A*-Thought: Efficient Reasoning via Bidirectional Compression for Low-Resource Settings

Xiaoang Xu[1] Shuo Wang[2*] Xu Han[2,4,5] Zhenghao Liu[3]
Huijia Wu[1] Peipei Li[1] Zhiyuan Liu[2,4,5] Maosong Sun[2,4,5] Zhaofeng He[1*]

[1]Beijing University of Posts and Telecommunications
[2]Dept. of Comp. Sci. & Tech., Tsinghua University, Beijing, China
[3]Northeastern University [4]Institute for AI, Tsinghua University, Beijing, China
[5]Beijing National Research Center for Information Science and Technology

Abstract

Large Reasoning Models (LRMs) achieve superior performance by extending the thought length. However, a lengthy thinking trajectory leads to reduced efficiency. Most of the existing methods are stuck in the assumption of overthinking and attempt to reason efficiently by compressing the Chain-of-Thought, but this often leads to performance degradation. To address this problem, we introduce A*-Thought, an efficient tree search-based unified framework designed to identify and isolate the most essential thoughts from the extensive reasoning chains produced by these models. It formulates the reasoning process of LRMs as a search tree, where each node represents a reasoning span in the giant reasoning space. By combining the A* search algorithm with a cost function specific to the reasoning path, it can efficiently compress the chain of thought and determine a reasoning path with high information density and low cost. In addition, we also propose a bidirectional importance estimation mechanism, which further refines this search process and enhances its efficiency beyond uniform sampling. Extensive experiments on several advanced math tasks show that A*-Thought effectively balances performance and efficiency over a huge search space. Specifically, A*-Thought can improve the performance of QwQ-32B by $2.39\times$ with low-budget and reduce the length of the output token by nearly 50% with high-budget. The proposed method is also compatible with several other LRMs, demonstrating its generalization capability. The code can be accessed at: `https://github.com/AI9Stars/AStar-Thought`.
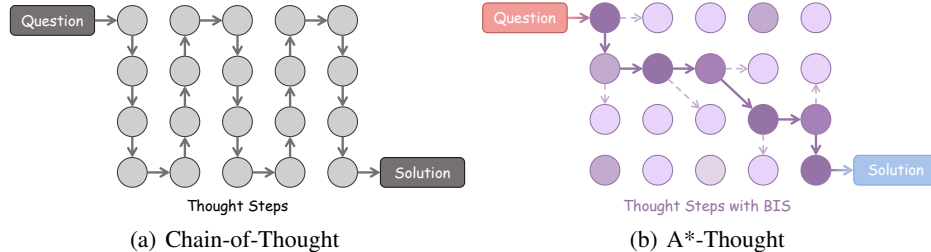
Figure 1: Illustration of the comparison between the standard CoT and the proposed A*-Thought. In A*-Thought, each thinking step is assigned a bidirectional importance score (BIS), represented by varying color shades. Guided by the carefully-designed cost functions, A*-Thought efficiently arrives at the solution using fewer steps, reducing the redundancy inherent in the original CoT.

---

* Corresponding authors.

1 Introduction

Large Reasoning Models (LRMs), such as o1 (OpenAI et al., 2024), R1 (DeepSeek-AI, 2025) and QwQ (Qwen Team, 2025), have demonstrated remarkable advancements in performance on a variety of complex tasks. These significant leaps are largely attributable to their capacity for sophisticated long-form reasoning, often operationalized through the generation of extended chain-of-thought (CoT) sequences. However, this enhanced reasoning capability introduces a substantial trade-off in terms of computational overhead during inference (Xia et al., 2025; Aggarwal & Welleck, 2025). The generation and processing of these lengthy CoTs inherently demand greater storage and computational resources. Consequently, the escalating costs associated with these long CoTs pose a considerable barrier to the practical application and widespread adoption of LRMs, particularly in resource-constrained environments such as on-device or end-point deployments (Hu et al., 2024; Allal et al., 2025) where computational budgets and memory footprints are strictly limited.

To address these efficiency concerns, a growing body of recent research has begun to investigate long-to-short methodologies. These approaches aim to significantly condense the reasoning processes inherent in LRMs, thereby enhancing their inference efficiency (Lee et al., 2025b; Han et al., 2025; Aytes et al., 2025; Xia et al., 2025; Qu et al., 2025; Hao et al., 2024). For example, Chen et al. (2025) identify and conceptualize the overthinking phenomenon in LRMs, where models may engage in unnecessarily verbose or circuitous reasoning. They propose to mitigate this by specifically training the models to favor more concise responses. In a complementary effort, Xiang et al. (2025) introduce a prompting strategy that guides LRMs to articulate their reasoning in shorter, more atomic steps, which cumulatively leads to a reduction in the total length of the generated thought process.

Rather than addressing a singular, narrowly defined issue within LRMs, this work introduces a unified framework designed to identify and isolate the most essential thoughts from the extensive reasoning chains produced by these models. An effective intermediate thinking step should ideally possess two core properties: (1) *High-quality*: The step's logical progression and substantive content must be accurate, sound, and directly relevant to the specific context and boundaries of the question. (2) *Informativeness*: The step must make a demonstrable contribution towards the target solution by furnishing critical information. This characteristic holds even if an intermediate conclusion, while perhaps imperfect or erroneous in isolation, ultimately facilitates the derivation of the correct final answer. The main challenges are two-fold:

> 1. How to identify high-quality and informative steps at the *step-level* ?
> 2. How to effectively assemble individual reasoning steps into a concise and effective reasoning path at the *path-level* ?

We propose A*-Thought, a novel framework for automatically discovering compact and effective reasoning paths by leveraging signals at both step and path levels. At the step level, a bidirectional importance estimation mechanism quantifies the significance of each thinking step based on its relevance to both the question and the prospective solution. At the path level, A* search is employed to efficiently navigate the exponential search space. This search utilizes cost functions that assess two key aspects: the quality of the current path and the conditional self-information of the solution given this path. These assessments collectively inform the estimated current and future cost to reach a desirable final solution. Experimental results demonstrate that A*-Thought successfully learns effective LRMs across diverse inference budgets, surpassing several representative baselines.

The primary contributions of this research are delineated as follows:

- We design a step-level bidirectional importance score (BIS) to evaluate the criticality of individual sentences. This scoring mechanism serves to significantly enhance effectiveness of the A* search procedure compared to standard sampling techniques.

- We introduce a path-level A* search algorithm tailored for compressing lengthy CoTs from LRMs. This algorithm strategically considers both current path quality and estimated future costs to optimize LRM performance under stringent output length constraints.

- The proposed algorithm demonstrates substantial empirical gains over several representative baselines. For instance, for QwQ-32B in concise output scenarios, it achieves a $3.53\times$ improvement in accuracy.
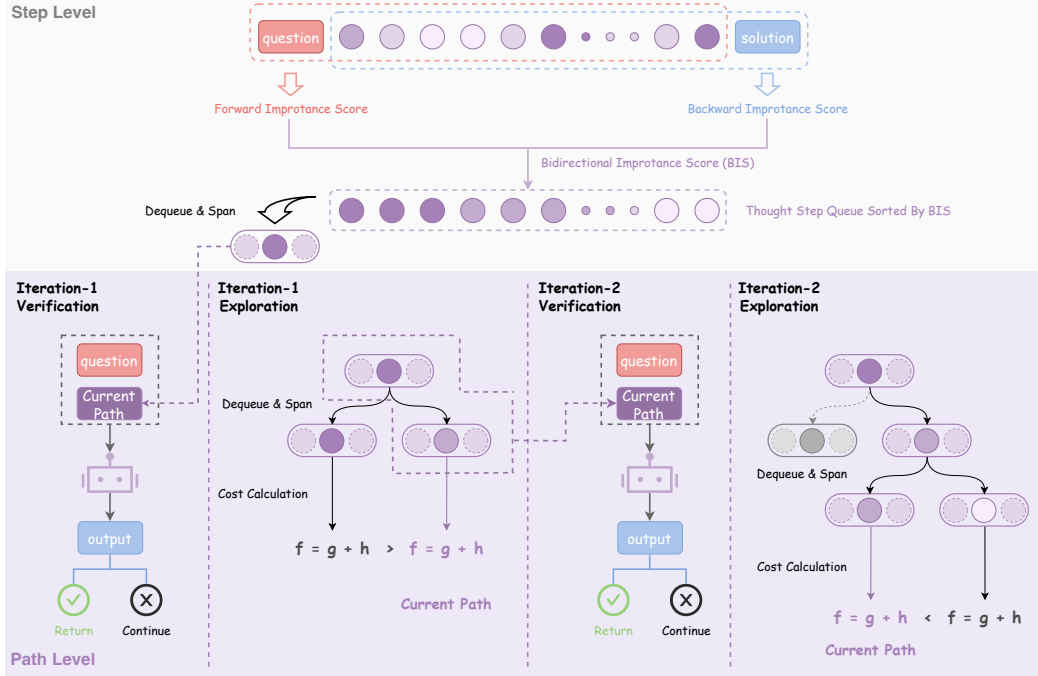
Figure 2: Illustration of A*-Thought, a long-CoT compression method. A*-Thought leverages signals at both the step and path levels. At the step-level, a bidirectional importance score assesses relevance to both the question and the solution. At the path-level, an A* search algorithm is employed, with cost functions designed to consider both current path quality and estimated future cost.

2 Problem Setup

Given an LRM $\mathcal{M}$, it can generate an extended thinking trajectory comprising N reasoning steps, denoted as $\mathbf{t} = \left\{ \mathbf{t}^{(1)}, \mathbf{t}^{(2)}, \ldots, \mathbf{t}^{(N)} \right\}$, and the corresponding solution $\mathbf{s}$ for a given question $\mathbf{q}$. This process can be represented as:

$$(\mathbf{t}, \mathbf{s}) := \mathcal{M}(\mathbf{q}). \tag{1}$$

However, as the number of reasoning steps N increases, the model may frequently switch its thinking modes. Sampling a complete thinking trajectory can hinder convergence of the solution estimation, thereby increasing the inference cost.

Our objective is to identify a subset $\mathbf{t}' \subseteq \mathbf{t}$ that preserves the LRM's reasoning performance to the greatest extent possible. A more compact thinking trajectory $\mathbf{t}'$ is expected to exhibit a higher knowledge density across various computational budgets.

3 Method

In this section, we first introduce the bidirectional importance score of each local thinking step, which is employed in the subsequent A* search process for thought compression.

3.1 Step-Level Bidirectional Importance Score

As mentioned in Section 2, it is prohibitively costly to explore all potential subsets of the long CoT. We therefore propose an importance score to improve sampling efficiency for thought selection. A related approach is LongLLMLingua (Jiang et al., 2024), which employs conditional probabilities to estimate the question-aware importance of each token in a long document. In complex tasks that require long CoTs, the importance of an intermediate thought is not only related to the question but also to the final solution. Relevance to the solution may provide important information for identifying useful thoughts. We therefore propose a bidirectional importance score that considers both the question and the final solution when estimating the importance of each thinking step.

Specifically, we assess importance at two levels, including the attention and model levels. At the attention level, we use attention weights to represent the importance of $\mathbf{x}$ for $\mathbf{y}$:

$$\text{ATTN}(\mathbf{y}|\mathbf{x}) = \frac{1}{H|\mathbf{y}||\mathbf{x}|} \sum_{h=1}^{H} \sum_{j=1}^{|\mathbf{y}|} \sum_{i=1}^{|\mathbf{x}|} a_h\left(y_j, x_i\right), \tag{2}$$

where $a_h(y_j, x_i)$ denotes the attention score of the query y_j to the key x_i for the h-th head. H represents the number of heads. A higher importance score indicates a more significant effect of $\mathbf{x}$ on $\mathbf{y}$. At the model level, we use the negative log-likelihood to assess the importance score of $\mathbf{x}$ for $\mathbf{y}$:

$$\text{NLL}(\mathbf{y}|\mathbf{x}) = -\frac{1}{|\mathbf{y}|} \sum_{j=1}^{|\mathbf{y}|} \log P\left(y_j|\mathbf{x}, \mathbf{y}_{<j}\right). \tag{3}$$

To assess the contribution of each thinking step $\mathbf{t}^{(n)}$ in the overall thought t, we analyze its influence on both the question $\mathbf{q}$ and the solution $\mathbf{s}$. Each thought is concatenated with the question and the solution, forming the sequences $\langle \mathbf{t}^{(n)}, \mathbf{q} \rangle$ and $\langle \mathbf{t}^{(n)}, \mathbf{s} \rangle$, respectively. Subsequently, we utilize a compact language model, specifically GPT-2[2], to quantify attention scores and NLL values. The selection of a smaller model enhances the efficiency of this importance estimation procedure.

Specifically, the bidirectional importance score of $\mathbf{t}^{(n)}$ is denoted as

$$\text{BIS}\left(\mathbf{t}^{(n)}\right) = \frac{(1-\alpha)\,\text{ATTN}\left(\mathbf{q}|\mathbf{t}^{(n)}\right) + \alpha\text{ATTN}\left(\mathbf{s}|\mathbf{t}^{(n)}\right)}{(1-\alpha)\,\text{NLL}\left(\mathbf{q}|\mathbf{t}^{(n)}\right) + \alpha\text{NLL}\left(\mathbf{s}|\mathbf{t}^{(n)}\right)}, \tag{4}$$

where α is a hyper-parameter to control the relative weighting of relevance to the question versus the solution. Figure 3 presents the distribution of BIS for a sequence of thought steps, where the thought steps are divided by "\n\n". The figure illustrates that not all steps hold high importance concerning both the question and the solution. Only a limited subset offers significant contributions. Subsequently, these BIS values will be leveraged in conjunction with the A* search algorithm to determine the pruned thought $\mathbf{t}'$.
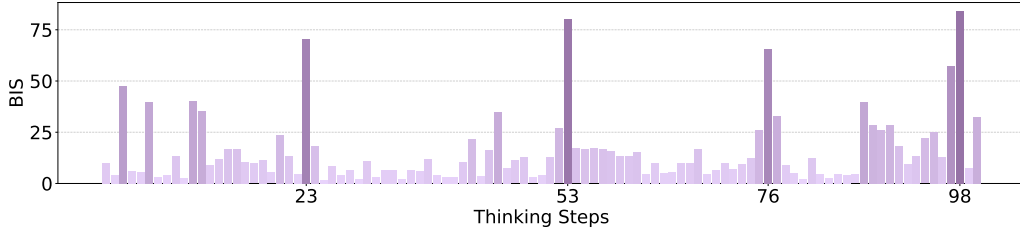

Figure 3: Distribution of BIS values for individual thinking steps in Long CoT.

3.2 Path-Level A* Search

Our objective is to extract an alternative thought trajectory $\mathbf{t}'$ from an original trajectory $\mathbf{t}$. For a trajectory $\mathbf{t}$ with N thinking steps, the 2^N candidate trajectories render exhaustive exploration computationally intractable for extended $\mathbf{t}$. We therefore employ the A* search algorithm for efficient traversal of this space. Following an initialization phase, the algorithm iteratively conducts verification and exploration. In the k-th iteration, a verification model $\mathcal{V}$ ascertains if the current candidate path $\mathbf{t}'_k$ can lead to the correct solution. During exploration, each path $\mathbf{t}'_k$ is evaluated using cost functions. The algorithm is detailed in the following subsections.

3.2.1 Overview

To enhance search efficiency, the thinking steps within the original long CoT $\mathbf{t} = \{\mathbf{t}^{(1)}, \mathbf{t}^{(2)}, \dots, \mathbf{t}^{(N)}\}$ are first sorted in descending order based on their BIS values, yielding

---

[2]`https://huggingface.co/openai-community/gpt2`.

$\mathbf{t}_{\text{sort}} = \left\{ \mathbf{t}^{(n_1)}, \mathbf{t}^{(n_2)}, \ldots, \mathbf{t}^{(n_N)} \right\}$. Subsequently, the A* search algorithm iteratively expands a search tree, denoted as $\mathcal{T}$, according to defined cost functions. In this tree, each node corresponds to a span centered on a specific thinking step, encompassing its immediately preceding and succeeding steps. Formally, a node associated with the thinking step $\mathbf{t}^{(n)}$ is represented as $\mathbf{r}^{(n)} = \langle \mathbf{t}^{(n-1)}, \mathbf{t}^{(n)}, \mathbf{t}^{(n+1)} \rangle$. This approach aims to mitigate the adverse effects of fragmented information that can arise from thought segmentation.

Initialization   By leveraging the bidirectional importance estimation mechanism detailed in Section 3.1, A*-Thought identifies a logical starting point within the potentially redundant thinking trajectory, thereby enhancing both efficiency and performance. Initially, a thought queue, denoted as $\mathcal{Q}$, is constructed using the sorted thought sequence $\mathbf{t}_{\text{sort}}$. Subsequently, the first thought is dequeued from $\mathcal{Q}$ to form the root node of the search tree $\mathcal{T}$. This selection ensures the implementation of a best-first sampling strategy throughout the subsequent search iterations.

Verification   To assess the efficacy of the current thinking path, denoted as $\mathbf{t}'_k$, which encompasses the thinking spans from the root node to the current active leaf node within the search tree $\mathcal{T}$, a validation model $\mathcal{V}$ is introduced. This model is employed to determine whether the current path $\mathbf{t}'_k$ successfully leads to the solution $\mathbf{s}$:

$$\mathcal{V}\left(\mathbf{q} + \mathbf{t}'_k\right) \begin{cases} \neq \mathbf{s}, & \text{expand } \mathcal{T} \\ = \mathbf{s}, & \text{return } \mathbf{t}' = \mathbf{t}'_k \end{cases} \tag{5}$$

It has been observed that verification tends to be ineffective for extremely short thought sequences, thereby offering limited guidance for the search process. Consequently, a lower boundary, denoted as $k_{\min}$, is established for verification. Verification is performed exclusively when the depth of the search tree, k, satisfies the condition $k \geq k_{\min}$.

Exploration   If the current active leaf node does not pass verification, the first W thoughts are dequeued from $\mathcal{Q}$ to function as next-level leaf nodes, denoted as $\{\mathbf{r}_1, \ldots, \mathbf{r}_W\}$. Each of these nodes is then appended to the current thinking path $\mathbf{t}'_k$ to construct a set of candidate thinking paths. we assign a cost function $f(\cdot)$ to each candidate thinking path, where

$$f\left(\mathbf{t}'_k + \mathbf{r}_w\right) = g\left(\mathbf{t}'_k + \mathbf{r}_w\right) + h\left(\mathbf{t}'_k + \mathbf{r}_w\right). \tag{6}$$

The design of our cost function, $f(\cdot)$, is informed by the A* search algorithm. Specifically, $g(\cdot)$ denotes the cumulative cost incurred from the root node to the current node. Concurrently, $h(\cdot)$ functions as a heuristic, providing an estimate of the prospective cost from the current node to the target solution. We select the node that with the minimal cost as the new active leaf node:

$$\hat{\mathbf{r}}_w = \operatorname*{argmin}_{w \in \{1, \cdots, W\}} f\left(\mathbf{t}'_k + \mathbf{r}_w\right). \tag{7}$$

The newly formed active thinking path, $\mathbf{t}'_{k+1} = \langle \mathbf{t}'_k, \hat{\mathbf{r}}_w \rangle$, subsequently proceeds to the next iteration of the process. Figure 2 shows an example for the search process. To prevent an excessively deep search tree, an upper bound, $k_{\max}$, is imposed on its depth. The search process is terminated, and $\mathbf{t}$ is directly returned when the current depth, k, reaches or exceeds this limit, i.e., $k \geq k_{\max}$. The resulting compact responses can be leveraged to distill LRMs, fostering enhanced thinking efficiency.

3.2.2   Design of Cost Functions

To identify an effective and compact thought, denoted as $\mathbf{t}'$, the quality of each intermediate thought $\mathbf{t}'_k$ is assessed from two perspectives: (1) the quality of the current intermediate thought, which is quantified by $g(\cdot)$; and (2) the estimated future cost associated with extending the current intermediate thought $\mathbf{t}'_k$ to the final thought sequence $\mathbf{t}'$.

Current Cost Function   The function $g(\cdot)$ measures the quality of the current intermediate thought $\mathbf{t}'_k$. A verification model leveraging its reasoning capabilities is employed to estimate this quality:

$$g\left(\mathbf{t}'_k\right) = -\frac{\beta}{|\mathbf{t}'_k|} \log P_{\mathcal{V}}\left(\mathbf{t}'_k | \mathbf{q}\right), \tag{8}$$

where β is the weight controlling the effect of the current cost.

5

Future Cost Function The function $h(n)$ estimates the cost from the current node to the goal, thereby influencing the efficiency of the search path. A higher estimated future cost suggests that a more extensive sequence of future thoughts will be required to reach the final solution from the current state. To quantify this, we employ the conditional self-information of the correct solution $\mathbf{s}$, given the current thought $\mathbf{t}_k'$ and the input question $\mathbf{q}$. Formally, it can be represented as:

$$h(\mathbf{t}_k') = \mathcal{I}\left(\mathbf{s}|\mathbf{q}, \mathbf{t}_k'\right). \tag{9}$$

Larger values of the conditional self-information $\mathcal{I}(\cdot)$ indicate a lower likelihood of generating the solution $\mathbf{s}$. We quantify $\mathcal{I}(\cdot)$ as follows:

$$\mathcal{I}\left(\mathbf{s}|\mathbf{q}, \mathbf{t}_k'\right) = -\frac{1}{|\mathbf{s}|} \log P_{\mathcal{V}}\left(\mathbf{s}|\mathbf{q}, \mathbf{t}_k'\right). \tag{10}$$

In particular, A*-Thought enhances reasoning efficiency by compressing the thought trajectory. This is achieved through the systematic reduction of redundant steps, thereby streamlining the path from the initial query to the final solution. Such targeted compression significantly improves the performance of LRMs, enabling them to deliver robust outcomes across diverse budgets.

4 Experiments

4.1 Setup

Backbones To evaluate the performance of A*-thought for compressing long CoT sequences for modern LRMs, we apply several compression methods to representative reasoning models, including QwQ-32B[3], DeepSeek-R1-Distill-Qwen-32B[4], and s1.1-32B[5].

Training Data and Verification Model We utilize the long CoT data released by Muennighoff et al. (2025)[6] as the original CoT data and employ the corresponding distilled model, s1.1-32B, as the verification model, following the approach detailed in Section 3.2.

Benchmarks We employ the following mathematical reasoning tasks in our experiments, all of which demand complex reasoning capabilities from LRMs: MATH500 (Lightman et al., 2023), AMC23 (AMC, 2025), OlympiadBench (He et al., 2024), and GSM8K (Cobbe et al., 2021). Model performance is evaluated using the following metrics:

- *Accuracy*: The proportion of model outputs that match the ground-truth answers, measuring the model's correctness.
- *Length*: The average length (i.e., number of tokens) of the model's response; longer responses typically incur higher inference costs.
- *Accuracy per Computation Unit* (ACU) (Ma et al., 2025b): A metric assessing the trade-off between performance and efficiency, calculated as $\text{ACU} = \text{Accuracy}/\text{Length}$.

Baselines We compare our method against the following baselines:

- *Chain-of-Draft* (CoD) (Xu et al., 2025): A prompt-based method designed to guide LRMs in generating compact reasoning steps, each typically comprising fewer than five words.
- *Break-the-Chain* (BtC) (Ding et al., 2024): A prompt-based method employing specialized prompting strategies to encourage LRMs to utilize shortcuts, thereby enabling them to rapidly explore reasoning clues while bypassing detailed intermediate steps.
- *TokenSkip* (Xia et al., 2025): A training-based method that first employs prompt compression (Jiang et al., 2024) to shorten long CoT data, and then uses this compressed data to train an efficient reasoning model.

For comparison, we also report the performance of the QwQ-32B model directly fine-tuned on the `s1K-1.1` dataset.

---

[3]`https://huggingface.co/Qwen/QwQ-32B`
[4]`https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-32B`
[5]`https://huggingface.co/simplescaling/s1.1-32B`
[6]`https://huggingface.co/datasets/simplescaling/s1K-1.1`

Training Details We trained all models, including training-based baselines and our proposed method, for 3 epochs with a peak learning rate of 1×10^{-5} and a warm-up ratio of 0.1. Training was conducted on 8 NVIDIA A100 80G GPUs, using a per-GPU batch size of 1 and 8 gradient accumulation steps. For our proposed method, the default hyperparameters were set as $\alpha = 0.5$ (Eq. 4) and $\beta = 0.1$ (Eq. 8). The lower bound for the verification depth, $k_{\min}$, is set to 5, while the upper bound for the search tree depth, $k_{\max}$, is set to 20. The exploration size W was set to 2.

Table 1: Experimental results of different long-to-short methods across several benchmarks. The best results are shown in **bold**, and the second-best results are underlined.

Methods	MATH500 Acc.$^{(\uparrow)}$	MATH500 Len.$^{(\downarrow)}$	AMC23 Acc.$^{(\uparrow)}$	AMC23 Len.$^{(\downarrow)}$	OlympiadBench Acc.$^{(\uparrow)}$	OlympiadBench Len.$^{(\downarrow)}$	GSM8K Acc.$^{(\uparrow)}$	GSM8K Len.$^{(\downarrow)}$	Average Acc.$^{(\uparrow)}$	Average Len.$^{(\downarrow)}$	ACU
Budget: 512 Tokens											
QwQ-32B	10.8	512.00	2.5	512.00	3.3	512.00	27.6	511.97	11.1	511.99	2.16
QwQ-32B w/ s1K-1.1	9.6	512.00	7.5	512.00	3.4	512.00	28.8	512.00	12.3	512.00	2.41
+ CoD	10.6	512.00	5.0	512.00	4.2	512.00	29.0	511.96	12.2	511.99	2.38
+ BtC Effective Shortcut	10.2	512.00	12.5	512.00	4.2	512.00	26.7	511.95	13.4	511.99	2.62
+ BtC Skip Steps	9.6	512.00	5.0	512.00	5.6	512.00	28.9	511.95	12.3	511.99	2.40
+ TokenSkip	10.8	511.05	2.5	512.00	3.9	512.00	26.4	508.11	10.9	510.79	2.13
+ A*-Thought	33.2	491.92	15.0	508.60	12.0	509.74	57.4	451.76	29.4	490.51	5.99
Budget: 1024 Tokens											
QwQ-32B	16.6	1016.85	15.0	1024.00	6.4	1023.93	49.1	951.96	21.8	1004.19	2.17
QwQ-32B w/ s1K-1.1	24.8	1023.52	17.5	1024.00	8.9	1023.94	60.1	999.80	27.8	1017.82	2.73
+ CoD	24.8	1023.37	5.0	1024.00	7.3	1023.64	60.1	996.84	24.3	1016.96	2.39
+ BtC Effective Shortcut	23.4	1022.88	7.5	1024.00	7.7	1023.92	61.3	1000.44	25.0	1017.81	2.45
+ BtC Skip Steps	23.4	1023.25	5.0	1024.00	7.6	1024.00	59.9	1000.93	24.0	1018.05	2.36
+ TokenSkip	22.4	995.96	12.5	1024.00	6.4	1019.61	49.7	934.74	22.8	993.58	2.29
+ A*-Thought	50.8	858.28	37.5	928.25	22.3	954.74	81.9	688.69	48.1	857.49	5.61
Budget: 2048 Tokens											
QwQ-32B	51.2	1844.96	25.0	1978.60	18.4	2021.95	80.4	1245.68	43.8	1772.80	2.47
QwQ-32B w/ s1K-1.1	60.0	1887.15	35.0	2000.95	23.3	2012.14	88.7	1474.00	51.8	1843.56	2.81
+ CoD	60.2	1894.54	30.0	2022.35	25.5	2018.02	89.5	1490.23	51.3	1856.29	2.76
+ BtC Effective Shortcut	60.8	1884.67	35.0	2004.65	23.7	2012.43	89.8	1473.25	52.3	1843.75	2.84
+ BtC Skip Steps	58.8	1884.96	35.0	2005.67	23.2	2013.05	89.2	1490.39	51.6	1848.52	2.79
+ TokenSkip	53.6	1685.34	35.0	1923.25	19.7	1943.68	86.7	1272.03	48.8	1706.08	2.86
+ A*-Thought	69.2	1271.76	45.0	1540.30	30.3	1625.89	91.2	843.69	58.9	1320.41	4.46
Budget: 4096 Tokens											
QwQ-32B	75.4	2798.67	55.0	3456.05	36.5	3645.22	85.8	1348.24	63.2	2812.05	2.25
QwQ-32B w/ s1K-1.1	79.6	2693.27	65.0	3485.95	42.4	3500.66	95.2	1624.11	70.6	2826.00	2.50
+ CoD	80.2	2719.00	60.0	3354.28	42.0	3488.67	95.0	1655.80	69.3	2804.44	2.47
+ BtC Effective Shortcut	79.6	2696.72	57.5	3355.43	42.4	3493.28	94.8	1636.45	68.6	2795.47	2.45
+ BtC Skip Steps	80.2	2710.83	57.5	3399.93	41.8	3494.41	94.9	1651.37	68.6	2814.14	2.44
+ TokenSkip	74.4	2336.29	52.5	3156.68	37.8	3289.44	94.8	1412.87	64.9	2548.82	2.55
+ A*-Thought	78.8	1699.78	65.0	2385.85	40.1	2546.45	93.1	874.54	69.3	1876.66	3.69

4.2 Main Results

The detailed experimental results, presented in Table 1, yield the following key insights:

Up to 2.39× accuracy and 2.49× ACU improvements in low-budget scenarios. Specifically, across all examined benchmarks, A*-Thought improve the average accuracy of QwQ-32B from 12.3 to 29.4 when the inference budget is constrained to 512 tokens. Concurrently, the ACU score improves from 2.41 to 5.99. Furthermore, in experiments with inference budgets of 1024 and 2048 tokens, A*-Thought consistently attained superior accuracy and the shortest response lengths.

Up to 33.59% length reduction without substantial accuracy drop in the 4096-token setting. For instance, for the QwQ-32B model, A*-Thought decreased the average response length from 2826.00 to 1876.66 tokens. This significant length reduction resulted in only a slight decrease in average accuracy (from 70.6% to 69.3%). Importantly, A*-Thought also attained the highest ACU score in this setting, outperforming both the prompt-based and the training based baselines.

Compatible with several models, A*-Thought demonstrates generalizability. Figure 4 and Figure 5 display the ACU and performance curves on three distinct backbone models: QwQ-32B, R1-Distill-32B, and s1.1-32B.[7] The results demonstrate A*-Thought's effectiveness across these LRMs, where it achieves the highest efficiency and accuracy under various budget conditions.

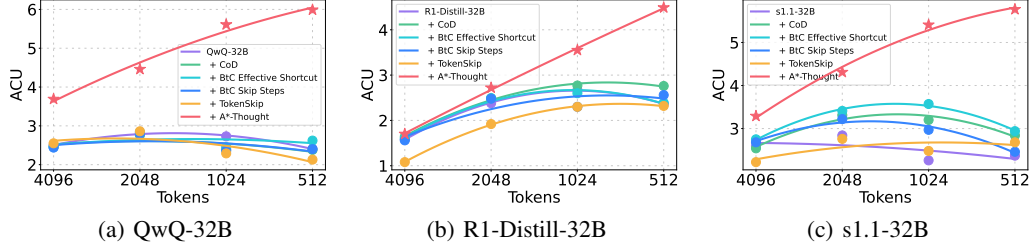[7]Detailed results are provided in Appendix D.

(a) QwQ-32B (b) R1-Distill-32B (c) s1.1-32B

Figure 4: ACU on different methods, which reflects performance-to-efficiency ratio of LRMs.
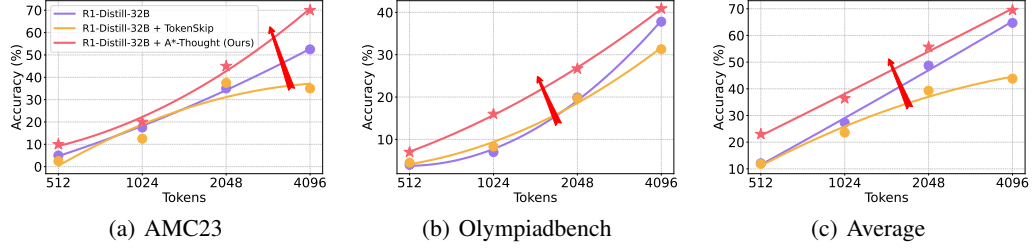

(a) AMC23 (b) Olympiadbench (c) Average

Figure 5: Performance of R1-Distill-32B augmented using TokenSkip and A*-Thought. "Average" denotes the average accuracy of the model in MATH500, AMC23, OlympiadBench, and GSM8K.

5 Analysis

5.1 Performance on More Benchmarks

To verify the generalization capabilities of A*-Thought, we conducted supplementary evaluations on the out-of-domain benchmarks LiveCodeBench and MMLU, as shown in Table 2. Although our model was trained exclusively on mathematical tasks, the results demonstrate that it effectively learned the A*-Thought reasoning pattern. This led to improved performance and higher inference efficiency, even on these non-mathematical tasks.

Table 2: Results of A*-Thought on out-of-domain benchmarks.

Methods	LiveCodeBench		MMLU		Average		ACU
	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	
Budget: 512 Tokens							
QwQ-32B	0.0	512.00	37.6	511.94	18.8	511.97	3.67
+ A*-Thought	**4.5**	**509.53**	**56.7**	**398.57**	**30.6**	**454.05**	**6.74**
Budget: 1024 Tokens							
QwQ-32B	0.0	1021.94	57.4	956.90	28.7	989.42	2.90
+ A*-Thought	**11.8**	**986.02**	**71.9**	**573.30**	**41.9**	**779.66**	**5.37**
Budget: 2048 Tokens							
QwQ-32B	3.5	1977.58	75.2	1323.26	39.4	1650.42	2.38
+ A*-Thought	**24.5**	**1734.28**	**79.0**	**671.41**	**51.8**	**1202.85**	**4.30**
Budget: 4096 Tokens							
QwQ-32B	12.0	3586.93	79.5	1584.05	45.8	2585.49	1.77
+ A*-Thought	**39.0**	**3044.51**	**80.3**	**733.90**	**59.7**	**1889.21**	**3.16**

5.2 Analysis on the Effect of A* Search

Table 3 presents an ablation study on the value of k_{min} in the A*-search process. The results show that with $k_{min} = 15$ and a 4K-token budget, the A*-Thought model consistently outperformed the strongest baseline across all benchmarks. Importantly, this performance was also achieved with higher token efficiency during inference.

Table 3: Effect of the exploration step limit k_{min} on model performance.

Methods	MATH500		AMC23		OlympiadBench		GSM8K		Average		ACU
	Acc.(↑)	Len.(↓)	Acc.(↑)	Len.(↓)	Acc.(↑)	Len.(↓)	Acc.(↑)	Len.(↓)	Acc.(↑)	Len.(↓)	
Budget: 4096 Tokens											
QwQ-32B	75.4	2798.67	55.0	3456.05	36.5	3645.22	85.8	1348.24	63.2	2812.05	2.25
QwQ-32B w/ s1K-1.1	79.6	2693.27	65.0	3485.95	42.4	3500.66	95.2	1624.11	70.6	2826.00	2.50
+ A*-Thought $k_{min} = 5$	78.8	**1699.78**	65.0	**2385.85**	40.1	**2546.45**	93.1	**874.54**	69.3	**1876.66**	**3.69**
+ A*-Thought $k_{min} = 15$	**80.8**	2184.34	**67.5**	2893.68	**44.2**	3063.96	**95.5**	1229.40	**72.0**	2342.85	3.07

5.3 Training Loss and Training Time

Figure 6 illustrates the training loss for various training-based methods, demonstrating that A*-Thought achieves the lowest loss. This potential advantage may be attributed to A*-Thought's utilization of the span around individual thoughts, thereby reducing the negative impact of interrupting the complete thinking process. Furthermore, by considering the quality of the intermediate path during the A* search process, we ensure the learnability of the compressed data.

(a) Fine-Tuning	(b) TokenSkip	(c) A*-Thought
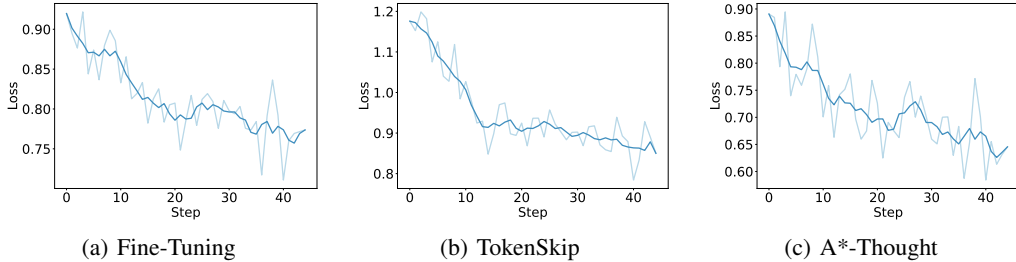
Figure 6: Training loss of the training-based methods discussed, using the QwQ-32B backbone.

Table 4 presents a comparison of compressed data sizes and their corresponding training times. Notably, A*-Thought exhibits a significantly higher compression ratio than TokenSkip, while achieving a lower training loss. These results further corroborate the efficacy of the proposed A*-Thought method. A more detailed ablation study on the components of A*-Thought, along with an analysis of the effect of hyperparameters, is presented in the Appendix.

Table 4: Amount of the training data and the corresponding time. ρ denotes the compression rate.

Training Set	ρ	Training Time (s)		
		QwQ-32B	R1-Distill-32B	s1.1-32B
s1K-1.1	100.00%	13819.60	13784.32	13638.07
+ TokenSkip	85.41% (↓14.59%)	12846.14 (↓7.04%)	12700.96 (↓7.86%)	12679.52 (↓7.03%)
+ A*-Thought	31.31% (↓68.69%)	10468.20 (↓24.25%)	10809.50 (↓21.58%)	10550.03 (↓22.64%)

5.4 Case Study

Figure 7 showcases a representative example of outputs from models trained with and without A*-Thought, given an identical question. While both models arrive at the correct solution, A*-Thought fosters a more streamlined and focused thinking path. Appendix G provides further comprehensive examples. These examples demonstrate that A*-Thought can train models to adopt a more concise thinking process. We believe this approach, with additional effort and integration with advanced training techniques, holds the potential to break through the current bottlenecks in LRMs.

Mikka wants to order a pizza with two toppings. He has 8 different toppings to choose from. How many different pizzas could he order?

Okay, so Mikka wants to order a pizza with two different toppings, and there are 8 different toppings available. I need to figure out how many different pizzas he could order. Hmm, let me think.

First, when the problem says "two different toppings," that means Mikka is choosing two distinct toppings from the 8 available. The question is asking about the number of different combi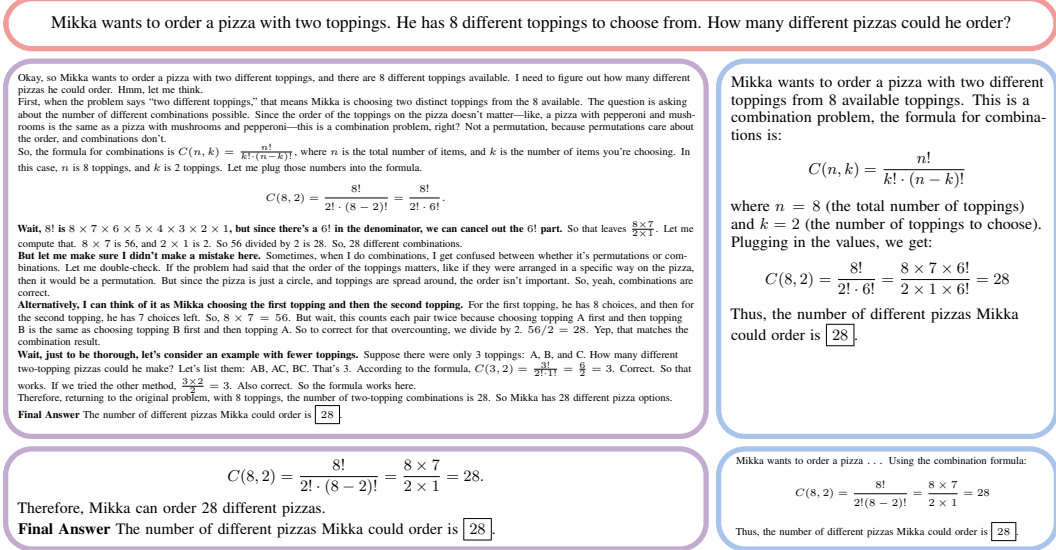nations possible. Since the order of the toppings on the pizza doesn't matter—like, a pizza with pepperoni and mushrooms is the same as a pizza with mushrooms and pepperoni—this is a combination problem, right? Not a permutation, because permutations care about the order, and combinations don't.

So, the formula for combinations is $C(n, k) = \frac{n!}{k! \cdot (n-k)!}$, where n is the total number of items, and k is the number of items you're choosing. In this case, n is 8 toppings, and k is 2 toppings. Let me plug those numbers into the formula.

$$C(8, 2) = \frac{8!}{2! \cdot (8-2)!} = \frac{8!}{2! \cdot 6!}.$$

Wait, 8! is $8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1$, **but since there's a** 6! **in the denominator, we can cancel out the** 6! **part.** So that leaves $\frac{8 \times 7}{2 \times 1}$. Let me compute that. 8×7 is 56, and 2×1 is 2. So 56 divided by 2 is 28. So, 28 different combinations.

But let me make sure I didn't make a mistake here. Sometimes, when I do combinations, I get confused between whether it's permutations or combinations. Let me double-check. If the problem had said that the order of the toppings matters, like if they were arranged in a specific way on the pizza, then it would be a permutation. But since the pizza is just a circle, and toppings are spread around, the order isn't important. So, yeah, combinations are correct.

Alternatively, I can think of it as Mikka choosing the first topping and then the second topping. For the first topping, he has 8 choices, and then for the second topping, he has 7 choices left. So, $8 \times 7 = 56$. But wait, this counts each pair twice because choosing topping A first and then topping B is the same as choosing topping B first and then topping A. So to correct for that overcounting, we divide by 2. $56/2 = 28$. Yep, that matches the combination result.

Wait, just to be thorough, let's consider an example with fewer toppings. Suppose there were only 3 toppings: A, B, and C. How many different two-topping pizzas could he make? Let's list them: AB, AC, BC. That's 3. According to the formula, $C(3, 2) = \frac{3!}{2! \cdot 1!} = \frac{6}{2} = 3$. Correct. So that works. If we tried the other method, $\frac{3 \times 2}{2} = 3$. Also correct. So the formula works here.

Therefore, returning to the original problem, with 8 toppings, the number of two-topping combinations is 28. So Mikka has 28 different pizza options.

Final Answer The number of different pizzas Mikka could order is $\boxed{28}$.

Mikka wants to order a pizza with two different toppings from 8 available toppings. This is a combination problem, the formula for combinations is:

$$C(n, k) = \frac{n!}{k! \cdot (n-k)!}$$

where $n = 8$ (the total number of toppings) and $k = 2$ (the number of toppings to choose). Plugging in the values, we get:

$$C(8, 2) = \frac{8!}{2! \cdot 6!} = \frac{8 \times 7 \times 6!}{2 \times 1 \times 6!} = 28$$

Thus, the number of different pizzas Mikka could order is $\boxed{28}$.

$$C(8, 2) = \frac{8!}{2! \cdot (8-2)!} = \frac{8 \times 7}{2 \times 1} = 28.$$

Therefore, Mikka can order 28 different pizzas.
Final Answer The number of different pizzas Mikka could order is $\boxed{28}$.

Mikka wants to order a pizza . . . Using the combination formula:

$$C(8, 2) = \frac{8!}{2!(8-2)!} = \frac{8 \times 7}{2 \times 1} = 28$$

Thus, the number of different pizzas Mikka could order is $\boxed{28}$.

Figure 7: Responses generated by QwQ-32B models trained with and without A*-Thought. Red box represents the question, purple box represents the thinking process, blue box represents the solution.

6 Related Works

Recent LRMs achieve complex reasoning capabilities, often described as "deep thinking", through significant computational effort during inference, a process sometimes referred to as test-time scaling. However, studies indicate that models utilizing long CoT processes can be susceptible to excessive computation, leading to inefficiencies often termed "overthinking". Consequently, considerable research has aimed to improve reasoning efficiency. Primary strategies include the development of refined prompting techniques (Renze & Guven, 2024; Ding et al., 2024; Han et al., 2025; Lee et al., 2025a; Aytes et al., 2025; Xu et al., 2025; Ma et al., 2025a) and methods for CoT compression (Hao et al., 2024; Cheng & Durme, 2024; Shen et al., 2025; Zhang et al., 2025; Su et al., 2025).

Specific approaches have targeted token-level or path-level optimizations. For example, Token-Skip (Xia et al., 2025) attempts to focus computation by selectively processing tokens deemed most relevant to the input query while omitting less pertinent ones. Another method, Retro-Search (Lu et al., 2025), inspired by Monte Carlo Tree Search, explores numerous potential reasoning paths within long CoT structures. Its goal is to mitigate redundancy across multiple paths (overthinking) while ensuring adequate exploration within individual paths (avoiding underthinking), issues potentially exacerbated by frequent shifts in reasoning strategy.

In contrast to methods focusing on specific causes of redundancy within long CoT, this paper proposes a heuristic approach. We employ a bidirectional importance score to guide the A* search algorithm. The objective is to identify a computationally efficient yet effective reasoning pathway, thereby contributing to the development of more resource-efficient LRMs.

7 Conclusion

We introduce A*-Thought, an innovative CoT compression algorithm designed to enhance the reasoning efficiency of contemporary LRMs. Our approach meticulously crafts concise reasoning pathways. This is achieved by first evaluating the necessity of individual reasoning steps using a novel bidirectional importance score. Subsequently, the thinking steps are assembled into a compact and coherent reasoning chain by employing a path-level A* search algorithm. A key aspect of our A* search process is the design of specialized cost functions. These functions assess not only the quality of the current partial reasoning path but also estimate the potential cost to complete the thought process, thereby providing helpful guidance for selecting the most promising reasoning trajectories. Experimental results demonstrate that A*-Thought can outperform several representative baselines.

Acknowledgments and Disclosure of Funding

References

Aggarwal, P. and Welleck, S. L1: Controlling how long a reasoning model thinks with reinforcement learning, March 2025.

Allal, L. B., Lozhkov, A., Bakouch, E., Blázquez, G. M., Penedo, G., Tunstall, L., Marafioti, A., Kydlíček, H., Lajarín, A. P., Srivastav, V., Lochner, J., Fahlgren, C., Nguyen, X.-S., Fourrier, C., Burtenshaw, B., Larcher, H., Zhao, H., Zakka, C., Morlon, M., Raffel, C., von Werra, L., and Wolf, T. Smollm2: When smol goes big – data-centric training of a small language model, 2025. URL https://arxiv.org/abs/2502.02737.

AMC. American mathematics competitions (amc), 2025. URL https://maa.org/student-programs/amc/.

Aytes, S. A., Baek, J., and Hwang, S. J. Sketch-of-thought: Efficient llm reasoning with adaptive cognitive-inspired sketching, March 2025.

Chen, X., Xu, J., Liang, T., He, Z., Pang, J., Yu, D., Song, L., Liu, Q., Zhou, M., Zhang, Z., Wang, R., Tu, Z., Mi, H., and Yu, D. Do not think that much for 2+3=? on the overthinking of o1-like llms, 2025. URL https://arxiv.org/abs/2412.21187.

Cheng, J. and Durme, B. V. Compressed chain of thought: Efficient reasoning through dense representations, December 2024.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, January 2025.

Ding, M., Liu, H., Fu, Z., Song, J., Xie, W., and Zhang, Y. Break the chain: Large language models can be shortcut reasoners, June 2024.

Han, T., Wang, Z., Fang, C., Zhao, S., Ma, S., and Chen, Z. Token-budget-aware llm reasoning, February 2025.

Hao, S., Sukhbaatar, S., Su, D., Li, X., Hu, Z., Weston, J., and Tian, Y. Training large language models to reason in a continuous latent space, December 2024.

He, C., Luo, R., Bai, Y., Hu, S., Thai, Z., Shen, J., Hu, J., Han, X., Huang, Y., Zhang, Y., Liu, J., Qi, L., Liu, Z., and Sun, M. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.211.

Hu, S., Tu, Y., Han, X., He, C., Cui, G., Long, X., Zheng, Z., Fang, Y., Huang, Y., Zhao, W., et al. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024.

Jiang, H., Wu, Q., Luo, X., Li, D., Lin, C.-Y., Yang, Y., and Qiu, L. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1658–1677, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.91.

Lee, A., Che, E., and Peng, T. How well do llms compress their own chain-of-thought? a token complexity approach, March 2025a.

Lee, A., Che, E., and Peng, T. How well do llms compress their own chain-of-thought? a token complexity approach, 2025b. URL `https://arxiv.org/abs/2503.01141`.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step, 2023. URL `https://arxiv.org/abs/2305.20050`.

Lu, X., Han, S., Acuna, D., Kim, H., Jung, J., Prabhumoye, S., Muennighoff, N., Patwary, M., Shoeybi, M., Catanzaro, B., and Choi, Y. Retro-search: Exploring untaken paths for deeper and efficient reasoning, April 2025.

Ma, W., He, J., Snell, C., Griggs, T., Min, S., and Zaharia, M. Reasoning models can be effective without thinking. https://arxiv.org/abs/2504.09858v1, April 2025a.

Ma, X., Wan, G., Yu, R., Fang, G., and Wang, X. Cot-valve: Length-compressible chain-of-thought tuning, February 2025b.

Muennighoff, N., Yang, Z., Shi, W., Li, X. L., Fei-Fei, L., Hajishirzi, H., Zettlemoyer, L., Liang, P., Candès, E., and Hashimoto, T. S1: Simple test-time scaling, February 2025.

OpenAI, :, Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., Iftimie, A., Karpenko, A., Passos, A. T., Neitz, A., Prokofiev, A., Wei, A., Tam, A., Bennett, A., Kumar, A., Saraiva, A., Vallone, A., Duberstein, A., Kondrich, A., Mishchenko, A., Applebaum, A., Jiang, A., Nair, A., Zoph, B., Ghorbani, B., Rossen, B., Sokolowsky, B., Barak, B., McGrew, B., Minaiev, B., Hao, B., Baker, B., Houghton, B., McKinzie, B., Eastman, B., Lugaresi, C., Bassin, C., Hudson, C., Li, C. M., de Bourcy, C., Voss, C., Shen, C., Zhang, C., Koch, C., Orsinger, C., Hesse, C., Fischer, C., Chan, C., Roberts, D., Kappler, D., Levy, D., Selsam, D., Dohan, D., Farhi, D., Mely, D., Robinson, D., Tsipras, D., Li, D., Oprica, D., Freeman, E., Zhang, E., Wong, E., Proehl, E., Cheung, E., Mitchell, E., Wallace, E., Ritter, E., Mays, E., Wang, F., Such, F. P., Raso, F., Leoni, F., Tsimpourlas, F., Song, F., von Lohmann, F., Sulit, F., Salmon, G., Parascandolo, G., Chabot, G., Zhao, G., Brockman, G., Leclerc, G., Salman, H., Bao, H., Sheng, H., Andrin, H., Bagherinezhad, H., Ren, H., Lightman, H., Chung, H. W., Kivlichan, I., O'Connell, I., Osband, I., Gilaberte, I. C., Akkaya, I., Kostrikov, I., Sutskever, I., Kofman, I., Pachocki, J., Lennon, J., Wei, J., Harb, J., Twore, J., Feng, J., Yu, J., Weng, J., Tang, J., Yu, J., Candela, J. Q., Palermo, J., Parish, J., Heidecke, J., Hallman, J., Rizzo, J., Gordon, J., Uesato, J., Ward, J., Huizinga, J., Wang, J., Chen, K., Xiao, K., Singhal, K., Nguyen, K., Cobbe, K., Shi, K., Wood, K., Rimbach, K., Gu-Lemberg, K., Liu, K., Lu, K., Stone, K., Yu, K., Ahmad, L., Yang, L., Liu, L., Maksin, L., Ho, L., Fedus, L., Weng, L., Li, L., McCallum, L., Held, L., Kuhn, L., Kondraciuk, L., Kaiser, L., Metz, L., Boyd, M., Trebacz, M., Joglekar, M., Chen, M., Tintor, M., Meyer, M., Jones, M., Kaufer, M., Schwarzer, M., Shah, M., Yatbaz, M., Guan, M. Y., Xu, M., Yan, M., Glaese, M., Chen, M., Lampe, M., Malek, M., Wang, M., Fradin, M., McClay, M., Pavlov, M., Wang, M., Wang, M., Murati, M., Bavarian, M., Rohaninejad, M., McAleese, N., Chowdhury, N., Chowdhury, N., Ryder, N., Tezak, N., Brown, N., Nachum, O., Boiko, O., Murk, O., Watkins, O., Chao, P., Ashbourne, P., Izmailov, P., Zhokhov, P., Dias, R., Arora, R., Lin, R., Lopes, R. G., Gaon, R., Miyara, R., Leike, R., Hwang, R., Garg, R., Brown, R., James, R., Shu, R., Cheu, R., Greene, R., Jain, S., Altman, S., Toizer, S., Toyer, S., Miserendino, S., Agarwal, S., Hernandez, S., Baker, S., McKinney, S., Yan, S., Zhao, S., Hu, S., Santurkar, S., Chaudhuri, S. R., Zhang, S., Fu, S., Papay, S., Lin, S., Balaji, S., Sanjeev, S., Sidor, S., Broda, T., Clark, A., Wang, T., Gordon, T., Sanders, T., Patwardhan, T., Sottiaux, T., Degry, T., Dimson, T., Zheng, T., Garipov, T., Stasi, T., Bansal, T., Creech, T., Peterson, T., Eloundou, T., Qi, V., Kosaraju, V., Monaco, V., Pong, V., Fomenko, V., Zheng, W., Zhou, W., McCabe, W., Zaremba, W., Dubois, Y., Lu, Y., Chen, Y., Cha, Y., Bai, Y., He, Y., Zhang, Y., Wang, Y., Shao, Z., and Li, Z. Openai o1 system card, 2024. URL `https://arxiv.org/abs/2412.16720`.

Qu, Y., Yang, M. Y. R., Setlur, A., Tunstall, L., Beeching, E. E., Salakhutdinov, R., and Kumar, A. Optimizing test-time compute via meta reinforcement fine-tuning, March 2025.

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025. URL `https://qwenlm.github.io/blog/qwq-32b/`.

Renze, M. and Guven, E. 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, November 2024. doi: 10.1109/FLLM63129.2024.10852493.

Shen, Z., Yan, H., Zhang, L., Hu, Z., Du, Y., and He, Y. Codi: Compressing chain-of-thought into continuous space via self-distillation, February 2025.

Su, D., Zhu, H., Xu, Y., Jiao, J., Tian, Y., and Zheng, Q. Token assorted: Mixing latent and text tokens for improved language model reasoning, February 2025.

Xia, H., Li, Y., Leong, C. T., Wang, W., and Li, W. Tokenskip: Controllable chain-of-thought compression in llms, February 2025.

Xiang, K., Liu, Z., Jiang, Z., Nie, Y., Cai, K., Yin, Y., Huang, R., Fan, H., Li, H., Huang, W., Zeng, Y., Yuan, Y.-J., Han, J., Hong, L., Xu, H., and Liang, X. Can atomic step decomposition enhance the self-structured reasoning of multimodal large models?, March 2025.

Xu, S., Xie, W., Zhao, L., and He, P. Chain of draft: Thinking faster by writing less, March 2025.

Zhang, J., Zhu, Y., Sun, M., Luo, Y., Qiao, S., Du, L., Zheng, D., Chen, H., and Zhang, N. Lightthinker: Thinking step-by-step compression, February 2025.

A A*-Thought Algorithm Detail

Algorithm 1 shows the details of the A* search algorithm to compress lengthy CoTs.

Algorithm 1 A*-Thought algorithm for compressing lengthy CoTs

---

1: **Input:**
2:     $\mathbf{q}$: question; $\mathbf{t}$: original CoT; $\mathbf{t}_{\mathrm{sort}}$: thought list sorted by BIS; $\mathbf{s}$: solution; $\mathcal{V}$: verification model; $k_{\min}$: min verification depth; $k_{\max}$: max search depth; W: number of observable nodes
3: **Output:**
4:     $\mathbf{t}' \subseteq \mathbf{t}$: compressed thought path
5: **procedure** A* SEARCH

                                                 ▷ (1) Initialization

6:         $\mathcal{Q} \leftarrow \mathbf{t}_{\mathrm{sort}}$
7:         $\mathbf{t}'_k \leftarrow \mathcal{Q}.pop()$
8:         $k = 1$
9:         **while** $\mathcal{Q}$ not empty **do**
10:             **if** $k \geq k_{\max}$ **then**
11:                 **return** $\mathbf{t}' = \mathbf{t}$
12:             **end if**

                                                 ▷ (2) Verification

13:             **if** $k \geq k_{\min}$ **and** $\mathcal{V}(\mathbf{q} + \mathbf{t}'_k) == \mathbf{s}$ **then**
14:                 **return** $\mathbf{t}' = \mathbf{t}'_k$
15:             **end if**

                                                 ▷ (3) Exploration

16:             $\{\mathbf{r}_1, \ldots, \mathbf{r}_W\} \leftarrow$ first W elements in $\mathcal{Q}$
17:             **for** $\mathbf{r}_w$ in $\{\mathbf{r}_1, \ldots, \mathbf{r}_W\}$ **do**
18:                 $f(\mathbf{t}'_k + \mathbf{r}_w) = g(\mathbf{t}'_k + \mathbf{r}_w) + h(\mathbf{t}'_k + \mathbf{r}_w)$
19:             **end for**
20:             $\hat{\mathbf{r}}_w = \mathrm{argmin}_{w \in \{1, \cdots, W\}} f(\mathbf{t}'_k + \mathbf{r}_w)$
21:             $\mathbf{t}'_{k+1} = \langle \mathbf{t}'_k, \hat{\mathbf{r}}_w \rangle$
22:             $\mathcal{Q}.pop(\hat{\mathbf{r}}_w)$
23:             $k = k + 1$
24:         **end while**
25:         **return** $\mathbf{t}'_k$
26: **end procedure**

B   Test-Time-Scaling Strategies

We ran the A*-Thought evaluation on MATH500 and LiveCodeBench, this time using a best-of-N strategy (with N=8) to calculate pass@k. The findings indicate that A*-Thought and test-time-scaling strategies can be used in combination, the experiment results are shown in Table 5.

Table 5: Experimental results on A*-Thought with test-time-scaling strategies.

Methods	MATH500				LiveCodeBench			
	pass@1	pass@2	pass@4	pass@8	pass@1	pass@2	pass@4	pass@8
Budget: 512 Tokens								
QwQ-32B	10.8	17.2	25.9	34.4	0.0	0.0	0.0	0.0
+ A*-Thought	**33.2**	**47.4**	**60.7**	**70.2**	**4.5**	**7.0**	**12.5**	**20.8**
Budget: 1024 Tokens								
QwQ-32B	16.6	29.6	42.1	53.2	0.0	0.1	0.1	0.3
+ A*-Thought	**50.8**	**65.4**	**75.8**	**81.8**	**11.8**	**20.4**	**31.9**	**44.8**
Budget: 2048 Tokens								
QwQ-32B	51.2	59.8	67.8	74.8	2.1	3.4	5.1	7.0
+ A*-Thought	**69.2**	**79.2**	**85.4**	**89.0**	**24.5**	**34.0**	**45.2**	**56.5**
Budget: 4096 Tokens								
QwQ-32B	75.4	80.2	84.0	86.8	12.4	17.7	23.0	28.0
+ A*-Thought	**78.8**	**86.8**	**91.1**	**93.2**	**39.0**	**50.9**	**60.9**	**68.8**

C   Performance of Models of Varying Sizes

This section reports experiments on Qwen2.5-7B-Instruct and Qwen2.5-14B-Instruct in Table 6.

Table 6: Experimental results on models of varying sizes.

Methods	MATH500		AMC23		OlympiadBench		GSM8K		Average		ACU
	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	
Budget: 512 Tokens											
TokenSkip-7B	9.4	511.41	2.5	512.00	4.0	512.00	25.5	508.40	10.4	511.0	2.03
A*-Thought-7B	**30.8**	**499.12**	**5.0**	**508.70**	**6.7**	**509.36**	**59.2**	**447.22**	**25.4**	**491.10**	**5.18**
TokenSkip-14B	9.2	510.89	7.5	512.00	4.8	512.00	29.0	511.17	12.6	511.52	2.47
A*-Thought-14B	**30.4**	**497.54**	**10.0**	**512.00**	**10.4**	**508.77**	**50.9**	**475.30**	**25.4**	**498.40**	**5.10**
Budget: 1024 Tokens											
TokenSkip-7B	24.8	986.62	10.0	1021.33	7.9	1019.11	51.0	907.40	23.4	983.62	2.38
A*-Thought-7B	**48.2**	**866.93**	**25.0**	**991.42**	**19.3**	**961.42**	**74.3**	**661.99**	**41.7**	**870.44**	**4.79**
TokenSkip-14B	23.6	991.65	10.0	1024.00	7.4	1020.22	47.8	960.13	22.2	999.00	2.22
A*-Thought-14B	**49.2**	**873.53**	**22.5**	**997.33**	**18.8**	**966.55**	**78.3**	**738.61**	**42.2**	**894.01**	**4.72**
Budget: 2048 Tokens											
TokenSkip-7B	35.4	1776.50	20.0	1968.47	14.7	1959.13	55.6	1534.37	31.4	1809.62	1.74
A*-Thought-7B	**59.2**	**1352.54**	**30.0**	**1715.22**	**25.4**	**1677.38**	**77.6**	**874.11**	**48.1**	**1404.81**	**3.42**
TokenSkip-14B	55.8	1678.03	20.0	1934.05	20.8	1940.58	73.8	1453.71	42.6	1751.59	2.43
A*-Thought-14B	**64.2**	**1308.18**	**40.0**	**1650.40**	**28.3**	**1610.04**	**86.1**	**930.89**	**54.7**	**1374.88**	**3.97**
Budget: 4096 Tokens											
TokenSkip-7B	39.8	3201.90	27.5	3593.93	20.9	3566.00	63.4	2651.40	37.9	3253.31	1.16
A*-Thought-7B	**61.8**	**1999.10**	**30.0**	**2840.70**	**29.6**	**3048.73**	**78.0**	**1271.18**	**49.9**	**2289.93**	**2.18**
TokenSkip-14B	68.4	2406.16	40.0	3338.80	32.5	3334.73	81.2	2054.78	55.5	2783.62	1.99
A*-Thought-14B	**70.8**	**1677.00**	**42.5**	**2350.32**	**34.6**	**2494.24**	**88.8**	**1100.52**	**59.2**	**1905.52**	**3.11**

D Detailed Results of R1-Distill-32B and s1.1-32B

This section reports experiments on R1-Distill-32B and s1.1-32B in Table 7 and Table 8, respectively.

Table 7: Experimental results on DeepSeek-R1-Distill-Qwen-32B.

Methods	MATH500		AMC23		OlympiadBench		GSM8K		Average		ACU
	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	
Budget: 512 Tokens											
R1-Distill-32B w/ s1K-1.1	11.2	512.00	5.0	512.00	4.0	512.00	28.1	512.00	12.1	512.00	2.36
+ CoD	10.0	512.00	7.5	512.00	4.0	512.00	35.0	512.00	14.1	512.00	2.76
+ BtC Effective Shortcut	9.6	512.00	5.0	512.00	4.0	512.00	30.3	512.00	12.2	512.00	2.39
+ BtC Skip Steps	11.4	512.00	7.5	512.00	4.2	512.00	29.4	512.00	13.1	512.00	2.56
+ TokenSkip	12.2	512.00	2.5	512.00	4.5	512.00	28.4	512.00	11.9	512.00	2.32
+ A*-Thought	**24.0**	512.00	**10.0**	512.00	**7.0**	512.00	**51.0**	512.00	**23.0**	512.00	**4.49**
Budget: 1024 Tokens											
R1-Distill-32B w/ s1K-1.1	24.2	1024.00	17.5	1024.00	7.0	1024.00	60.5	1024.00	27.3	1024.00	2.67
+ CoD	25.8	1024.00	10.0	1024.00	8.0	1024.00	69.8	1024.00	28.4	1024.00	2.77
+ BtC Effective Shortcut	24.4	1024.00	10.0	1024.00	8.8	1024.00	63.5	1024.00	26.7	1024.00	2.60
+ BtC Skip Steps	19.4	1024.00	7.5	1024.00	7.3	1024.00	60.2	1024.00	23.6	1024.00	2.30
+ TokenSkip	19.0	1024.00	12.5	1024.00	8.3	1024.00	54.4	1023.74	23.6	**1023.94**	2.30
+ A*-Thought	**39.6**	1024.00	**20.0**	1024.00	**16.0**	1024.00	**69.8**	1024.00	**36.4**	1024.00	**3.55**
Budget: 2048 Tokens											
R1-Distill-32B w/ s1K-1.1	54.2	2048.00	35.0	2048.00	19.9	2048.00	85.8	2048.00	48.7	2048.00	2.38
+ CoD	57.8	2048.00	35.0	2048.00	22.4	2048.00	88.4	2048.00	50.9	2048.00	2.49
+ BtC Effective Shortcut	55.8	2048.00	37.5	2048.00	23.3	2048.00	87.6	2048.00	51.1	2048.00	2.49
+ BtC Skip Steps	56.2	2048.00	37.5	2048.00	21.4	2048.00	87.9	2048.00	50.8	2048.00	2.48
+ TokenSkip	40.8	2048.00	37.5	2048.00	19.7	2048.00	59.1	2048.00	39.3	2048.00	1.92
+ A*-Thought	**61.8**	2048.00	**45.0**	2048.00	**26.7**	2048.00	**89.4**	2048.00	**55.7**	2048.00	**2.72**
Budget: 4096 Tokens											
R1-Distill-32B w/ s1K-1.1	**77.4**	4096.00	52.5	4096.00	37.8	4096.00	90.9	4096.00	64.7	4096.00	1.58
+ CoD	75.4	4096.00	60.0	4096.00	40.8	4096.00	91.7	4096.00	67.0	4096.00	1.64
+ BtC Effective Shortcut	75.8	4096.00	55.0	4096.00	39.8	4096.00	91.7	4096.00	65.6	4096.00	1.60
+ BtC Skip Steps	73.2	4096.00	52.5	4096.00	38.6	4096.00	**92.1**	4096.00	64.1	4096.00	1.56
+ TokenSkip	51.0	**4014.70**	35.0	4096.00	31.3	**4068.53**	57.8	**3969.45**	43.8	**4037.17**	1.08
+ A*-Thought	75.4	4096.00	**70.0**	4096.00	**40.9**	4096.00	91.7	4096.00	**69.5**	4096.00	**1.70**

Table 8: Experimental results on s1.1-32B.

Methods	MATH500		AMC23		OlympiadBench		GSM8K		Average		ACU
	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	
Budget: 512 Tokens											
s1.1-32B	9.8	512.00	5.0	512.00	5.5	512.00	28.3	511.85	12.2	511.96	2.37
+ CoD	12.6	511.86	7.5	512.00	5.3	512.00	32.8	511.21	14.6	511.77	2.84
+ BtC Effective Shortcut	13.4	511.76	10.0	512.00	4.5	512.00	32.3	511.69	15.1	511.86	2.94
+ BtC Skip Steps	12.2	512.00	2.5	512.00	4.5	512.00	31.2	511.95	12.6	511.99	2.46
+ TokenSkip	14.0	511.23	7.5	512.00	3.9	512.00	29.4	509.62	13.7	511.21	2.68
+ A*-Thought	**34.0**	**494.15**	12.5	**505.98**	11.9	509.73	56.1	474.03	28.6	495.97	**5.77**
Budget: 1024 Tokens											
s1.1-32B	25.6	1014.86	10.0	1024.00	8.8	1023.93	47.5	997.19	23.0	1015.00	2.26
+ CoD	37.8	1006.51	20.0	1019.02	11.6	1021.62	58.7	957.21	32.0	1001.09	3.20
+ BtC Effective Shortcut	42.8	1001.81	25.0	1024.00	13.8	1020.92	61.3	950.63	35.7	999.34	3.57
+ BtC Skip Steps	32.8	1011.61	15.0	1022.65	13.4	1022.07	58.5	975.83	29.9	1008.04	2.97
+ TokenSkip	28.4	983.36	15.0	1020.08	9.8	1019.31	45.4	955.14	24.7	994.47	2.48
+ A*-Thought	**55.4**	**862.14**	**32.5**	**969.05**	**22.3**	**960.90**	**80.9**	**738.52**	**47.8**	**882.65**	**5.41**
Budget: 2048 Tokens											
s1.1-32B	63.2	1765.50	35.0	1986.60	26.6	1978.44	80.9	1518.36	51.4	1812.23	2.84
+ CoD	69.6	1661.53	40.0	1906.22	31.9	1933.45	84.9	1339.1	56.6	1710.08	3.31
+ BtC Effective Shortcut	70.2	1634.15	42.5	1926.10	34.4	1928.26	85.0	1313.87	58.0	1700.60	3.41
+ BtC Skip Steps	67.4	1699.65	40.0	1929.10	32.5	1954.23	84.6	1383.99	56.1	1741.74	3.22
+ TokenSkip	59.2	1636.53	37.5	1903.90	27.3	1916.75	70.1	1586.98	48.5	1761.04	2.76
+ A*-Thought	**74.4**	**1229.54**	37.5	**1676.65**	34.4	**1635.22**	**88.6**	**913.33**	**58.7**	**1363.69**	**4.31**
Budget: 4096 Tokens											
s1.1-32B	80.0	2437.12	**62.5**	3261.12	45.1	3364.11	**92.3**	1762.46	**70.0**	2706.20	2.59
+ CoD	74.4	2701.37	55.0	3041.45	47.9	3228.24	90.8	1582.40	67.0	2638.37	2.54
+ BtC Effective Shortcut	80.0	2209.99	60.0	3132.45	**49.3**	3161.29	88.8	1609.22	69.5	2528.24	2.75
+ BtC Skip Steps	81.2	2325.75	60.0	3169.82	46.4	3222.67	90.4	1673.10	69.5	2597.84	2.68
+ TokenSkip	69.2	2404.56	57.5	3057.28	41.5	3230.59	77.7	2375.19	61.5	2766.91	2.22
+ A*-Thought	77.8	**1629.87**	52.5	**2744.00**	42.4	**2611.72**	91.9	**1055.83**	66.2	**2010.36**	**3.29**

16

E Additional Ablation Analysis

By default, we report the average accuracy using QwQ-32B over all the examined benchmarks.

E.1 Analysis on BIS

The design of BIS significantly influences the quality of assessing the importance of individual thought steps. ATTN (attention level importance) and NLL (model level importance) represent distinct measures of reasoning step significance. As demonstrated in Figure 8, manipulating their individual and combined effects on BIS reveals that their joint application is superior for improving the model performance across various budgets.
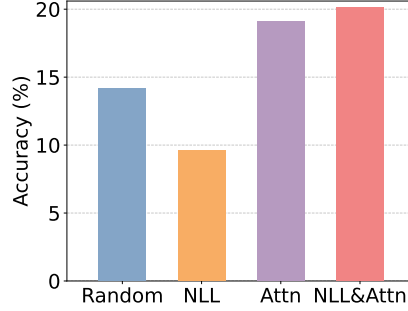
Figure 8: Effect of ATTN and NLL on BIS under the 512-token budget.

The parameter α modulates the balance between question and solution information within BIS. We conducted an ablation analysis, varying α across the discrete set of values [0, 0.25, 0.5, 0.75, 1], to determine its effect on reasoning data quality and, consequently, on model performance. Specifically, with $\alpha = 0$, the score is determined only by the information related to the question. In contrast, setting $\alpha = 1$ results in a score based entirely on information pertaining to the solution. As Figure 9 illustrates, optimal model performance is achieved when the BIS effectively integrates both question and solution perspectives, guided by an appropriate setting of α that ensures a judicious allocation of their respective contributions.

(a) 512 Tokens   (b) 1024 Tokens

(c) 2048 Tokens   (d) 4096 Tokens
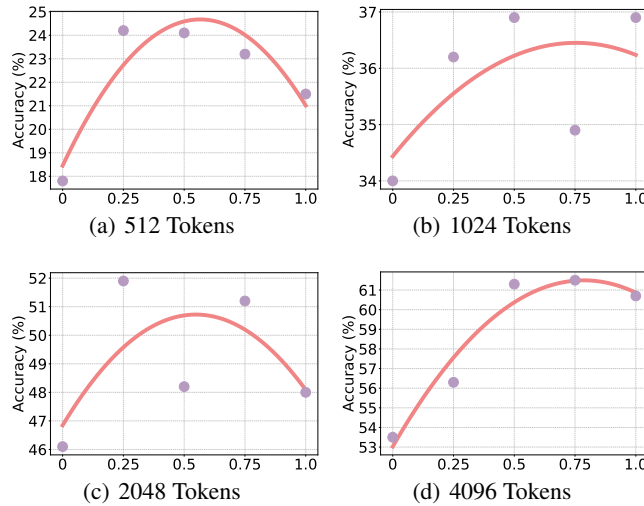
Figure 9: Effect of the hyperparameter α on model performance.

E.2 Analysis on A* Search

Appropriately setting the maximum exploration steps $k_{\max}$ are keys to optimizing the trade-off between performance and efficiency.

Figure 10 highlights distinct trends: moderate exploration steps are more effective for low-budget scenarios (512-2048 tokens). In contrast, for a 4096-token budget, performance benefits from a greater number of exploration steps. This is likely because more extensive exploration (i.e., deep search) can lead to more concise overall reasoning paths or solutions. Based on these observations, we set $k_{\max} = 20$ in our main experiments by default.
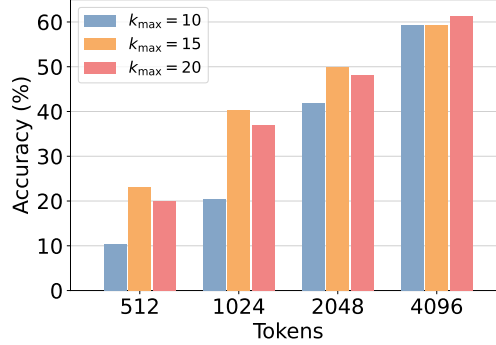
Figure 10: Relationship between the exploration step limit $k_{\max}$ and model performance.

The parameter β is used to adjust the weight of the current cost function $g(\cdot)$ in the overall cost function $f(\cdot)$. In the following supplementary experiments, we discussed its discrete values in [0.1, 0.5, 0.9] on the ARC and LiveCodeBench, the experiment results are shown in Table 9.

Table 9: Effect of the hyperparameter β on model performance.

Methods	ARC		LiveCodeBench		Average		ACU
	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	Acc.$^{(\uparrow)}$	Len.$^{(\downarrow)}$	
Budget: 512 Tokens							
$\beta = 0.1$	**63.5**	**381.02**	4.5	509.53	**34.00**	**445.28**	**7.64**
$\beta = 0.5$	52.5	438.03	4.0	510.73	28.25	474.38	5.96
$\beta = 0.9$	48.2	469.22	**5.8**	**506.15**	27.00	487.69	5.54

F   The Prompt used in this Work

This section details the prompts utilized in this work, including the system prompts presented in Table 10, and the specific CoD (Chain-of-Draft) prompts along with two variants of BtC (Break-the-Chain) baseline prompts, which are shown in Table 11.

Table 10: System prompt

System Prompt

Your role as an assistant involves thoroughly exploring questions through a systematic long thinking process before providing the final precise and accurate solutions. This requires engaging in a comprehensive cycle of analysis, summarizing, exploration, reassessment, reflection, backtracing, and iteration to develop well-considered thinking process.
 	
 Please structure your response into two main sections: Thought and Solution. In the Thought section, detail your reasoning process using the specified format:
 	
 `<|begin_of_thought|>`
 {thought with steps separated with '\n\n'}
 `<|end_of_thought|>`
 	
 Each step should include detailed considerations such as analisying questions, summarizing relevant findings, brainstorming new ideas, verifying the accuracy of the current steps, refining any errors, and revisiting previous steps.
 	
 In the Solution section, based on various attempts, explorations, and reflections from the Thought section, systematically present the final solution that you deem correct. The solution should remain a logical, accurate, concise expression style and detail necessary step needed to reach the conclusion, formatted as follows:
 	
 `<|begin_of_solution|>`
 {final formatted, precise, and clear solution}
 `<|end_of_solution|>`
 	
 Now, try to solve the following question through the above guidelines:

Table 11: Specific prompt for CoD and two variants of BtC

Methods	Prompt
CoD	Think step by step, but only keep a minimum draft for each thinking step, with 5 words at most.
BtC Effective Shortcut	Rapidly evaluate and use the most effective reasoning shortcut to answer the question.
BtC Skip Steps	Let's skip as much as possible.

G   Example of the A*-Thought Trajectories

This section provides two comparative examples that illustrate the CoT produced by A*-Thought versus the original CoT of QwQ-32B and s1.1-32B. Figures 11 and 12 demonstrate that A*-Thought successfully reduces redundant thought trajectories while preserving reasoning ability.

Convert the point $(0, 3)$ in rectangular coordinates to polar coordinates. Enter your answer in the form (r, θ), where $r > 0$ and $0 \le \theta < 2\pi$.

QwQ-32B

Okay, so I need to convert the rectangular coordinates $(0, 3)$ to polar coordinates. Hmm, let me remember how to do this. Polar coordinates are given as (r, θ), where r is the distance from the origin to the point, and θ is the angle made with the positive x-axis.

First, to find r, the formula is $r = \sqrt{x^2 + y^2}$. In this case, x is 0 and y is 3. Plugging those into the formula, I get $r = \sqrt{0^2 + 3^2} = \sqrt{0 + 9} = \sqrt{9} = 3$. Okay, so r is 3. That seems straightforward.

Now, for the angle θ. The formula for θ is usually $\theta = \arctan\left(\frac{y}{x}\right)$. But wait, here x is 0. That would make the denominator zero, which would make the tangent undefined. So, $\arctan\left(\frac{y}{x}\right)$ would be $\arctan\left(\frac{3}{0}\right)$, which is undefined. But arctangent of infinity is $\frac{\pi}{2}$, right? Because as x approaches 0 from the positive side, $\frac{y}{x}$ approaches positive infinity, so $\arctan$ approaches $\frac{\pi}{2}$. But since x is exactly 0, we have to consider the direction. If x is 0 and y is positive, the point is on the positive y-axis, which is $\theta = \frac{\pi}{2}$. If x is 0 and y is negative, it's $\theta = \frac{3\pi}{2}$. In this case, y is 3, which is positive, so θ should be $\frac{\pi}{2}$.

But let me double-check. Sometimes when converting coordinates, especially when x is 0, the angle can be tricky. Let me visualize the point $(0, 3)$. In rectangular coordinates, that's 3 units up along the y-axis. So if I imagine the polar coordinate system, the angle θ is measured from the positive x-axis. If I rotate $\frac{\pi}{2}$ radians (which is 90 degrees) from the positive x-axis, I end up pointing straight up along the positive y-axis. That's exactly where the point $(0, 3)$ is. So $\theta = \frac{\pi}{2}$ makes sense here.

Therefore, combining r and θ, the polar coordinates should be $\left(3, \frac{\pi}{2}\right)$. But let me make sure there's not another way to represent this. Since polar coordinates can have negative r values and angles adjusted by π, but the problem specifies that $r > 0$ and $0 \le \theta < 2\pi$. So we need r positive and θ within that interval. Since θ is already $\frac{\pi}{2}$, which is between 0 and 2π, and r is 3, which is positive, this should be the correct answer.

Wait, but just to confirm, what if I tried to use another angle? For example, could θ be $\frac{5\pi}{2}$? But $\frac{5\pi}{2}$ is equivalent to $\frac{\pi}{2}$ when you subtract 2π, so it's the same direction. But since θ is required to be less than 2π, $\frac{5\pi}{2}$ is out of range. So $\frac{\pi}{2}$ is the correct angle here. Also, if I tried to use a negative r, like $r = -3$, then θ would be $\frac{\pi}{2} + \pi = \frac{3\pi}{2}$. But r must be greater than 0, so that's not allowed. Therefore, $\left(3, \frac{\pi}{2}\right)$ is the only valid polar coordinate representation for the point $(0, 3)$ under the given constraints.

I think that's all. Let me recap: the point is on the positive y-axis, so the radius is 3, and the angle is $\frac{\pi}{2}$. Yep, that seems right. I can't think of any mistakes here. The formulas for r and θ both lead to these values, and considering the position of the point confirms it. So the final answer should be $\left(3, \frac{\pi}{2}\right)$.

Final Answer: The polar coordinates of the point $(0, 3)$ are $\boxed{\left(3, \frac{\pi}{2}\right)}$.

To convert the rectangular coordinates $(0, 3)$ to polar coordinates, we start by calculating the radius r using the formula $r = \sqrt{x^2 + y^2}$. Here, $x = 0$ and $y = 3$:

$$r = \sqrt{0^2 + 3^2} = \sqrt{9} = 3$$

Next, we determine the angle θ. Since $x = 0$ and $y = 3$ is positive, the point lies on the positive y-axis. The angle θ for a point on the positive y-axis is $\frac{\pi}{2}$.

We verify that $\theta = \frac{\pi}{2}$ is within the required interval $0 \le \theta < 2\pi$ and that $r = 3$ is positive. Thus, the polar coordinates are:

$$\boxed{\left(3, \frac{\pi}{2}\right)}$$

A*-Thought-QwQ-32B

So, in this case, r is 3. Then θ is the angle made with the positive x-axis. The point is $(0, 3)$, which is on the positive y-axis. So, the angle θ here is $\frac{\pi}{2}$ radians, because that's 90 degrees, pointing straight up along the y-axis.

But let me verify. The formulas for converting from rectangular to polar coordinates are $r = \sqrt{x^2 + y^2}$, which gives $r = \sqrt{0^2 + 3^2} = \sqrt{9} = 3$. Then θ is $\arctan\left(\frac{y}{x}\right)$. But wait, x is 0 here. So, $\arctan\left(\frac{y}{x}\right)$ would be $\arctan\left(\frac{3}{0}\right)$. However, division by zero is undefined, but in the context of polar coordinates, when x is 0, θ is either $\frac{\pi}{2}$ or $-\frac{\pi}{2}$, depending on the sign of y. Since y is positive (3), θ is $\frac{\pi}{2}$. So, that checks out. **But let me make sure there isn't another angle that could represent the same point.** For example, sometimes adding 2π to θ gives the same direction, but since the problem requires θ to be between 0 and 2π, $\frac{\pi}{2}$ is already within that range. So, the answer should be $(3, \frac{\pi}{2})$.

Final Answer: The polar coordinates are $\boxed{\left(3, \frac{\pi}{2}\right)}$.

To convert the point $(0, 3)$ from rectangular coordinates to polar coordinates, we use the formulas $r = \sqrt{x^2 + y^2}$ and $\theta = \arctan\left(\frac{y}{x}\right)$.

1. Calculate r:

$$r = \sqrt{0^2 + 3^2} = \sqrt{9} = 3$$

2. Determine θ:
   - Since $x = 0$ and $y = 3$, the point lies on the positive y-axis.
   - The angle θ for the positive y-axis is $\frac{\pi}{2}$.

Thus, the polar coordinates are $\boxed{\left(3, \frac{\pi}{2}\right)}$.

Figure 11: A specific example comparing QwQ-32B and A*-Thought-QwQ-32B (Ours), which red box represents the question, purple box represents the thought path, blue box represents the solution.

What is the value of x such that $\log_2(\log_2(2x-2)) = 2$?

s1.1-32B

Okay, so I need to find the value of x that satisfies the equation $\log_2(\log_2(2x-2)) = 2$. Hmm, nested logarithms, that could be a bit tricky. Let me take it step by step.

First, let me recall that if I have $\log_a(\log_b(\text{expression})) = c$, then I need to solve the inner logarithm first. But maybe I can work from the outside in. Since the outer logarithm is base 2 and equals 2, that means that the argument of the outer log must be 2^2. Because $\log_2(y) = 2$ implies $y = 2^2 = 4$. So, the inner logarithm must equal 4. Wait, let me confirm that. If $\log_2(\text{something}) = 2$, then that something must be 2^2, which is 4. So, the equation simplifies to $\log_2(2x-2) = 4$. Then, solving for the inner logarithm, $\log_2(2x-2) = 4$ implies that $2x-2$ must be 2^4, which is 16. Therefore, $2x-2 = 16$. Solving for x, add 2 to both sides: $2x = 18$, then divide by 2: $x = 9$. So, $x = 9$.

But wait, I need to check if this value of x actually satisfies the original equation. Let's plug $x = 9$ back into the original equation and see.

Original equation: $\log_2(\log_2(2x-2)) = 2$.

First, compute the inner expression: $2x-2 = 2 \times 9 - 2 = 18 - 2 = 16$. Then, $\log_2(16) = 4$, because $2^4 = 16$. Then, $\log_2(4) = 2$, since $2^2 = 4$. Therefore, the original equation becomes $\log_2(4) = 2$, which is correct. So, $x = 9$ works.

But wait, I should also check if there are any domain restrictions. The argument of the inner logarithm, $2x-2$, must be positive because you can't take the logarithm of a non-positive number. So, $2x-2 > 0$. Solving for x: $2x > 2$, so $x > 1$. Since $x = 9$ is greater than 1, it's valid. Also, the argument of the outer logarithm is $\log_2(2x-2)$, which must be positive because the logarithm is only defined for positive numbers. Wait, but the equation is $\log_2(\log_2(2x-2)) = 2$. For $\log_2(\log_2(2x-2))$ to be defined, $\log_2(2x-2)$ must be positive. Because the logarithm function is only defined for positive arguments. So, $\log_2(2x-2) > 0$. Since $\log_2(y) > 0$ when $y > 1$, this implies that $2x-2 > 1$. So, $2x > 3$, hence $x > 3/2$. So, the domain is $x > 3/2$. Since $x = 9$ is greater than $3/2$, it's valid.

Therefore, the solution is $x = 9$. But let's verify once again. Let's compute the inner expression: $2x-2 = 2 \times 9 - 2 = 16$. Then, $\log_2(16) = 4$. Then, $\log_2(4) = 2$, which matches the original equation. So, it's correct.

Wait, just to make sure, let me check if there's any other possible solution. The equation $\log_2(\log_2(2x-2)) = 2$. Let me think. The function $\log_2$ is a strictly increasing function, so the equation $\log_2(y) = 2$ has only one solution $y = 4$. Then, the equation $\log_2(2x-2) = 4$ has only one solution $2x-2 = 16$, leading to $x = 9$. So, there's only one solution.

Wait, but let me check if there's any extraneous solutions. When solving equations with logarithms, sometimes you can get extraneous solutions if you squared both sides or something. But in this case, we just used the definition of logarithms and solved linear equations. So, $x = 9$ is the only solution and it satisfies the domain restrictions. Therefore, the answer is $x = 9$.

Final Answer $\boxed{9}$

To solve the equation $\log_2(\log_2(2x-2)) = 2$, we start by recognizing that the outer logarithm must equal 2, which implies that the argument of the outer logarithm must be $2^2 = 4$. Therefore, we set up the equation: $\log_2(2x-2) = 4$ Next, we solve for the inner logarithm. The equation $\log_2(2x-2) = 4$ implies that $2x-2 = 2^4 = 16$. Solving for x: $2x-2 = 16 \implies 2x = 18 \implies x = 9$ We verify the solution by substituting $x = 9$ back into the original equation: Compute the inner expression: $2 \cdot 9 - 2 = 16$, compute the inner logarithm: $\log_2(16) = 4$, compute the outer logarithm: $\log_2(4) = 2$. The original equation is satisfied, confirming that $x = 9$ is indeed the solution. Additionally, we check the domain restrictions to ensure that the arguments of the logarithms are positive. The inner argument $2x-2$ must be positive, which requires $x > 1$, and the argument of the outer logarithm, $\log_2(2x-2)$, must also be positive, requiring $x > \frac{3}{2}$. Since $x = 9$ satisfies both conditions, it is valid.

Thus, the value of x is $\boxed{9}$.

A*-Thought-s1.1-32B

So:
$\log_2(2x-2) = 4$
Therefore:
$2x - 2 = 16$
Solving for x:
$2x = 16 + 2 = 18$
$x = 18/2 = 9$
So $x = 9$.

But let's check if this satisfies the original equation.

Original equation:
$\log_2(\log_2(2x-2)) = 2$
Plug in $x = 9$:
$\log_2(\log_2(2 \times 9 - 2)) = \log_2(\log_2(18 - 2)) = \log_2(\log_2(16))$.
$\log_2(16) = 4$, because $2^4 = 16$.
So:
$\log_2(4) = 2$, which is correct.
Therefore, $x = 9$ is the correct solution.

Final Answer $\boxed{9}$

To solve the equation $\log_2(\log_2(2x-2)) = 2$, we start by isolating the inner logarithm. We know that if $\log_2(y) = 2$, then $y = 2^2 = 4$. Therefore, we can rewrite the equation as: $\log_2(2x-2) = 4$. Next, we solve for $2x-2$ by converting the logarithmic equation to its exponential form. Since $\log_2(2x-2) = 4$, it follows that: $2x - 2 = 2^4 = 16$. Solving for x, we add 2 to both sides: $2x = 18$. Then, divide by 2: $x = 9$. To verify, we substitute $x = 9$ back into the original equation:

$$\log_2(\log_2(2 \cdot 9 - 2)) = \log_2(\log_2(16)) = \log_2(4) = 2$$

The solution satisfies the original equation, confirming that the value of x is $\boxed{9}$.

Figure 12: A specific example comparing s1.1-32B and A*-Thought-s1.1-32B (Ours), which red box represents the question, purple box represents the thought path, blue box represents the solution.

H Limitation and Future Works

At present, the application of A*-Thought is confined to supervised fine-tuning (SFT). However, in light of recent studies showcasing the efficacy of reinforcement learning (RL) in advancing the reasoning capacities of LRMs, we intend to investigate the extension of our methodology to an RL-based approach.

I Impact Statement

The profound thinking abilities of LRMs allow them to master complex scenarios, but at the cost of significant computing resource consumption. Our work focuses on developing more efficient and green AI, which promises to decrease the energy footprint of LRM deployment and, crucially, enable their application in more resource-scarce environments like endpoint devices.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist",**
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly provide the main contribution and key results.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: Limitations are discussed in Appendix G.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: This work does not contain theoretical assumptions and proofs.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Experimental details can be found in Section 4 and Appendices C, D, and E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Links to the involved resources (e.g., data and model) are listed in Section 3 and 4.

   Guidelines:
   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: The related experimental settings are detailed in Section 4 and Appendices D and E.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: In light of the proposed method's substantial improvement and the high computational cost, we did not undertake repeated experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Computation devices and training time are explained in Section 4 and 5.

Guidelines:

- The answer NA means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: This paper conform the mentioned Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Please refer to Appendix H for broader impacts.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Citations for all related original papers are provided, and adherence to their respective licenses has been ensured.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This paper does not create new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper does not involve such efforts.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: Section 2, 3, and 4 detail the usage of LLM in the study.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.