustration of the method for RL fine-tuning with length reward designs. In principle, the length reward assigns higher rewards to short, correct answers and penalizes lengthy or wrong answers to achieve efficient reasoning LLMs.

Existing works leverage traditional RL optimization techniques combined with ***explicit length-based reward*** to control the length of CoT reasoning are shown in Figure 5. The detailed length rewards are shown in Table 1, and the details of different reinforcement learning optimization objectives, accompanied with length rewards, are shown in Table 2. The work (Arora & Zanette, 2025) proposes utilizing length-based rewards conditioned on correctness, where shorter correct answers receive higher rewards. They then apply traditional policy gradient methods guided by this reward scheme to encourage LLMs to produce concise reasoning steps. Expanding from the policy gradient, the following discussed work is primarily built upon proximal policy optimization (PPO) (Schulman et al., 2017) with CoT length penalty. Demystifying (Yeo et al., 2025) presents empirical findings from RL experiments examining how reasoning capability is influenced by length. They demonstrate that RL does not consistently or reliably increase the length and complexity of CoT reasoning, emphasizing the necessity of controlling CoT length growth to ensure stable performance. To mitigate these issues, they proposed a Cosine Reward based on a Dirichlet function of a concise reward formula (Loshchilov & Hutter, 2016) and the proposed "exceed length penalty" scores. Due to the performance impact of CoT length, Kimi k1.5 (Team et al., 2025) incorporates a length penalty into its policy optimization (a variant of online policy mirror decent (Tomar et al., 2020)) to improve long CoT activations and facilitate effective model merging. Besides optimizing with length penalty reward, L1 (Aggarwal & Welleck, 2025) modify the training data with the designated length constraint instruction (i.e., Think for N tokens) before launching the policy optimization with pre-trained reasoning LLMs. O1-Pruner (Luo et al., 2025) introduces the Length-Harmonizing Reward, combined with a PPO-style loss, to optimize reasoning LLMs by effectively shortening the CoT length. Specifically, the Length-Harmonizing Reward is computed based on the ratio of CoT lengths between the reference model output and the predicted results. Additionally, this reward incorporates accuracy-based constraints comparing predictions to the reference model outputs, ensuring that shortening the reasoning process does not degrade task performance. Without relying on a reference model, DAST (Shen et al., 2025b) employs SimPO (Meng et al., 2024) to fine-tune reasoning LLMs using a constructed length-preference dataset. This dataset is generated based on a self-defined token-length budget measurement L_{budget}, defined as a linear combination of the average token length of correct responses and the maximum allowed generation length.

These RL-based methods enable the mitigation of overthinking in reasoning-capable LLMs, where overthinking refers to unnecessarily extended reasoning processes, leading to longer inference times and exceeding computational budgets. By achieving nearly lossless alignment with the original reasoning capabilities of LLMs, these budget-efficient RL strategies democratize the deployment of reasoning LLMs in resource-constrained scenarios.

3.2 SFT with Variable-Length CoT Data

Fine-tuning LLMs with variable-length CoT data is an effective way to improve the efficiency of reasoning. As shown in Figure 6, this series of works typically involves: (1) Constructing variable-length CoT reasoning datasets via various methods, and (2) Applying SFT with collected data on reasoning models to enable LLMs to learn compact reasoning chains that encapsulate effective knowledge. Note that this method is not limited to RL-trained reasoning models; it can also directly enhance reasoning models by injecting efficient reasoning capabilities, similar to those used in distilled reasoning models.(e.g., DeepSeek-R1-Distill-Qwen (Guo et al., 2025)).

The key question is: How to collect variable-length CoT reasoning data, especially for short CoT data?

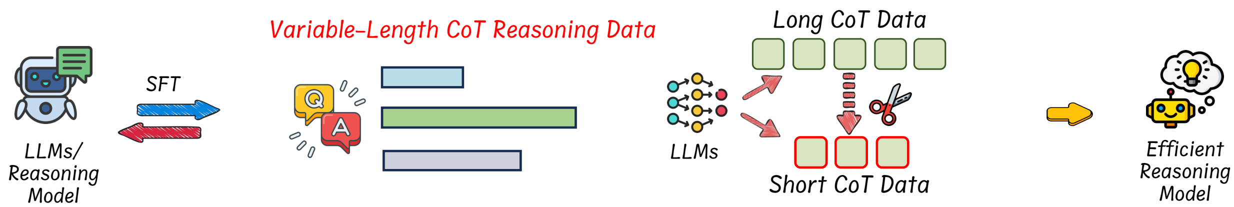

Figure 6: Illustration of methods for utilizing SFT with variable-length CoT reasoning datasets.

3.2.1 Constructing Variable-Length CoT Reasoning Datasets

Variable-length CoT reasoning datasets refer to datasets of long/short reasoning steps that could guide LLMs to achieve correct answers. Existing works typically gather long CoT data by prompting pre-trained reasoning models with questions. Based on the long CoT data, the key challenge is: *How to collect short CoT data?* Overall, variable-length CoT reasoning datasets can be created via either post-reasoning or during-reasoning. We list some detailed approaches in Table 3.

Post-reasoning CoT Compression. This approach collects short CoT data by reducing redundant reasoning steps after full-length reasoning, either by heuristic criterion or LLMs (Yu et al., 2024; Kang et al., 2024; Xia et al., 2025). Specifically, Yu et al. (2024) uses reasoning-capable LLMs to generate the reasoning and answers. After generating full-length CoT data, they discard the reasoning process, only using the questions and answers to distill system-1 LLMs. Another work C3oT improves the reasoning efficiency by compressing the reasoning process (Kang et al., 2024). The long CoT reasoning steps were generated by explicitly prompting LLMs. Then, it employs GPT-4 as a compressor to reduce the length of the reasoning process while ensuring the compressed reasoning retains all key information and removes redundant words. In addition, TokenSkip reduce the reasoning steps driven by interpretation (Xia et al., 2025). It estimates the semantic importance of each reasoning part to the final answer and reduces the reasoning tokens. The

Table 3: Comparison of various approaches that utilize SFT with variable-length CoT reasoning datasets.

Method	Source Data	Reasoning Pruning	SFT	LLMs
Self-Training (Munkhbat et al., 2025)	GSM8K MATH	Sampling N reasoning then select the shortest one	Standard	Llama-3.2-{1B,3B} Llama-3.1-8B
TokenSkip (Xia et al., 2025)	GSM8K MATH	Skip tokens according to semantic importance	Standard	LLaMA-3.1-8B-Instruct Qwen2.5-Instruct
C3oT (Kang et al., 2024)	GSM8K MathQA ECQA StrategyQA	GPT-4 as compressor to make concise reasoning	Standard	Llama-2-chat-{7B,13B}
Distilling2-1 (Yu et al., 2024)	OASST2	Removing reasoning	Standard	Llama-2-70B-chat
Token-Budget (Han et al., 2024)	GSM8K GSM8K-Z MathBench	Persuing an optimal token budget for LLMs to complete the reasoning	Standard	Llama-3.1-8B-Instruct
CoT-Valve (Ma et al., 2025b)	GSM8K PRM800k	Merging parameters of non-reasoning and long reasoning LLMs	Progressive	QwQ-32B-Preview DeepSeek-R1-Distill-Llama-8B LLaMA-3.1-8B LLaMA-3.2-1B Qwen32B-Instruct
LearnSkip (Liu et al., 2024a)	Analog of Algebra Multi-digit Addition Directional Reasoning	Stage 1: Manually skipping Stage 2: Prompting LLMs for shorter reasoning	Standard & Progressive	Llama-2-7B Phi-3-mini (3.8B)

important parts preserve the key reasoning steps that could improve the accuracy of the final answer. The advantage of post-reasoning CoT compression is that it can achieve a higher reduction rate of the reasoning steps, which advances more efficient reasoning.

Obtaining Compressed CoT Data during Reasoning. This approach collects short CoT data by prompting LLMs to generate short reasoning steps during inference and reasoning (Liu et al., 2024a; Munkhbat et al., 2025; Han et al., 2024; Ma et al., 2025b). Specifically, Liu et al. (2024a) proposes a human-like step-skipping method for generating shorter reasoning steps. In the first stage, based on the original training datasets, they manually create solutions by skipping steps, either guided by human expertise or by randomly merging or removing steps. Further, these concise data are labeled with prompts such as "Solve it in n steps.". After SFT, the model is able to generate shorter reasoning paths. In the second stage, they prompt this model to solve problems by intrinsically skipping or compressing steps during reasoning. The generated concise reasoning steps with questions and answers are collected as datasets, which are then used in SFT to make LLMs solve problems with fewer steps. Moreover, Token-Budget (Han et al., 2024) has an important insight: an optimal token budget helps LLMs actively follow the token constraint to complete the reasoning process. Motivated by this insight, it proposes a binary search-based method to achieve the optimal token budgets, and follow these budgets to generate short reasoning steps. In addition, Munkhbat et al. (2025) proposes a sampling-based method to improve reasoning efficiency. Specifically, it examines the distribution of reasoning lengths and finds that shorter solutions appear more frequently than the typical reasoning length. Driven by this finding, it proposes a Best-of-N (BoN) Sampling at test time, which generates N paths of reasoning and selects the shortest one. These short reasoning paths are collected as the dataset. Finally, CoT-Valve (Ma et al., 2025b) controls the reasoning length by mix-up the parameters of long reasoning and non-reasoning LLMs for generating variable-length reasoning steps. They also release their mixed dataset, such as MixChain-Z-GSM8K and MixChain-C-LIMO. The advantage of CoT compression during reasoning is that the naturally generated reasoning steps align with the intrinsic knowledge of LLMs, which advances more effective learning of LLMs.

3.2.2 Fine-Tuning Approaches

After collecting variable-length CoT data, existing works fine-tune LLMs to achieve efficient reasoning in several ways, which include standard fine-tuning (e.g., parameter-efficient fine-tuning such as LoRA (Hu et al., 2022) or full fine-tuning) and progressive fine-tuning.

Standard Fine-tuning. Most of the work adopts standard methods to fine-tune LLMs (Liu et al., 2024a; Munkhbat et al., 2025; Yu et al., 2024; Kang et al., 2024; Xia et al., 2025; Han et al., 2024). Specifically, these approaches adopt LoRA (Hu et al., 2022) or full fine-tuning (Kang et al., 2024) to minimize the perplexity loss function or DPO loss function (Han et al., 2024) on the reasoning-efficient datasets. The LoRA enables LLMs to adapt to short reasoning steps with less than 1% of the parameters tuned. In addition, (Liu et al., 2024a) observed the growing reasoning efficiency can generalize to out-of-domains beyond the collected datasets.

Progressive Fine-tuning. Progressive fine-tuning aims to smoothly reduce the reasoning steps during fine-tuning (Ma et al., 2025b; Liu et al., 2024a). One way is to progressively reduce the reasoning steps of data during fine-tuning LLMs, as employed in (Liu et al., 2024a). Another effective way is to progressively adjust the generation of reasoning steps, as proposed by CoT-Valve (Ma et al., 2025b). Specifically, it first learns LoRA adaptor $\Delta\theta_{\mathrm{N}}$ and θ_{L}, where LLMs with $\Delta\theta_{\mathrm{N}}$ have no reasoning steps, and that with $\Delta\theta_{\mathrm{L}}$ have long reasoning. Then, it mix-up $\Delta\theta_{\mathrm{N}}$ and $\Delta\theta_{\mathrm{long}}$ by $\alpha\Delta\theta_{\mathrm{N}}+(1-\alpha)\Delta\theta_{\mathrm{L}}$ to generate a dataset reasoning with variable length. Here $0<\alpha<1$ controls the parameter to shift from $\Delta\theta_{\mathrm{N}}$ to $\Delta\theta_{\mathrm{L}}$, controlling the reasoning length generated by LLMs. Finally, it fine-tunes LLMs on the generated data while progressively reducing α from 1 to 0. In this way, reasoning efficiency is progressively improved during fine-tuning.

4 Reasoning Output-based Efficient Reasoning

From the perspective of reasoning steps in the output, these works focus on modifying the output paradigm to enhance the ability of LLMs to reason concisely and efficiently.

4.1 Compressing Reasoning Steps into Fewer Latent Representation

Although standard CoT methods improve LLM performance by explicitly writing reasoning steps, recent work (Deng et al., 2024) has shown that simply adding intermediate "thinking" tokens, or even meaningless filler (e.g., "......") (Pfau et al., 2024), can also increase performance. Geiping et al. (2025) scales up deeper reasoning through recurrent expansions in the hidden space rather than verbose text. These findings highlight that the benefit often lies in more hidden computation rather than purely textual decompositions. Building on the insight that latent reasoning can allow LLMs to reason more efficiently and flexibly, *with fewer (or no) explicit textual intermediate steps, several new methods focus on compressing or replacing explicit CoT with more compact latent representations.*

The key question is: How to compress reasoning steps into latent space?

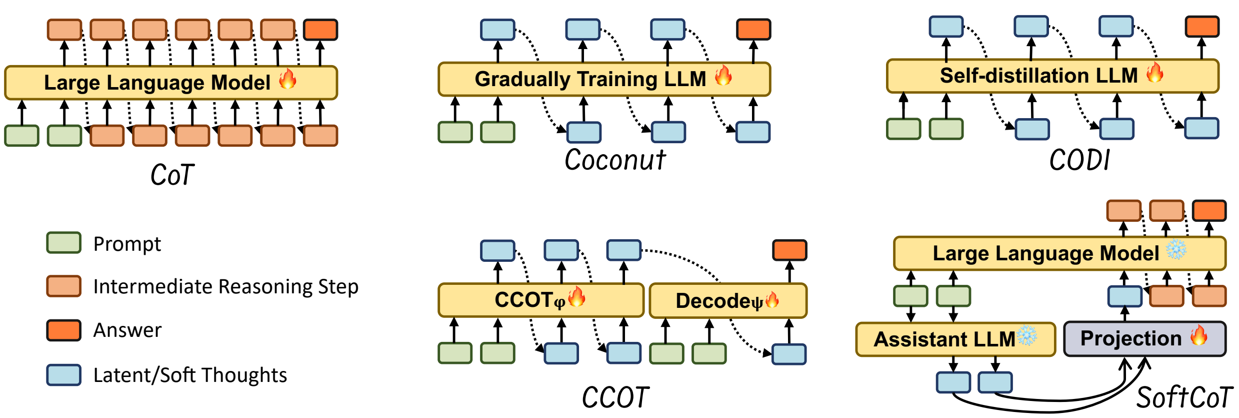

Figure 7: Comparison of methods of compressing reasoning steps into fewer latent representations.

In general, these methods can be categorized into two types: training LLMs to inference using latent representations or using an auxiliary model. A visualized comparison of some of these approaches is presented in Figure 7.

Training LLMs to Leverage Latent Representations. Among the first explorations, Coconut (Chain of Continuous Thought) (Hao et al., 2024) treats the final-layer hidden states of an LLM as "continuous thought" to replace traditional discrete tokens. It then reuses these hidden states as the next input embeddings. Trained step by step, Coconut gradually adds these latent CoT tokens. The results suggest that compressing tokens into latent representations improves both accuracy and efficiency by reducing the number of intermediate "thinking" tokens. CODI (Shen et al., 2025c) leverages a different training process compared to Coconut, which learns the continuous latent CoT via *self-distillation*. In CODI, the model serves both teacher and student, jointly learning explicit and implicit CoT while aligning hidden activations on the token, generating the final answer. This self-distillation process enables LLMs to perform reasoning internally without generating explicit CoT tokens. Similarly, CCOT (Cheng & Van Durme, 2024) condenses long CoT reasoning into short *contentful and continuous contemplation tokens*. First, it precomputes the full CoT for a query and selects the most important hidden states as a gold standard for compression. The CCOT module (a LoRA) is trained to predict these key tokens. Then, the DECODE module (another LoRA) is trained on the query plus compressed tokens. During inference, CCOT generates compressed tokens, which DECODE uses to produce concise reasoning steps. Another type of work, summarization-based dynamic reasoning, as mentioned in Section 4.2 explores compressing and summarizing reasoning steps in discrete space during inference, which is similar to the introduction of "contemplation token".

Another work, Heima (Shen et al., 2025a), inspired by Coconut (Hao et al., 2024), brings latent reasoning into Multimodal Large Langue Models (MLLMs). Instead of always using full, lengthy reasoning explanations, Heima replaces each stage of detailed reasoning with a single "thinking token". With this change, the training data is updated. Instead of long textual explanations, each reasoning stage is just one of these thinking tokens. Then, they continue fine-tuning the model to achieve efficient reasoning. Token Assorted (Su et al., 2025) adopts a hybrid approach. During training, part of the CoT is replaced by discrete latent tokens learned via a VQ-VAE (Van Den Oord et al., 2017), and then the LLM is trained with a partial and high-level abstract of the reasoning steps. The authors show that mixing text tokens with latent tokens can facilitate training and inference by representing some reasoning steps in a compact latent form. Other than explicitly compressing the discrete tokens into latent space, Saunshi et al. (2025) demonstrates that looping a k-layer transformer L times can emulate the performance of a kL-layer model. This looping mechanism effectively increases the depth of the model depth without adding parameters, enabling iterative reasoning processes within the latent space. The study reveals that *looped models implicitly generate latent thoughts*, allowing them to simulate multiple steps of CoT reasoning through successive loops.

Training Auxiliary Modules while Keeping LLMs Frozen. While most methods for continuous-space reasoning fine-tune the pre-trained LLM, SoftCoT (Xu et al., 2025c) keeps the underlying LLM frozen. A lightweight auxiliary model generates instance-specific soft thought tokens projected into the embedding space of the frozen LLM. Experiments show that SoftCoT consistently boosts performance, demonstrating the viability of augmenting LLMs with external latent reasoning tokens.

These methods hint at a broader move toward latent reasoning, where critical thinking occurs in compressed, non-textual forms. Such approaches can unlock improved speed, adaptive inference, parallel backtracking, and new ways to interpret or partially reveal the model reasoning. As LLMs grow larger and tasks become more complex, balancing thorough reasoning with computational efficiency is greatly beneficial from these flexible and compact latent CoT paradigms.

4.2 Dynamic Reasoning Paradigm during Inference

Existing works focus on *modifying the reasoning paradigm* for more efficient inference. The key during inference is choosing the proper criterion to guide the reasoning strategy. Current training-free approaches explore dynamic reasoning using various criteria, such as reward-guided, confidence-based, and consistency-based selective reasoning. Additionally, a summarization-based dynamic reasoning method intrinsically integrates the output summarization paradigm of LLMs during training.

💡 ***The key question is: Which criterion to guide the inference? What is the appropriate efficient inference paradigm?***

Table 4: Comparison of different methods of dynamic reasoning paradigm of test time compute during inference.

Category	Method	Training-free?	Baseline and Its Drawbacks	Method Description
Reward-guided Efficient Reasoning	Speculative Rejection (Sun et al., 2024)	Yes	**Best-of-N (BoN) Decoding**: underutilizes GPU memory and computational resources during the early stages, leading to lower final rewards.	Starts BoN with a large initial batch size and rejects unpromising sequences periodically, efficiently achieving higher rewards.
	Reward-Guided Speculative Decoding (RSD) (Liao et al., 2025a)	Yes	**Speculative Decoding**: strictly enforces unbiasedness, discarding useful intermediate outputs and leading to computational inefficiency.	A speculative decoding method that leverages a reward model (PRM) to selectively accept high-quality outputs from a lightweight draft model, reducing computation while preserving accuracy.
Confidence/ Certainty-based Adaptive Reasoning	Dynamic Parallel Tree Search (Ding et al., 2025)	Yes	**Tree-of-Thoughts:** difficult to parallelize due to frequent switching of reasoning focus, and inefficient because of redundant exploration of suboptimal solutions	Dynamically parallelizes node expansion through adaptive batching and implements a search-and-transition mechanism (including *Early Stop* and *Deep Seek*) to prune unpromising paths early.
	Dynasor (Certaindex-based Scheduling) (Fu et al., 2024)	Yes	**Serving systems with uniform resource allocation**: allocate compute uniformly, leading to inefficient resource usage and unmet latency targets	Dynamically allocates compute for reasoning queries based on *Certaindex*, a statistical measure of reasoning progress, to maximize accuracy within resource constraints.
	FastMCTS (Li et al., 2025a)	Yes	**Rejection Sampling**: inefficient, discards intermediate steps, and fails to balance problem difficulty	An MCTS-inspired sampling strategy that efficiently generates high-quality multi-step reasoning data, providing step-level evaluation signals and balanced sampling across problem difficulties.
	Length-filtered Vote (Wu et al., 2025)	Yes	**Majority Voting**: ignores reasoning quality, includes suboptimal CoT lengths, and suffers from noisy predictions	A voting strategy that selects answers with the optimal CoT length by filtering out excessively short or long reasoning paths.
Consistency-based Selective Reasoning	Self-Truncation Best-of-N (ST-BoN) (Wang et al., 2025)	Yes	**Best-of-N Sampling**: fully generates all samples and relies on costly reward models	Estimates the most promising sample early via internal embedding consistency, truncating inferior samples prematurely.
Summarization-based Dynamic Reasoning	LightThinker (Zhang et al., 2025)	No	**Chain-of-Thought (CoT)**: high memory and computational overhead due to the generation of an excessive number of tokens	Trains LLMs to learn when and how to compress intermediate thoughts, condensing long reasoning chains into gist tokens, and uses a sparse-patterned attention mask during inference to enhance computational efficiency.
	InftyThink (Yan et al., 2025)	No	**Monolithic Reasoning**: reasoning output is verbose, and can quickly exceed the context window limit of the LLM, resulting in poor performance	An iterative reasoning paradigm that interleaves reasoning steps with intermediate summarization, enabling unbounded reasoning depth without architectural modifications.

4.2.1 Dynamic Reasoning via Explicit Criteria

Train-time scaling with RL (Guo et al., 2025) can significantly enhance the reasoning ability of LLMs. However, it requires substantial computational resources to scale up the model training, making it prohibitively expensive (Guo et al., 2025). As an alternative, researchers have explored test-time reasoning, also known as test-time scaling (Snell et al., 2024). Instead of relying on training to learn CoT reasoning steps, test-time scaling leverages various inference strategies that allow models to "think longer and broader" on complex problems. This approach consistently improves performance on challenging math and code problems that require reasoning by increasing the computational resources allocated during inference (Snell et al., 2024; Beeching et al.).

Test-time scaling utilizes various inference strategies to generate longer and higher-quality CoT responses. There are several ways to scale up the inference. (1) Best-of-N sampling (Sun et al., 2024; Wang et al., 2025) involves generating multiple responses for a given prompt, expanding the search space to identify better solutions. After generation, the best response is selected using either majority voting, where the most frequently occurring response is chosen; or by a reward model, which evaluates response quality based on pre-defined criteria. This method has been shown to significantly enhance the reasoning capabilities of LLMs (Beeching et al.). (2) Beam-based searching (Ding et al., 2025; Fu et al., 2024; Beeching et al.), which differs from Best-of-N by structuring generation into multiple steps. Instead of generating an entire response in one pass, beam search selects the most promising intermediate outputs with process reward model (Uesato et al., 2022) at each step, while discarding less the optimal ones. This enables a more fine-grained optimization of both response generation and evaluation. (3) Monte Carlo Tree Search (MCTS) (Li et al., 2025a), where multiple solution paths are explored in parallel. MCTS generates partial responses along different branches of a solution tree, evaluates them, and back-propagates reward values to earlier nodes. The model then selects the branch with the highest cumulative reward, ensuring a more refined selection process compared to traditional beam search.

Although test-time scaling can significantly reduce train-time scaling up overhead (Beeching et al.), the large number of generated responses still makes inference computationally expensive. To address this, recent works have been exploring methods to optimize test-time scaling.

Reward-guided Efficient Reasoning. Speculative Rejection (Sun et al., 2024) is an efficient inference-time reasoning algorithm that optimizes Best-of-N (BoN) decoding by dynamically reducing computational overhead (as shown in Figure 8, left). It generates multiple responses until memory limits are nearly reached, then *discards low-quality outputs based on evaluation by a reward model*. This adaptive filtering substantially reduces inference costs compared to vanilla BoN. On the other hand, Reward-Guided Speculative Decoding (RSD) (Liao et al., 2025a) enhances the efficiency of speculative decoding specifically for multi-step reasoning tasks. Unlike traditional speculative decoding methods, which strictly require exact token matching between the draft model and target model, *RSD leverages a Process Reward Model (PRM) to dynamically evaluate intermediate outputs* from the smaller, more efficient draft model. Outputs with high reward scores are directly accepted, while those with lower scores are further refined by a larger, more capable target model.

Confidence/Certainty-based Adaptive Reasoning. Dynamic Parallel Tree Search (DPTS) (Ding et al., 2025) optimizes tree-based reasoning in LLMs by addressing two main inefficiencies by introducing: (1) *Parallelism Streamline* optimizes memory and compute by storing only incremental KV cache updates and dynamically adjusting the number of extended nodes based on available GPU memory, (2) *Search and Transition Mechanism* balances exploration and exploitation using confidence-based criteria. Overall, during inference, the system cuts off uncertain paths to save time. FastMCTS (Li et al., 2025a) is another confidence-based method that aims to optimize multi-step reasoning data synthesis. Traditional rejection sampling generates multiple candidate responses independently, selecting only the correct ones, but it is often inefficient and struggles with imbalanced sampling. Inspired by MCTS, FastMCTS prioritizes high-confidence traces for deep reasoning. Additionally, it adjusts tree expansion based on problem complexity, improving both efficiency and reasoning diversity. Another line of research leverages certainty or uncertainty measures to guide adaptive reasoning. Certaindex (Fu et al., 2024), a certainty metric, quantifies the confidence of LLMs throughout reasoning using semantic entropy, reward model scores, or a combination of both. A higher certaindex indicates that further reasoning steps are unlikely to change the final answer, allowing early termination to free resources for more challenging queries. Dynasor, an inference system built on this principle, optimizes compute scheduling by dynamically tracking reasoning progress instead of allocating resources uniformly. Length-filtered Vote (Wu et al., 2025) is another work that leverages uncertainty to im-

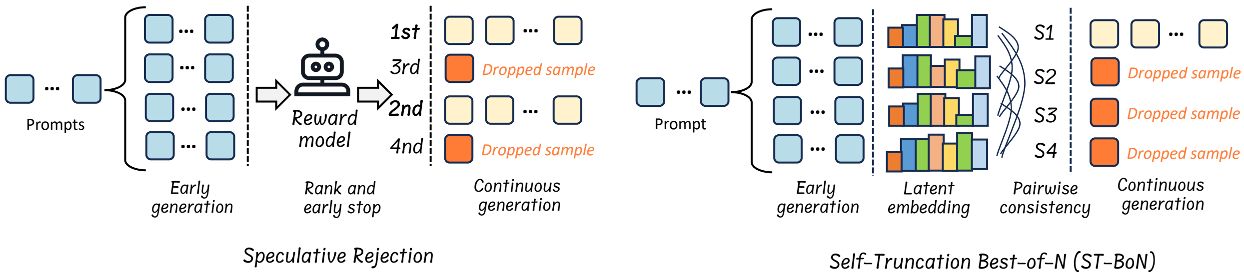

Figure 8: Examples of efficient Best-of-N sampling methods. *(Left) Speculative Rejection* (Sun et al., 2024) uses a reward model to estimate partial generation quality. It then early stops the sampled sequence with lower scores. *(Right) ST-BoN* (Wang et al., 2025) evaluates the latent embedding of the early generation. The latent embedding of each thinking path will be used to calculate pairwise consistency between other tokens. The sequence with the highest consistency is more likely to arrive at the correct answer.

prove CoT reasoning. The study finds that longer reasoning chains do not always improve accuracy; instead, performance initially improves but eventually declines due to error accumulation. The authors provide a mathematical analysis proving the existence of an optimal CoT length, determined by model capability and task difficulty. To exploit this, they propose Length-filtered Voting, a length-aware majority voting method that groups answers by CoT length and selects the most reliable group based on prediction uncertainty.

Consistency-based Selective Reasoning. Self-Truncation Best-of-N (ST-BoN) (Wang et al., 2025) enhances BoN sampling efficiency by introducing early termination (as shown in Figure 8, right), similar to Speculative Rejection (Sun et al., 2024). However, unlike Speculative Rejection using reward models, ST-BoN leverages consistency as the metric to measure the importance. Specifically, it leverages the consistency of latent embeddings to evaluate response quality. The core insight is that "the closer a sample is to others, the more likely its path will lead to the correct answer". Then, ST-BoN selects the most consistent Chain-of-Embedding (CoE) to others and regards it as the optimal sample.

4.2.2 Summarization-based Dynamic Reasoning

Some existing methods choose to optimize reasoning efficiency by training LLMs to *summarize intermediate thinking steps*. LightThinker (Zhang et al., 2025) proposes to train LLMs to learn when and how to compress intermediate reasoning steps. Instead of storing long thought chains, LightThinker compresses verbose reasoning into compact "gist tokens" to reduce memory and computational costs. Implementing this summarization paradigm requires a sparse-patterned attention mask, ensuring the model focuses only on essential compressed representations. InftyThink (Yan et al., 2025) introduces an iterative reasoning method that enables essentially infinite reasoning chains while maintaining strong accuracy without surpassing the context window limit. It achieves this by iteratively generating a thought, summarizing it, and discarding previous thoughts and summaries, retaining only the most recent summary. Additionally, InftyThink provides a technique for converting existing reasoning datasets into an iterative format for training models under this paradigm.

5 Input Prompts-based Efficient Reasoning

From the perspective of input prompts and questions, these works focus on enforcing length constraints or routing LLMs based on the characteristics of input prompts to enable concise and efficient reasoning.

5.1 Prompt-guided Efficient Reasoning

Prompt-guided efficient reasoning *explicitly instructs LLMs to generate fewer reasoning steps*, can be a straightforward and highly effective method for improving the efficiency of reasoning models. As shown in Table 5, different methods propose different prompts to ensure concise reasoning outputs from the model.

Table 5: A summary of prompts used with reasoning models to generate concise reasoning outputs. For further details, refer to Section 5.1.

Method	Prompt
TALE-EP (Han et al., 2024)	**Budget Estimation**: (...) Task: Analyze the given question and estimate the minimum number of tokens required to generate a complete and accurate response. Please give the response by strictly following this format: [[budget]], for example, Budget: [[12]]. **Token-budget-aware CoT**: Please answer the above question. Let's think step by step and use less than <Token-Budget> tokens.
CoD (Xu et al., 2025b)	Think step by step, but only keep a minimum draft for each thinking step, with 5 words at most. Return the answer at the end of the response after a separator ####.
CCoT (Renze & Guven, 2024)	Be concise.
Token Complexity (Lee et al., 2025)	**BulletPoints** (...) only use bullet points. **OnlyNumbers** (...) only use numbers or equations. **NoSpaces** (...) do not use any spaces or line breaks. **NoProperGrammar** (...) do not use proper grammar. **AbbreviateWords** (...) abbreviate words as much as possible. **WordLimit(k)** (...) use at most k words. $k \in \{1, \ldots, 100\}$ **CharLimit(k)** (...) use at most k letters. $k \in \{1, \ldots, 500\}$ **TokenLimit(k)** (...) use at most k tokens. $k \in \{1, \ldots, 500\}$ **StepLimit(k)** (...) use at most k steps. $k \in \{1, \ldots, 5\}$ **ChineseCoT** (...) Respond in Chinese **ChineseCoT(k)** (...) Use at most k Chinese characters. $k \in \{1, \ldots, 500\}$

The key question is: Which prompts can accurately control the reasoning length of LLMs?

Enforcing Concise Reasoning via Varying Prompts. Token-Budget (Han et al., 2024) proposes setting a token budget in prompts to reduce unnecessary reasoning tokens. To optimize efficiency while preserving accuracy, TALE-EP (Han et al., 2024) introduces a training-free, zero-shot method for budget estimation. TALE-EP first estimates a reasonable token budget by prompting the LLM itself. It then incorporates this estimate into a prompt that specifies the token constraint, guiding the LLM to generate a more token-efficient yet accurate response. This work is also categorized in Section 3.2 with further SFT. CoD (Xu et al., 2025b) observes that LLMs often generate excessively verbose reasoning steps, whereas humans typically record only the most essential insights. To enhance reasoning efficiency, they propose Chain-of-Draft prompting. Similar to CoT prompting, CoD encourages step-by-step reasoning but introduces policies to limit verbosity. For instance, their prompt instructs: "*Think step by step, but only keep a minimum draft for each thinking step, with at most five words.*" They find that this approach preserves the necessary intermediate steps while maintaining accuracy, significantly reducing token usage. Lee et al. (2025) systematically studies the relationship between reasoning length and model accuracy across various prompts with explicit compression instructions (e.g., "*use 10 words or less*"). Their analysis reveals a universal trade-off between reasoning length and accuracy, showing that different prompt-based compression strategies align on the same accuracy-compression curve. They hypothesize that each task has an intrinsic *token complexity*, the minimum number of tokens required for successful problem-solving. By computing information-theoretic limits on the accuracy-compression trade-off, they found that existing prompt-based compression methods fall far short of these limits, indicating significant room for improvement. Renze & Guven (2024) introduced Concise Chain-of-Thought (CCoT) prompting, a technique that prompts LLMs to perform step-by-step reasoning while explicitly instructing them to "*be concise.*" MARP (Chen et al., 2024a) introduces modifying prompts to limit single-step computations, effectively refining the reasoning boundary. Further, they increase the per-step computation and decrease global planning steps.

Fine-tuning after Prompting. As noted in Section 3, some approaches collect short CoT data using prompt-based methods, then apply SFT to develop an efficient reasoning model (Han et al., 2024). Beyond performing direct prompt-based reasoning, these fine-tuned models often deliver more promising performance when tackling complex reasoning challenges.

5.2 Prompts Attribute-Driven Reasoning Routing

User-provided prompts can range from easy to difficult tasks. Routing strategies for efficient reasoning dynamically determine how language models handle queries based on their complexity and uncertainty. Ideally, *reasoning models can automatically assign simpler queries to faster but less reasoning-capable LLMs, while directing more complicated queries to slower but stronger reasoning LLMs.*

The key question is: What criterion should be used to determine the attributes (e.g., difficulty) of prompts?

Unknown Criteria. Anthropic releases Claude 3.7 Sonnet (Anthropic, 2023), notable for being the first hybrid reasoning model. Claude 3.7 Sonnet was developed through RL, enabling it to allocate more time to complex reasoning tasks that require deeper analysis, ultimately producing better results. The model offers two response modes: quick answers or step-by-step thinking. Users can leverage API to manage the amount of time the model spends thinking. Although the specifics of the routing criterion remain unclear, Claude 3.7 Sonnet represents the first hybrid reasoning model, setting a foundation for subsequent routing-based large reasoning models.

Training a Classifier. RouteLLM (Ong et al., 2024) trains a query router to dispatch incoming queries to suitable LLMs based on complexity. The authors utilize a substantial amount of preference data collected from Chatbot Arena as training data, enabling effective routing decisions for question-answering and reasoning tasks. Consequently, simpler queries are directed to low-latency LLMs, while complex queries are assigned to higher-latency, more powerful LLMs, significantly accelerating overall reasoning efficiency. Sketch-of-Thought (SoT) (Aytes et al., 2025) leverages routing and prompting to minimize token usage during reasoning. A lightweight DistilBERT-based router dynamically selects the most suitable paradigm based on the characteristics of the questions. Inspired by cognitive science, SoT employs three distinct paradigms: *Conceptual Chaining*, which connects ideas with minimal verbalization; *Chunked Symbolism*, which structures mathematical reasoning into concise symbolic representations; and *Expert Lexicons*, which adopts domain-specific shorthand used by experts.

Uncertainty. Besides relying on additional routers, Self-Ref (Chuang et al., 2025c) enables LLMs to autonomously decide when to route by extracting intrinsic uncertainty scores as self-routing indicators. Specifically, they fine-tune uncertainty-specialized tokens within the LLMs to align uncertainty predictions with prediction correctness in both question-answering and reasoning tasks. This ensures that only uncertain or incorrect outputs trigger routing to more capable LLMs, which decreases the latency of LLM inference. Confident (Chuang et al., 2025a) aims to provide calibrated data for predicting and initializing routing strategies in both LLM question-answering and reasoning tasks without requiring access to user queries. This approach enables more efficient and reliable decision-making in determining whether an LLM should confidently generate an answer or escort the query to a stronger model, ultimately improving reasoning efficiency from a query-level perspective in online LLM service scenarios.

6 Reasoning Abilities via Efficient Training Data and Model Compression

6.1 Training Reasoning Models with Less Data

Improving the efficiency of reasoning models requires optimizing not just the model architecture but also the data used for training. Recent work has shown that carefully selecting, structuring, and leveraging training data can significantly reduce data requirements while maintaining or even improving reasoning performance. Although all approaches focus on efficient data selection, they vary in defining and utilizing efficiency.

💡 ***The key question is: How to construct less but high-quality training data?***

Minimal but High-Impact Data Selection. LIMO (Ye et al., 2025) challenges the conventional belief that complex reasoning tasks require extensive training data. They introduce LIMO, a framework that elicits sophisticated reasoning abilities using minimal but precisely curated examples. By choosing high-quality questions based on *Level of difficulty, Generality, and Knowledge Diversity* and high-quality solutions based on *Optimal Structural Organization, Effective Cognitive Scaffolding, and Rigorous Verification*, with only 817 carefully selected training samples, LIMO can outperform previous models that utilized over 100,000 examples. s1 (Muennighoff et al., 2025) focuses on enhancing reasoning performance by controlling test-time computational resources. They curate a compact dataset based on *Quality*, *Difficulty* and *Diversity*, s1K, comprising 1,000 high-quality questions paired with reasoning traces. Through supervised fine-tuning on this dataset and implementing "budget forcing", which regulates the reasoning duration during inference, s1-32B exceeds OpenAI o1-preview on MATH and AIME24, demonstrating that strategic test time scaling can effectively enhance reasoning capabilities without extensive training data.

Self-Verification as a Data-Efficient Training Signal. S^2R (Ma et al., 2025a) infuse LLMs with self-verification and self-correction abilities through RL. Initially, models are fine-tuned on a curated dataset to establish these capabilities. Subsequently, RL both at the outcome level and the process level is employed to enhance these skills further. With only 3,100 initialization samples, their fine-tuned models consistently improve the performance on reasoning tasks among all base models. S^2R fine-tuned Qwen2.5-Math-7B can outperform models trained on comparable amounts of long CoT distilled data on the MATH500 and GSM8K.

6.2 Reasoning Capabilities of Small Language Models via Distillation and Model Compression

LLMs have demonstrated remarkable reasoning capabilities across various complex tasks, benefiting from their extensive training on diverse datasets. However, their substantial computational and memory demands pose challenges for deployment in resource-constrained environments, such as edge devices, mobile applications, and real-time systems. In scenarios where efficiency, cost, or latency is a primary concern, Small Language Models (SLMs) offer a viable alternative. The ability of SLMs to retain strong reasoning capabilities while operating under strict resource constraints is crucial for expanding the accessibility and practicality of AI-powered reasoning systems. To achieve this, two main categories of approach are explored: Distillation and Model Compression.

💡 ***The key question is: How do small language models perform on reasoning tasks? What impact does model compression (e.g., quantization) have on their reasoning abilities?***

Distillation. Distillation is a crucial technique for transferring the reasoning capabilities of LLMs to SLMs while maintaining efficiency. However, Li et al. (2025c) finds a phenomenon named *Small Model Learnability Gap*, which highlights the challenges of distilling complex reasoning processes from large model to small model, showing that SLMs struggle to emulate the reasoning depth of their larger counterparts. To address this, various approaches have been proposed. Both Li et al. (2025c) and Chenglin et al. (2024) explored mixed distillation, with Li et al. (2025c) blending long and short CoT reasoning examples, while Chenglin et al. (2024) combined CoT and PoT (Program of Thought) to improve the effectiveness of knowledge distillation from LLMs to SLMs on specific tasks. In comparison, Feng et al. (2024) introduced counterfactual distillation, augmenting the training set by masking causal features in the original question, prompting the LLM to complete the masked text, and generating multi-view CoT (positive and negative views) of each data for enhancing the effectiveness of knowledge distillation. In addition, Zhu et al. (2024) developed a feedback-driven distillation technique that iteratively refines distillation datasets. They first prompt an LLM to generate an initial distillation dataset, then expand it by creating diverse and complex questions from existing ones, and finally, this enriched dataset is used to fine-tune SLMs. Another strategy, proposed by Zhao et al. (2024), incorporates probing and retrieval mechanisms into the distillation pipeline. It trains two complementary distilled SLMs, a probing model and a reasoning model, where the probing model retrieves relevant knowledge, which the reasoning model then uses to construct a step-by-step rationale for the answer. Chen et al. (2024b) introduced adaptive thinking during distillation, allowing the models to dynamically adjust reasoning strategies based on the complexity of the task. Furthermore, Liao et al. (2025b) proposed SKIntern, a framework that internalizes symbolic knowledge into SLM to improve CoT reasoning quality and efficiency, while Zhang et al. (2024) introduces SCORE, a pipeline that generates self-correction data from SLMs and fine-tunes the model to function as a self-correcting reasoner. These diverse distillation techniques demonstrate that efficiently transferring reasoning capabilities from LLMs to SLMs requires not only reducing the model size but also carefully and strategically structuring the knowledge transfer process to preserve logical depth and generalization.

Pruning and Quantization. Beyond directly distilling knowledge from LLMs to SLMs, an alternative approach involves compressing an LLM into an SLM using techniques such as quantization and pruning. Srivastava et al. (2025) conducted a comprehensive study analyzing the impact of various model compression techniques on reasoning ability. Their findings reveal that *quantization, which reduces model precision to lower-bit representations, preserves reasoning performance remarkably well*, allowing SLMs to maintain logical coherence and problem-solving capabilities while significantly reducing memory and computational costs.

In contrast, *pruning, which removes specific weights or neurons in the model based on their importance, leads to severe degradation in reasoning quality*, disrupting the model's ability to follow multi-step logical processes. This suggests that compression-based approaches are more effective than training SLMs from scratch, as they allow models to retain reasoning structures inherited from LLMs. However, a critical challenge remains: SLMs often struggle with the instruction following, indicating that compression alone is insufficient. Additional fine-tuning or adaptation methods may be required to align compressed models with user intent and ensure they can effectively interpret and execute complex reasoning tasks.

7 Evaluation and Benchmark

Recent research has introduced innovative benchmarks and evaluation frameworks to systematically assess the reasoning capabilities of LLMs. As LLMs continue to advance in their ability to perform complex reasoning tasks, the need for rigorous, standardized evaluation metrics and frameworks has become increasingly important.

Evaluating Overthinking. Cuadron et al. (2025) introduces a framework to systematically analyze the "overthinking" in LLMs, where models favor extended internal reasoning over necessary environmental interactions. By examining 4,018 trajectories in agentic tasks, the study identified patterns such as Analysis Paralysis, Rogue Actions, and Premature Disengagement. Cuadron et al. (2025) also proposed a novel "overthinking score" and showed a strong correlation between higher scores and decreased task performance. Mitigation strategies such as selecting solutions with lower overthinking scores can improve performance by 30% and at the same time reduce computational overhead by 43%.

Effect of Long CoT Reasoning. Yeo et al. (2025) provides a comprehensive analysis of the mechanism underlying long CoT reasoning. In addition to presenting several key insights, they propose a reward design to enhance the stability of reasoning ability during training and reduce the CoT length, which is also shown in Section 3.1. Jin et al. (2024) reveals a strong relationship between the length of the reasoning chain and the effectiveness of model outputs. Models tend to perform better with extended reasoning steps, suggesting the CoT length is more crucial than accuracy for effective problem-solving.

8 Applications and Discussion

8.1 Applications

Autonomic Driving. Efficient reasoning LLMs are able to greatly improve autonomic driving (Cui et al., 2024; Xing et al., 2025) by helping them understand large amounts of sensor data in a human-like way. They make the cars better at making decisions, so the vehicles can plan for difficult driving situations and react quickly when unexpected events occur. By combining information from cameras, LiDAR, radar, and other sensors, these models help cars drive more safely, choose better routes, and assess risks as they happen. Moreover, because they can explain why they make certain decisions, both passengers and regulators feel more confident in the technology, and the cars can interact more smoothly with smart road systems.

Embodied AI. Efficient reasoning LLMs make embodied AI (Duan et al., 2022) much smarter by helping robots and smart devices understand and react to the world around them. These models process lots of data from cameras, sensors, and other inputs in a way that resembles human thinking. This deep understanding means that a robot can quickly decide the best way to move, handle unexpected changes, and interact safely with people. For example, in a busy factory or a home setting, a robot using these models can navigate obstacles, adjust to new situations, and even explain its actions in simple terms. Altogether, efficient reasoning LLMs boost the reliability, safety, and usefulness of embodied AI systems in daily environments.

Healthcare. Efficient reasoning LLMs would improve healthcare (He et al., 2023) by helping doctors and researchers work with large amounts of medical data more easily. They can quickly analyze patient records, test results, and medical research to spot important trends and patterns that might be hard to see otherwise. This support can lead to faster and more accurate diagnoses, better treatment recommendations, and fewer mistakes. In addition, these models can break down complex medical information into plain language, making it easier for both medical professionals and patients to understand. Generally, efficient reasoning LLMs make healthcare processes smoother and more reliable, leading to better care and outcomes for patients.

8.2 Discussion and Future Directions

Improving Reasoning Ability. From another perspective on efficiency, improving reasoning performance is an important topic (Chen et al., 2025; Sui et al., 2025). To prioritize promising avenues by discarding ineffective strategies early, Meta-Reasoner (Sui et al., 2025) leverages contextual multi-armed bandits for evaluating reasoning progress and selecting the optimal strategy. In each round, the LLM produces a new reasoning step, and the meta-reasoner evaluates its output and generates a progress report, the meta-reasoner uses contextual multi-arm bandit to choose the best guidance strategy for the reasoning step. ITT (Chen et al., 2025) treats each transformer layer as a step in an internal thinking process. By dynamically allocating extra processing to difficult tokens through adaptive routing, ITT enables smaller language models to achieve performance comparable to larger models while using fewer training resources.

Safety of Efficient Reasoning. Safety and efficiency in LLMs often pull in opposite directions, as optimizing one always leads to the performance degradation of the other. When enhancing safety, such as filtering harmful content, mitigating adversarial attacks, and enabling self-correction, the reasoning model typically requires additional computational resources and longer reasoning sequences, leading to increased inference costs and slower response times. Conversely, prioritizing efficiency by minimizing token usage and computational overhead may reduce the reasoning ability to self-reflect, verify its outputs, or defend against adversarial manipulations. This trade-off reflects the well-known principle that there is no “free lunch”, making it crucial to strike a careful balance between safety and efficiency. Kuo et al. (2025) investigates the robustness of safety checks in large CoT reasoning models, revealing severe security flaws in commercial systems. They introduce the malicious-educator benchmark and demonstrate that with their hijacking

Chain-of-Thought (H-CoT) attack, models can drastically reduce their refusal rates, leading to the generation of harmful content. Li et al. (2025b) investigate the safety of long reasoning models. It is observed that while longer outputs enable self-correction and enhance safety, some attack strategies exploit extended generations. They propose a dynamic output length control via an RL-based method to maintain both reasoning quality and security. Balancing safety and efficiency in long reasoning models remains a challenging yet crucial area of investigation.

RL vs. SFT, which is better? When comparing RL (Section 3.1) and SFT (Section 3.2) for creating efficient reasoning language models, the answer is unclear as each method has its own strengths. RL allows a model to learn by trial and error, rewarding it for satisfactory decisions, which can assist it find creative ways to solve problems in new situations. However, this approach can sometimes be unpredictable and require a lot of training. On the other hand, SFT teaches the model using carefully chosen efficient CoT examples constructed by either humans or models, leading to more consistent behavior and easier control. Yet, SFT might struggle when faced with challenges that are not covered in its training data. In practice, combining both methods might be a promising direction and potentially works best because it harnesses the creativity of RL and the reliability of SFT, resulting in a model that is both adaptable and stable.

Evaluation and Benchmark. While previous works have introduced certain evaluation frameworks and benchmarks for efficient reasoning, they still lack a comprehensive and standardized leaderboard for fair comparison. Several challenges hinder advancement in this area: First, different studies often utilize varying model architectures, making direct comparison difficult. Second, the evaluation tasks and datasets used are not consistent across works. Third, disparities in training resources further complicate benchmarking. Therefore, there is a pressing need to establish a comprehensive evaluation framework and public leaderboards that can facilitate transparent, reproducible, and holistic assessment of efficient reasoning methods.

Multimodal Efficient Reasoning. Building on the advances in reasoning with LLMs, recent efforts have extended reasoning techniques to multimodal large language models (MLLMs) Huang et al. (2025) for tackling visually grounded and complex tasks. As computational demands increase with multimodal data, there is a growing need for concise and efficient reasoning within MLLMs. However, optimizing reasoning length and computation in multimodal settings introduces unique challenges, such as modality alignment and cross-modal information fusion. Addressing these issues and developing effective strategies for efficient multimodal reasoning represents a promising and impactful research direction.

Comparisons of Efficient Reasoning Methods. Efficient reasoning methods can be divided into three main categories: model-based, output-based, and input-based approaches. Specifically, the model-based category includes RL-based and SFT-based methods. RL-based approaches often suffer from instability and are expensive to train, while SFT-based methods are generally more stable but require additional computational resources and costs for dataset construction. Output-based methods encompass both latent space and dynamic inference-time computation techniques. The key distinction here is that dynamic inference-time computation is typically training-free. However, these methods are not primarily designed to reduce reasoning length, but rather to decrease inference-time computational requirements or implement paradigms like Best-of-N reasoning. Input-based methods include input-prompt and query-based strategies. Input-prompt methods are advantageous due to their simplicity. Adding a budget-controlling prompt can effectively achieve concise reasoning. Query-based routing, on the other hand, involves integrating specific modules to evaluate the question difficulty, which can be implemented in both training and training-free settings.

Reasoning Length vs. Usefulness and Interpretability. Reducing reasoning length may improve computational efficiency, but it also raises questions about the usefulness and interpretability of the resulting outputs. Shorter reasoning traces can accelerate inference and reduce resource consumption; however, if reasoning steps are overly condensed, they may omit important intermediate justifications, potentially reducing the transparency and trustworthiness of the decisions of models. More systematic investigation is needed to understand how compression impacts the practical utility and clarity of model outputs.

Faithfulness of Efficient Reasoning. Efficient reasoning methods should ideally retain interpretable and logically consistent intermediate steps, ensuring that outputs remain meaningful. Nonetheless, there are observed cases where LLMs generate correct final answers even if some intermediate steps are meaningless. Ensuring that reduced-length reasoning remains both accurate and interpretable is an open research challenge.

8.3 Broader Impacts

Efficient reasoning models hold promise for making advanced LLMs more accessible and sustainable, reducing computational costs and energy consumption. However, as these models become more widely adopted, it is crucial to ensure their transparency, fairness, and robustness, especially in high-stakes applications.

9 Conclusion

This paper provides the first structured survey of efficient reasoning in LLMs, categorizing existing approaches into three areas: model-based, reasoning output-based, and input prompts-based methods. Additionally, it discusses efficient data utilization, reasoning capabilities of smaller models, evaluation techniques, and benchmarking, accompanied by a continuously updated public repository to support future research. Crucially, efficient reasoning approaches offer significant practical benefits across various domains: reducing computational costs in healthcare diagnostics, enhancing real-time decision-making and safety in autonomous driving, boosting the reliability and usefulness of embodied AI systems, and enabling quicker, more profitable responses in financial algorithmic trading and risk assessment. These advancements highlight the broad economic and societal value of efficient reasoning in LLMs.

References

Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*, 2025. 4, 6, 7

Anthropic. Claude 3.7 sonnet, 2023. URL https://www.anthropic.com/news/claude-3-7-sonnet. Accessed: March 10, 2025. 4, 16

Daman Arora and Andrea Zanette. Training language models to reason efficiently. *arXiv preprint arXiv:2502.04463*, 2025. 4, 6, 7

Simon A Aytes, Jinheon Baek, and Sung Ju Hwang. Sketch-of-thought: Efficient llm reasoning with adaptive cognitive-inspired sketching. *arXiv preprint arXiv:2503.05179*, 2025. 4, 16

Edward Beeching, Lewis Tunstall, and Sasha Rush. Scaling test-time compute with open models. URL https://huggingface.co/spaces/HuggingFaceH4/blogpost-scaling-test-time-compute. 13

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17682–17690, 2024. 4

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. 1

Qiguang Chen, Libo Qin, Jiaqi Wang, Jingxuan Zhou, and Wanxiang Che. Unlocking the capabilities of thought: A reasoning boundary framework to quantify and optimize chain-of-thought. *Advances in Neural Information Processing Systems*, 37:54872–54904, 2024a. 4, 16

Xiaoshu Chen, Sihang Zhou, Ke Liang, and Xinwang Liu. Distilling reasoning ability from large language models with adaptive thinking. *arXiv preprint arXiv:2404.09170*, 2024b. 4, 18

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*, 2024c. 2, 5

Yilong Chen, Junyuan Shang, Zhenyu Zhang, Yanxi Xie, Jiawei Sheng, Tingwen Liu, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. Inner thinking transformer: Leveraging dynamic depth scaling to foster adaptive internal thinking. *arXiv preprint arXiv:2502.13842*, 2025. 19

Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*, 2024. 4, 11

Li Chenglin, Qianglong Chen, Liangyue Li, Caiyu Wang, Feng Tao, Yicheng Li, Zulong Chen, and Yin Zhang. Mixed distillation helps smaller language models reason better. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1673–1690, 2024. 4, 17, 18

Yu-Neng Chuang, Leisheng Yu, Guanchu Wang, Lizhe Zhang, Zirui Liu, Xuanting Cai, Yang Sui, Vladimir Braverman, and Xia Hu. Confident or seek stronger: Exploring uncertainty-based on-device llm routing from benchmarking to generalization. *arXiv preprint arXiv:2502.04428*, 2025a. 16

Yu-Neng Chuang, Leisheng Yu, Guanchu Wang, Lizhe Zhang, Zirui Liu, Xuanting Cai, Yang Sui, Vladimir Braverman, and Xia Hu. Confident or seek stronger: Exploring uncertainty-based on-device llm routing from benchmarking to generalization, 2025b. URL https://arxiv.org/abs/2502.04428. 4

Yu-Neng Chuang, Helen Zhou, Prathusha Kameswara Sarma, Parikshit Gopalan, John Boccio, Sara Bolouki, and Xia Hu. Learning to route llms with confidence tokens, 2025c. URL https://arxiv.org/abs/2410.13284. 4, 16

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. 1

Codeforces. Codeforces - competitive programming platform, 2025. URL https://codeforces.com/. Accessed: 2025-03-18. 1

Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pp. 72–83. Springer, 2006. 5

Alejandro Cuadron, Dacheng Li, Wenjie Ma, Xingyao Wang, Yichuan Wang, Siyuan Zhuang, Shu Liu, Luis Gaspar Schroeder, Tian Xia, Huanzhi Mao, Nicholas Thumiger, Aditya Desai, Ion Stoica, Ana Klimovic, Graham Neubig, and Joseph E. Gonzalez. The danger of overthinking: Examining the reasoning-action dilemma in agentic tasks, 2025. URL https://arxiv.org/abs/2502.08235. 4, 18

Can Cui, Yunsheng Ma, Xu Cao, Wenqian Ye, Yang Zhou, Kaizhao Liang, Jintai Chen, Juanwu Lu, Zichong Yang, Kuei-Da Liao, et al. A survey on multimodal large language models for autonomous driving. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 958–979, 2024. 19

Yingqian Cui, Pengfei He, Jingying Zeng, Hui Liu, Xianfeng Tang, Zhenwei Dai, Yan Han, Chen Luo, Jing Huang, Zhen Li, et al. Stepwise perplexity-guided refinement for efficient chain-of-thought reasoning in large language models. *arXiv preprint arXiv:2502.13260*, 2025. 4

Yuntian Deng, Yejin Choi, and Stuart Shieber. From explicit cot to implicit cot: Learning to internalize cot step by step. *arXiv preprint arXiv:2405.14838*, 2024. 10

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 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)*, pp. 4171–4186, 2019. 1

Yifu Ding, Wentao Jiang, Shunyu Liu, Yongcheng Jing, Jinyang Guo, Yingjie Wang, Jing Zhang, Zengmao Wang, Ziwei Liu, Bo Du, et al. Dynamic parallel tree search for efficient llm reasoning. *arXiv preprint arXiv:2502.16235*, 2025. 4, 12, 13

Jiafei Duan, Samson Yu, Hui Li Tan, Hongyuan Zhu, and Cheston Tan. A survey of embodied ai: From simulators to research tasks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(2): 230–244, 2022. 19

Tao Feng, Yicheng Li, Li Chenglin, Hao Chen, Fei Yu, and Yin Zhang. Teaching small language models reasoning through counterfactual distillation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5831–5842, 2024. 4, 18

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*. OpenReview, 2023. 2

Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Aurick Qiao, and Hao Zhang. Efficiently serving llm reasoning programs with certaindex. *arXiv preprint arXiv:2412.20993*, 2024. 4, 12, 13

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *arXiv preprint arXiv:2502.05171*, 2025. 10

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 1

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. 1, 5, 6, 8, 12, 13

Tingxu Han, Chunrong Fang, Shiyu Zhao, Shiqing Ma, Zhenyu Chen, and Zhenting Wang. Token-budget-aware llm reasoning. *arXiv preprint arXiv:2412.18547*, 2024. 2, 4, 9, 10, 15, 16

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024. 2, 4, 11

Kai He, Rui Mao, Qika Lin, Yucheng Ruan, Xiang Lan, Mengling Feng, and Erik Cambria. A survey of large language models for healthcare: from data, technology, and applications to accountability and ethics. *arXiv preprint arXiv:2310.05694*, 2023. 19

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. 1

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022. 9, 10

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025. 20

Mingyu Jin, Qinkai Yu, Dong Shu, Haiyan Zhao, Wenyue Hua, Yanda Meng, Yongfeng Zhang, and Mengnan Du. The impact of reasoning step length on large language models. *arXiv preprint arXiv:2401.04925*, 2024. 4, 19

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. C3ot: Generating shorter chain-of-thought without compromising effectiveness. *arXiv preprint arXiv:2412.11664*, 2024. 4, 8, 9, 10

Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European conference on machine learning*, pp. 282–293. Springer, 2006. 5

Martin Kuo, Jianyi Zhang, Aolin Ding, Qinsi Wang, Louis DiValentin, Yujia Bao, Wei Wei, Da-Cheng Juan, Hai Li, and Yiran Chen. H-cot: Hijacking the chain-of-thought safety reasoning mechanism to jailbreak large reasoning models, including openai o1/o3, deepseek-r1, and gemini 2.0 flash thinking. *arXiv preprint arXiv:2502.12893*, 2025. 19

Ayeong Lee, Ethan Che, and Tianyi Peng. How well do llms compress their own chain-of-thought? a token complexity approach. *arXiv preprint arXiv:2503.01141*, 2025. 4, 15

Peiji Li, Kai Lv, Yunfan Shao, Yichuan Ma, Linyang Li, Xiaoqing Zheng, Xipeng Qiu, and Qipeng Guo. Fastmcts: A simple sampling strategy for data synthesis. *arXiv preprint arXiv:2502.11476*, 2025a. 4, 12, 13

Xuying Li, Zhuo Li, Yuji Kosuga, and Victor Bian. Output length effect on deepseek-r1's safety in forced thinking. *arXiv preprint arXiv:2503.01923*, 2025b. 20

Yuetai Li, Xiang Yue, Zhangchen Xu, Fengqing Jiang, Luyao Niu, Bill Yuchen Lin, Bhaskar Ramasubramanian, and Radha Poovendran. Small models struggle to learn from strong reasoners. *arXiv preprint arXiv:2502.12143*, 2025c. 4, 17, 18

Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, et al. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*, 2025d. 1

Baohao Liao, Yuhui Xu, Hanze Dong, Junnan Li, Christof Monz, Silvio Savarese, Doyen Sahoo, and Caiming Xiong. Reward-guided speculative decoding for efficient llm reasoning. *arXiv preprint arXiv:2501.19324*, 2025a. 4, 12, 13

Huanxuan Liao, Shizhu He, Yupu Hao, Xiang Li, Yuanzhe Zhang, Jun Zhao, and Kang Liu. Skintern: Internalizing symbolic knowledge for distilling better cot capabilities into small language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 3203–3221, 2025b. 4, 18

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100, 2024. 2

Fan Liu, Wenshuo Chao, Naiqiang Tan, and Hao Liu. Bag of tricks for inference-time computation of llm reasoning. *arXiv preprint arXiv:2502.07191*, 2025a. 4

Runze Liu, Junqi Gao, Jian Zhao, Kaiyan Zhang, Xiu Li, Biqing Qi, Wanli Ouyang, and Bowen Zhou. Can 1b llm surpass 405b llm? rethinking compute-optimal test-time scaling, 2025b. URL `https://arxiv.org/abs/2502.06703`. 4

Tengxiao Liu, Qipeng Guo, Xiangkun Hu, Cheng Jiayang, Yue Zhang, Xipeng Qiu, and Zheng Zhang. Can language models learn to skip steps? *arXiv preprint arXiv:2411.01855*, 2024a. 4, 9, 10

Yuliang Liu, Junjie Lu, Zhaoling Chen, Chaofeng Qu, Jason Klein Liu, Chonghan Liu, Zefan Cai, Yunhui Xia, Li Zhao, Jiang Bian, et al. Adaptivestep: Automatically dividing reasoning step through model confidence. *arXiv preprint arXiv:2502.13943*, 2025c. 4

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*, 2024b. 2

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 7

Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning. *arXiv preprint arXiv:2501.12570*, 2025. 2, 4, 5, 6, 7

Ruotian Ma, Peisong Wang, Cheng Liu, Xingyan Liu, Jiaqi Chen, Bang Zhang, Xin Zhou, Nan Du, and Jia Li. S^{2}r: Teaching llms to self-verify and self-correct via reinforcement learning, 2025a. URL https://arxiv.org/abs/2502.12853. 4, 17

Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. Cot-valve: Length-compressible chain-of-thought tuning. *arXiv preprint arXiv:2502.09601*, 2025b. 2, 4, 9, 10

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024. 7, 8

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL https://arxiv.org/abs/2501.19393. 4, 17

Tergel Munkhbat, Namgyu Ho, Seohyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. Self-training elicits concise reasoning in large language models. *arXiv preprint arXiv:2502.20122*, 2025. 4, 9, 10

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms with preference data. *arXiv preprint arXiv:2406.18665*, 2024. 16

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms with preference data, 2025. URL https://arxiv.org/abs/2406.18665. 4

OpenAI. Learning to reason with llms. urlhttps://openai.com/index/learning-to-reason-with-llms/, 2024. Accessed: 15 March 2025. 1, 6

Shubham Parashar, Blake Olson, Sambhav Khurana, Eric Li, Hongyi Ling, James Caverlee, and Shuiwang Ji. Inference-time computations for llm reasoning and planning: A benchmark and insights, 2025. URL https://arxiv.org/abs/2502.12521. 4

Jacob Pfau, William Merrill, and Samuel R Bowman. Let's think dot by dot: Hidden computation in transformer language models. *arXiv preprint arXiv:2404.15758*, 2024. 10

Yuxiao Qu, Matthew YR Yang, Amrith Setlur, Lewis Tunstall, Edward Emanuel Beeching, Ruslan Salakhutdinov, and Aviral Kumar. Optimizing test-time compute via meta reinforcement fine-tuning. *arXiv preprint arXiv:2503.07572*, 2025. 4

Matthew Renze and Erhan Guven. The benefits of a concise chain of thought on problem-solving in large language models. In *2024 2nd International Conference on Foundation and Large Language Models (FLLM)*, pp. 476–483. IEEE, 2024. 4, 15

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J Reddi. Reasoning with latent thoughts: On the power of looped transformers. *arXiv preprint arXiv:2502.17416*, 2025. 4, 11

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 7

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. 7

Xuan Shen, Yizhou Wang, Xiangxi Shi, Yanzhi Wang, Pu Zhao, and Jiuxiang Gu. Efficient reasoning with hidden thinking. *arXiv preprint arXiv:2501.19201*, 2025a. 4, 11

Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai Wang, and Shiguo Lian. Dast: Difficulty-adaptive slow-thinking for large reasoning models. *arXiv preprint arXiv:2503.04472*, 2025b. 4, 6, 8

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. Codi: Compressing chain-of-thought into continuous space via self-distillation. *arXiv preprint arXiv:2502.21074*, 2025c. 4, 11

Luohe Shi, Hongyi Zhang, Yao Yao, Zuchao Li, and Hai Zhao. Keep the cost down: A review on methods to optimize llm's kv-cache consumption. *arXiv preprint arXiv:2407.18003*, 2024. 2

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024. 13

Gaurav Srivastava, Shuxiang Cao, and Xuan Wang. Towards reasoning ability of small language models. *arXiv preprint arXiv:2502.11569*, 2025. 4, 18

DiJia Su, Hanlin Zhu, Yingchen Xu, Jiantao Jiao, Yuandong Tian, and Qinqing Zheng. Token assorted: Mixing latent and text tokens for improved language model reasoning. *arXiv preprint arXiv:2502.03275*, 2025. 4, 11

Yuan Sui, Yufei He, Tri Cao, Simeng Han, and Bryan Hooi. Meta-reasoner: Dynamic guidance for optimized inference-time reasoning in large language models. *arXiv preprint arXiv:2502.19918*, 2025. 19

Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang, Jiahao Qiu, Ming Yin, Mengdi Wang, Peter Bartlett, and Andrea Zanette. Fast best-of-n decoding via speculative rejection. *arXiv preprint arXiv:2410.20290*, 2024. 4, 12, 13, 14

Lin Sun, Guangxiang Zhao, Xiaoqi Jian, Yuhan Wu, Weihong Lin, Yongfu Zhu, Linglin Zhang, Jinzhu Wu, Junfeng Ran, Sai-er Hu, et al. Tinyr1-32b-preview: Boosting accuracy with branch-merge distillation. *arXiv preprint arXiv:2503.04872*, 2025. 4

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025. 4, 5, 7

Qwen Team. Qwq-32b-preview. urlhttps://qwenlm.github.io/blog/qwq-32b-preview/. Accessed: 15 March 2025. 5, 6

Manan Tomar, Lior Shani, Yonathan Efroni, and Mohammad Ghavamzadeh. Mirror descent policy optimization. *arXiv preprint arXiv:2005.09814*, 2020. 7

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

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022. 13

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 11

Guangya Wan, Yuqi Wu, Jie Chen, and Sheng Li. Dynamic self-consistency: Leveraging reasoning paths for efficient llm sampling. *arXiv preprint arXiv:2408.17017*, 2024. 4

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. 4

Yiming Wang, Pei Zhang, Siyuan Huang, Baosong Yang, Zhuosheng Zhang, Fei Huang, and Rui Wang. Sampling-efficient test-time scaling: Self-estimating the best-of-n sampling in early decoding. *arXiv preprint arXiv:2503.01422*, 2025. 4, 12, 13, 14

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. 1, 3, 4

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, et al. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond. *arXiv preprint arXiv:2503.10460*, 2025. 4

Yuyang Wu, Yifei Wang, Tianqi Du, Stefanie Jegelka, and Yisen Wang. When more is less: Understanding chain-of-thought length in llms. *arXiv preprint arXiv:2502.07266*, 2025. 4, 12, 13

Heming Xia, Yongqi Li, Chak Tou Leong, Wenjie Wang, and Wenjie Li. Tokenskip: Controllable chain-of-thought compression in llms. *arXiv preprint arXiv:2502.12067*, 2025. 4, 8, 9, 10

Kun Xiang, Zhili Liu, Zihao Jiang, Yunshuang Nie, Kaixin Cai, Yiyang Yin, Runhui Huang, Haoxiang Fan, Hanhui Li, Weiran Huang, et al. Can atomic step decomposition enhance the self-structured reasoning of multimodal large models? *arXiv preprint arXiv:2503.06252*, 2025. 4

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pp. 38087–38099. PMLR, 2023. 2

Shuo Xing, Chengyuan Qian, Yuping Wang, Hongyuan Hua, Kexin Tian, Yang Zhou, and Zhengzhong Tu. Openemma: Open-source multimodal model for end-to-end autonomous driving. In *Proceedings of the Winter Conference on Applications of Computer Vision*, pp. 1001–1009, 2025. 19

Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, et al. Towards large reasoning models: A survey of reinforced reasoning with large language models. *arXiv preprint arXiv:2501.09686*, 2025a. 1

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*, 2025b. 4, 15

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot: Soft chain-of-thought for efficient reasoning with llms. *arXiv preprint arXiv:2502.12134*, 2025c. 4, 11

Yuchen Yan, Yongliang Shen, Yang Liu, Jin Jiang, Mengdi Zhang, Jian Shao, and Yueting Zhuang. Inftythink: Breaking the length limits of long-context reasoning in large language models. *arXiv preprint arXiv:2503.06692*, 2025. 4, 12, 14

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023. 4

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning, 2025. URL `https://arxiv.org/abs/2502.03387`. 4, 17

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. Demystifying long chain-of-thought reasoning in llms. *arXiv preprint arXiv:2502.03373*, 2025. 2, 4, 6, 7, 18

Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. Distilling system 2 into system 1. *arXiv preprint arXiv:2407.06023*, 2024. 4, 8, 9, 10

Zishun Yu, Tengyu Xu, Di Jin, Karthik Abinav Sankararaman, Yun He, Wenxuan Zhou, Zhouhao Zeng, Eryk Helenowski, Chen Zhu, Sinong Wang, et al. Think smarter not harder: Adaptive reasoning with inference aware optimization. *arXiv preprint arXiv:2501.17974*, 2025. 4

Jiayi Yuan, Hongyi Liu, Shaochen Zhong, Yu-Neng Chuang, Songchen Li, Guanchu Wang, Duy Le, Hongye Jin, Vipin Chaudhary, Zhaozhuo Xu, Zirui Liu, and Xia Hu. KV cache compression, but what must we give in return? a comprehensive benchmark of long context capable approaches. In *Findings of the Association for Computational Linguistics: EMNLP 2024*. Association for Computational Linguistics, November 2024. 2

Jintian Zhang, Yuqi Zhu, Mengshu Sun, Yujie Luo, Shuofei Qiao, Lun Du, Da Zheng, Huajun Chen, and Ningyu Zhang. Lightthinker: Thinking step-by-step compression. *arXiv preprint arXiv:2502.15589*, 2025. 4, 12, 14

Yunxiang Zhang, Muhammad Khalifa, Lajanugen Logeswaran, Jaekyeom Kim, Moontae Lee, Honglak Lee, and Lu Wang. Small language models need strong verifiers to self-correct reasoning. *arXiv preprint arXiv:2404.17140*, 2024. 18

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023. 2

Yichun Zhao, Shuheng Zhou, and Huijia Zhu. Probe then retrieve and reason: Distilling probing and reasoning capabilities into smaller language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 13026–13032, 2024. 4, 18

Xunyu Zhu, Jian Li, Can Ma, and Weiping Wang. Improving mathematical reasoning capabilities of small language models via feedback-driven distillation. *arXiv preprint arXiv:2411.14698*, 2024. 4, 18