DART: Distilling Autoregressive Reasoning to Silent Thought

Anonymous ACL submission

Abstract

Chain-of-Thought (CoT) reasoning has significantly advanced Large Language Models (LLMs) in solving complex tasks. However, its autoregressive paradigm leads to significant computational overhead, hindering its deployment in latency-sensitive applications. To address this, we propose **DART** (**D**istilling **A**utoregressive **R**easoning to Silent **T**hought), a self-distillation framework that enables LLMs to replace autoregressive CoT with non-autoregressive Silent Thought (ST). Specifically, DART introduces two training pathways: the CoT pathway for traditional reasoning and the ST pathway for generating answers directly from a few ST tokens. The ST pathway utilizes a lightweight Reasoning Evolvement Module (REM) to align its hidden states with the CoT pathway, enabling the ST tokens to evolve into informative embeddings. During inference, only the ST pathway is activated, leveraging evolving ST tokens to deliver the answer directly. Extensive experimental results demonstrate that DART achieves comparable reasoning performance to existing baselines while offering significant efficiency gains, serving as a feasible alternative for efficient reasoning.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable performance (DeepSeek-AI et al., 2025; OpenAI, 2025) across various reasoning tasks by leveraging Chain-of-Thought (CoT) (Wei et al., 2022), which decomposes complex problems into intermediate reasoning steps. Despite these successes, the autoregressive nature of CoT introduces substantial computational cost, resulting in increased latency and limiting its effectiveness in real-time applications (Sui et al., 2025).

To alleviate this computational burden, implicit CoT reasoning (Deng et al., 2023, 2024) performs implicit reasoning in the hidden state rather than the explicit CoT tokens to avoid extra computation. Continuous thought methods (Hao et al., 2024; Cheng and Durme, 2024) compress discrete textual tokens into compact, continuous representations, reducing the number of intermediate tokens without obvious degradation in reasoning capability. However, these existing approaches either suffer from unsatisfactory performance or remain haunted by the autoregressive generation paradigm, leading to suboptimal efficiency.

To this end, we propose DART (Distilling Autoregressive Reasoning to Silent Thought), a novel framework that enables the LLMs to internalize the autoregressive CoT into non-autoregressive Silent Thought (ST) with an excellent efficiency-efficacy trade-off. To be specific, DART employs two pathways in the training procedure as shown in Figure 1, namely: the CoT pathway, which generates both the answer tokens and the explicit CoT tokens; and the ST pathway, which focuses solely on generating answers, conditioned on the ST tokens concatenated after the question. Additionally, the ST pathway introduces a lightweight Reasoning Evolvement Module (REM) to align the hidden state of the last word preceding the answer with that of the CoT pathway. During inference, initial ST tokens are appended to user input and processed through the REM-equipped ST pathway. Analogous to human cognition that progresses from vague conceptual abstraction to concrete resolution, these ST tokens evolve into increasingly informative embeddings as they propagate through the network, ultimately serving as a context-aware bridge between the instruction and its logically grounded response. Empirical results demonstrate that DART achieves significant efficiency gains while maintaining comparable performance. To summarize, our contributions are as follows:

- We explore non-autoregressive ST as a promising alternative to the CoT paradigm, providing valuable insights for future work;

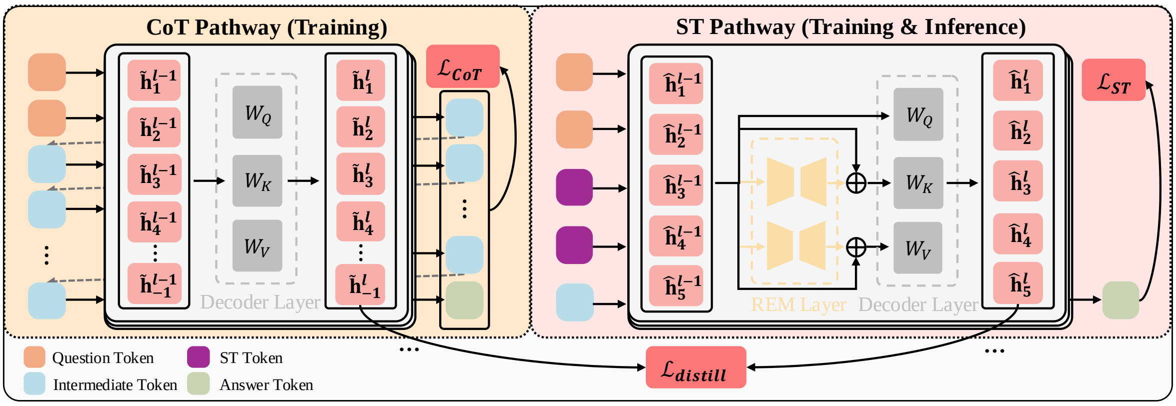

Figure 1: Overall Framework of DART. During inference, we employ the ST pathway to respond directly without step-by-step reasoning in prior work (Wei et al., 2022; Hao et al., 2024). The shared intermediate token represents the separator token. Feed-forward layer in the decoder layer is omitted for simplicity.

- We introduce DART, a simple but effective framework that employs REM to align autoregressive CoT with non-autoregressive ST in a dual-pathway architecture;
- We conduct extensive experiments to validate DART on multiple reasoning benchmarks, demonstrating its remarkable efficiency alongside satisfactory accuracy and interpretability.

2 Related Work

Empirical results and theoretical analysis (Feng et al., 2023; Liu et al., 2024) have demonstrated the effectiveness of CoT developed from supervised fine-tuning (Yue et al., 2024; Yu et al., 2024) and reinforcement learning (Wang et al., 2024; Shao et al., 2024; DeepSeek-AI et al., 2025). However, intermediate steps in the CoT reasoning will cause extra computational cost, resulting in low throughput. To reduce this computation overhead, Coconut (Hao et al., 2024) employs curriculum learning to fine-tune an LLM capable of autoregressively generating final-layer hidden states to serve as the replacement of CoT tokens. These final-layer hidden states, dubbed continuous thought, are more information-dense, thus reducing the intermediate steps. One contemporaneous work, CODI (Shen et al., 2025) also exploits the continuous thought but employs an end-to-end distillation framework rather than the curriculum learning. Despite the impressive performance of these methods, their efficiency is still limited by the autoregressive pattern. On the other hand, iCoT (Deng et al., 2023, 2024) manages to embed the CoT reasoning within the model's hidden space. However, it lacks scalability for the larger models (Shen et al., 2025).

3 Method

3.1 Dual-Pathway Architecture

Given a question Q, our goal is to fine-tune a causal decoder-only LLM parameterized by θ to provide the proper answer $Y = \{y_i\}_{i=1}^{M}$. In DART, we introduce a dual-pathway architecture to allows two distinct answering ways during training.

Chain-of-Thought Pathway. This pathway adheres to the conventional CoT approach, where the model first produces a sequence of intermediate reasoning steps $Z = \{z_i\}_{i=1}^{N}$ before producing the final answer. During training, the cross-entropy loss for next token prediction is adopted for optimizing this pathway:

$$\mathcal{L}_{CoT} = -\frac{1}{N}\sum_{i=1}^{N} \log\left(z_i \mid Q, z_{1:i-1}; \theta\right)$$
$$- \frac{1}{M}\sum_{i=1}^{M} \log\left(y_i \mid Q, Z, y_{1:i-1}; \theta\right).$$

Notably, the first $t-1$ tokens of Z are indeed CoT tokens, while the remaining are separator tokens shared with the ST pathway. In this paper, we fix $z_{t:N}$ as the answer prompt "*Answer:*".

Silent Thought Pathway. In contrast, the ST pathway directly generates the answer conditioned on the preceding ST sequence $S = \{s_i\}_{i=1}^{C}$ and separators $z_{t:N}$. Here, each s_i is a special token `<st>` and C is set as 20 in this paper. The objective function of this pathway can be formulated as

$$\mathcal{L}_{ST} = -\frac{1}{M}\sum_{i=1}^{M} \log\left(y_i \mid Q, X, y_{1:i-1}; \theta, \phi\right),$$

where $X = [S; z_{t:N}]$ and ϕ is the parameters associated with REM to be detailed in Section 3.2.

3.2 REM-based Self-Distillation

Our preliminary experiments show that enabling the evolution of the ST token requires more fine-grained supervision from CoT data to capture deeper intrinsic reasoning patterns. As revealed by the prior work (Dai et al., 2023), the intermediate words essentially impose a shift to the hidden state of the last word before the answer. We can approximate this effect at the l-th decoder layer as

$$\tilde{\mathbf{a}}^l \approx \mathbf{a}^l + W_V^l H_Z^{l-1} (W_K^l H_Z^{l-1})^T \mathbf{q}^l,$$

$$\tilde{\mathbf{h}}^l \approx \mathbf{h}^l + f\left(W_V^l H_Z^{l-1} (W_K^l H_Z^{l-1})^T \mathbf{q}^l\right),$$

where $f(\cdot)$ denotes the feed-forward layer; $\mathbf{q}^l$ is the attention query vector of z_N in l-th decoder layer; $\mathbf{a}^l$ and $\mathbf{h}^l$ indicate the output of the attention head and the output hidden state, given the question-only input; H_Z^{l-1} represents the input hidden state of intermediate token sequence Z; and W_K^l, W_V^l are the key and value projection matrices. A detailed derivation is provided in the Appendix A.

Since the intermediate tokens are autoregressively generated conditioned on the question Q and the model parameters θ, the induced shift can be viewed as $g_{\theta^{1:l}}\left(h_\theta\left(Q\right)\right)$ where $Z = h_\theta(Q)$ and $\theta^{1:l}$ denotes the parameter of the first l layers. Given that flattening the function $h_\theta(\cdot)$ is non-trivial, we propose to approximate the process by introducing a lightweight REM module at each decoder layer, which also leverages both the parameters θ and the in-context information from Q. Specifically, to induce such a shift, REM adapts the standard attention mechanism as:

$$\widehat{\mathbf{a}} \approx W_V \bar{W}_R^V [H_Q; H_X] \left(W_K \bar{W}_R^K [H_Q; H_X]\right)^T \mathbf{q},$$

$$\widehat{\mathbf{h}} \approx \mathbf{h} + g_{\theta^{1:l}, \phi^{1:l}}([Q; X]).$$

where $\bar{W}_R^J = \frac{\alpha}{d} W_{R_2}^J {W_{R_1}^J}^T + I$ for $J \in \{K, V\}$. Here, $W_{R_1}^J, W_{R_2}^J \in \mathbb{R}^{n \times d}$ are learnable matrices injected before the key and value projection matrices; n, d are the hidden state dimension and the REM projection space dimension; and α is a scaling hyperparameter. The superscript for the layer index is omitted for simplicity. REM offers two key advantages: (1) It introduces a few additional parameters while enabling rich interactions between Q and θ, effectively capturing contextual and domain-specific knowledge; (2) It is a simple plug-in module compatible with any decoder-only LLM, which can be seamlessly merged into the original architecture without increasing inference-time parameters.

Based on the analysis, we adopt the following distillation loss to guide the learning process:

$$\mathcal{L}_{distill} = \frac{1}{L} \sum_{l=1}^{L} \frac{1}{\sigma(\tilde{\mathbf{h}}^l)} \left\| \tilde{\mathbf{h}}^l - \widehat{\mathbf{h}}^l \right\|_1,$$

where $\sigma(\cdot)$ denotes the standard deviation within a batch. By aligning these hidden states, the function $g_{\theta^{1:l}, \phi^{1:l}}([Q; X])$ is encouraged to approximate $g_{\theta^{1:l}}\left(h_\theta\left(Q\right)\right)$, thereby distilling the reasoning capability from the CoT pathway into the ST pathway. Furthermore, we empirically show in Section 4.3 that the initial meaningless token `<st>` will evolve into an informative latent representation as it goes through the REM-equipped ST pathway, simulating a blur-to-concrete thinking process. To summarize, the overall objective function of DART is

$$\mathcal{L}_{DART} = \mathcal{L}_{CoT} + \mathcal{L}_{ST} + \lambda \mathcal{L}_{distll},$$

where $\lambda = 20$ is a trade-off hyperparameter.

4 Experiments

To validate the design of DART, we conduct extensive experiments and present the key results. Additional implementation details and extended findings are provided in Appendices B-D.

4.1 Experimental Settings

Datasets. Following previous work (Deng et al., 2024; Hao et al., 2024), we fine-tune models on GSM8K-Aug (Deng et al., 2023), an augmented version of GSM8K (Cobbe et al., 2021), which includes diverse reasoning traces. To mitigate the risk of models memorizing the final answer from CoT sequences, the last step is omitted during training (Shen et al., 2025). For out-of-distribution evaluation, we adopt GSM-HARD (Gao et al., 2023), SVAMP (Patel et al., 2021), and MultiArith (Roy and Roth, 2015) as robustness benchmarks.

Baselines. We compare DART against three autoregressive methods and three non-autoregressive methods, namely: (1) CoT, which fine-tunes the model on CoT data to perform the traditional CoT reasoning; (2) Coconut (Hao et al., 2024), which trains the model with CoT data in a mutil-stage manner and leverages autoregressively generated continuous thought; (3) CODI, similar to Coconut but employing a one-stage distillation framework using both CoT and no-CoT data; (4) No-CoT, which trains the model on no-CoT data to answer directly; (5)iCoT (Deng et al., 2024), which trains

		In-Distribution		Out-of-Distribution					
Methods	Is NAR?	GSM8K		GSM-HARD		SVAMP		MultiArith	
CoT	✗	58.8	425	13.4	477	59.9	227	97.2	218
Coconut (Hao et al., 2024)	✗	50.6	390	11.2	483	53.1	181	96.5	217
CODI (Shen et al., 2025)	✗	55.6†	143	12.8†	153	61.1†	141	96.1†	132
No-CoT	✓	32.5	36	7.1	57	40.6	34	61.3	33
iCoT (Deng et al., 2024)	✓	19.0†	36	4.4 †	57	40.9†	34	39.0†	33
PauseFT (Goyal et al., 2024)	✓	32.1	37	7.3	60	40.4	35	59.2	33
DART (Ours)	✓	42.6	37	10.9	60	50.5	35	84.8	33

Table 1: Results on GSM8K, GSM-HARD, SVAMP, and MultiArith. Accuracy (%) is on the left and inference time (ms) on the right for each benchmark. NAR stands for non-autoregressive. †The result is from (Shen et al., 2025).

Methods	Accuracy (%)
No-CoT	32.5
DART	42.6
w/o $\mathcal{L}_{distill}$	33.7
w/o ST	36.8
w/o REM	36.2
w/ LoRA-REM	40.8

Table 2: Ablation studies. *LoRA-REM* indicates that we apply LoRA (Hu et al., 2022) as REM.

the model to reason in the hidden space by applying stepwise internaliztion; (6) and PauseFT (Goyal et al., 2024), which inserts C special filler tokens `<pause>` between the question and answer to allow extra computations. All methods are fine-tuned on Llama-3.2-1B (Dubey et al., 2024) for consistency.

4.2 Empirical Results

Comparison between Baselines. Table 1 summarizes the performance across GSM8K, GSM-HARD, SVAMP, and MultiArith. To ensure a fair comparison of efficiency, we measure the inference time of all baselines on a Nvidia A10 GPU, even for those whose accuracy is sourced from prior reports. As shown, DART achieves the best performance among all NAR baselines on GSM8K, delivering a notable 10.1% accuracy gain with negligible latency overhead (only 1 ms). On all out-of-distribution datasets, DART consistently outperforms other NAR methods, indicating robust generalization beyond the training distribution. While AR methods obtain higher accuracy, they suffer from significantly reduced inference efficiency due to stepwise generation, despite efforts to compress reasoning steps. These results demonstrate that DART achieves a compelling trade-off between accuracy and efficiency by fully leveraging distilled CoT knowledge in a single-step latent space.

Ablation Study. To validate the contributions of key DART components, we evaluate the variants with certain components omitted or replaced. Our findings in Table 2 are as follows: (1) Omitting $\mathcal{L}_{distill}$, which transfers reasoning patterns from CoT trajectories, leads to a substantial performance drop; (2) Excluding the ST tokens impairs accuracy, likely due to the loss of CoT-derived positional priors; (3) Our proposed REM significantly enhances reasoning capability compared to using either no additional module or the vanilla LoRA.

4.3 Interpretability Analysis

We decode the final hidden states of ST tokens into natural language using the model's word embeddings, finding that 69.9% of the translated tokens match the words in ground-truth CoT. This ratio further rises to 80.1% for cases that yield the correct answer. Additionally, we observe that the ST tokens also reflect a progressive reasoning process similar to CoT. For example, the top-1 ST translation is "`192 192 192 192 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24`", corresponding to the CoT "`«3*64=192» «192/8=24»`".

5 Conclusion

We present DART, a fine-tuning framework that empowers LLMs to perform implicit reasoning in a non-autoregressive manner. By distilling knowledge from CoT data, the models trained with DART achieve a remarkable balance between accuracy and latency using the evolving ST tokens. Extensive experiments confirm DART's robustness and the effectiveness of its design.

Limitations

Additional Training Resources. Due to its dual-pathway architecture, DART requires more computational resources. In our experiments, the proportion of trainable parameters is 2.86%, compared to 1.79% for LoRA.

Supervision Signal. Currently, DART aligns only the activation value of the last word before the answer between the CoT and ST pathways. This limited supervision may be suboptimal, as it can overlook some information in intermediate tokens. Incorporating more comprehensive supervision signals may help DART achieve better performance.

Demand for CoT data. DART relies on CoT data for knowledge distillation, which may not always be available. One potential solution is to use large LLMs capable of CoT reasoning to generate synthetic CoT data, though this approach incurs additional preprocessing costs.

References

Jeffrey Cheng and Benjamin Van Durme. 2024. Compressed chain of thought: Efficient reasoning through dense representations. *CoRR*, abs/2412.13171.

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

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. 2023. Why can GPT learn in-context? language models secretly perform gradient descent as meta-optimizers. In *Findings of the Association for Computational Linguistics*, pages 4005–4019.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 81 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948.

Yuntian Deng, Yejin Choi, and Stuart M. Shieber. 2024. From explicit cot to implicit cot: Learning to internalize cot step by step. *CoRR*, abs/2405.14838.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart M. Shieber. 2023. Implicit chain of thought reasoning via knowledge distillation. *CoRR*, abs/2311.01460.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 82 others. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. 2023. Towards revealing the mystery behind chain of thought: A theoretical perspective. In *Advances in Neural Information Processing Systems 36*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. PAL: program-aided language models. In *International Conference on Machine Learning*, volume 202, pages 10764–10799.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. 2024. Think before you speak: Training language models with pause tokens. In *The Twelfth International Conference on Learning Representations*.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space. *CoRR*, abs/2412.06769.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations*.

Zhiyuan Liu, Hong Liu, Denny Zhou, and Tengyu Ma. 2024. Chain of thought empowers transformers to solve inherently serial problems. In *The Twelfth International Conference on Learning Representations*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

OpenAI. 2025. Learning to reason with llms.

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

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in NaturalLanguage Processing*, pages 1743–1752.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. 2025. CODI: compressing chain-of-thought into continuous space via self-distillation. *CoRR*, abs/2502.21074.

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

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 352*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2024. Mammoth: Building math generalist models through hybrid instruction tuning. In *The Twelfth International Conference on Learning Representations*.

A Analysis of the Shift Value

For better readability, we first summarize the mathematical notations adopted for this paper in Table 3.

Similar to (Dai et al., 2023), we mainly focus on the effect of the intermediate sequence X on the hidden state of the last separator token z_N. Firstly, we can derive the simplified expression for the activation $\tilde{\mathbf{a}}^l$ of z_N as follows:

$$
\begin{aligned}
\tilde{\mathbf{a}}^l &= W_V^l[H_Q; H_Z]\mathrm{softmax}\left(\frac{W_K^l[H_Q; H_Z]}{\sqrt{n}}\right)^T \mathbf{q}^l \\
&\approx W_V^l[H_Q; H_Z]\left(W_K^l[H_Q; H_Z]\right)^T \mathbf{q}^l \\
&= W_V^l H_Q \left(W_K^l H_Q\right)^T + W_V^l H_Z (W_K^l H_Z)^T \mathbf{q}^l \\
&\triangleq \mathbf{a}^l + W_V^l H_Z (W_K^l H_Z)^T \mathbf{q}^l,
\end{aligned}
$$

where $W_K^l, W_V^l \in \mathbb{R}^{n\times n}$ are the key and value projection matrices of the l-th decoder layer, H_Q, H_Z are the input hidden state of question Q and intermediate tokens Z, $\mathbf{q}^l$ is the attention query vector corresponding to z_N, and $\mathbf{a}^l$ is the activation when only Q is given. The superscript $l-1$ for H_Q, H_Z is omitted for simplicity. The approximation in the second step is obtained by omitting the softmax operation and scaling factor $\sqrt{n}$. Then, by going through the feed-forward layer $f(\cdot)$, we can get the hidden state of z_N in the l-th layer as follows:

$$
\tilde{\mathbf{h}}^l \approx \mathbf{h}^l + f\left(W_V^l H_Z (W_K^l H_Z)^T \mathbf{q}^l\right)
$$

where $\mathbf{h}^l = f(\mathbf{a}^l)$. Since Z can be viewed as the output of LLM given Q, we further define

$$
g_{\theta^{1:l}}\left(h_\theta\left(Q\right)\right) \triangleq f\left(W_V^l H_Z (W_K^l H_Z)^T \mathbf{q}^l\right).
$$

Hence, the CoT effectively injects a shift in the hidden state of z_N, which can be parameterized by the model parameters θ and input Q. Based on this, we employ REM to construct $g_{\theta^{1:l},\phi^{1:l}}([Q; X])$ and apply an L1 distance loss to approximate this shift.

B Datasets

Statistics. The statistics of utilized datasets are provided in Table 4

Examples We provide some examples of the data used in our experiments.

GSM8K-Aug

Question = "Andy receives a monthly salary of $800 but he has to pay a tax of 7%. How much is his net salary?"
CoT = "«800*7/100=56» «800-56=744»"
Answer = "744"

GSM-HARD

Question = "A robe takes 2287720 bolts of blue fiber and half that much white fiber. How many bolts in total does it take?"
Answer = "3431580.0"

Notation	Mathematical Meaning
l	The index of the current decoder layer, usually used as a superscript.
n, d	The hidden state dimension and the REM projection dimension.
α	The hyperparameter for scaling in REM.
Q	The question sequence.
$Y = \{y_i\}_{i=1}^{M}$	The answer sequence of length M.
$Z = \{z_i\}_{i=1}^{N}$	The intermediate sequence of length N.
$S = \{s_i\}_{i=1}^{C}$	The ST sequence of length C, with each s_i set as a special token `<st>`.
t	The index separating the CoT sequence $z_{1:t-1}$ and the separators $z_{t:N}$ in Z.
$[\cdot;\cdot]$	The concatenation operation for matrix pairs and sequence pairs.
$X = [S; z_{t:N}]$	The concatenation of the ST sequence and separators.
W_K^l, W_V^l	The key and value projection matrices.
W_{R1}^J, W_{R2}^J	The REM matrices for key projection matrices when $J = K$ and for value projection matrices when $J = V$.
H_Q^l, H_Z^l, H_X^l	The output hidden state matrices associated with Q, Z, and X.
$\mathbf{q}^l$	The attention query vector of the last separator z_N.
$\mathbf{a}^l, \tilde{\mathbf{a}}^l, \widehat{\mathbf{a}}^l$	The output vectors of the attention head corresponding to the last separator z_N in the no-CoT, CoT, and ST cases.
$\mathbf{h}^l, \tilde{\mathbf{h}}^l, \widehat{\mathbf{h}}^l$	The output hidden states of the last separator z_N corresponding to the no-CoT, CoT, and ST cases.
θ, ϕ	The parameters of LLM and REM.
$\theta^{1:l}, \phi^{1:l}$	The parameters of the first l layers of LLM and REM.
$f(\cdot)$	The feed-forward function in the decoder layer.
$h_\theta(\cdot)$	The generation function for the model to produce an answer sequence conditioned on the input.
$g_{\theta^{1:l}}(\cdot), g_{\theta^{1:l},\phi^{1:l}}(\cdot)$	The function for the model to produce a shift value conditioned on the input and the parameters in the subscript.

Table 3: Frequently used notations along with their mathematical meaning.

Dataset	Training	Evaluation
GSM8K-Aug	385620	1319
GSM-HARD	-	1319
SVAMP	-	1000
MultiArith	-	600

Table 4: Dataset statistics. GSM-HARD, SVAMP and MultiArith are only used for evaluation.

SVAMP

Question = "Each pack of dvds costs 76 dollars. If there is a discount of 25 dollars on each pack. How much do you have to pay to buy each pack?"
Answer = "51.0"

MultiArith

Question = "Faye had 34 coloring books. If she gave away 3 of them, but then bought 48 more, how many would she have total?"
Answer = "79"

C Implementation Details

For all experiments, we set $d = 128$ and $\alpha = 32$ for REM, consistent with the configuration used for LoRA (Hu et al., 2022) to fine-tune the CoT pathway. We employ the AdamW (Loshchilov and Hutter, 2019) with a cosine annealing learning rate schedule following a 3% warm-up period. Additionally, we enable the `bf16` in the trainer and evaluate the models using bfloat16 precision.

We fine-tune the Llama-3.2-1B, the primary base model in our experiments, for 10 epochs with the learning rate initialized as 8e-4.

For Coconut, we adopt their official implementation and use the same number of training epochs as in our experiments.

D More Experimental Results.

Sensitivity Analysis on ST Token Number. We conduct the experiments on GSM8K with various values of the ST token number C. As shown in Figure 2, the accuracy rapidly increases during the initial stage as C grows, then stabilizes once the token number becomes sufficient, demonstrating both the necessity and robustness of using ST tokens. Furthermore, thanks to the non-autoregressive paradigm, DART introduces negligible latency even as C increases.

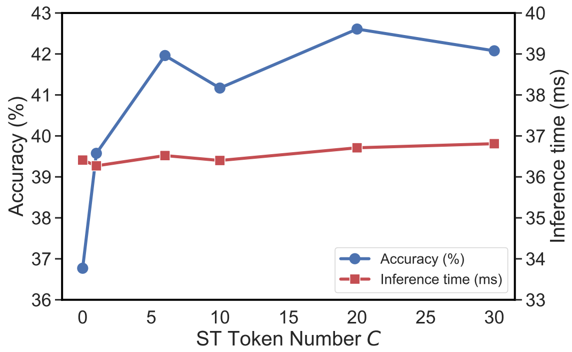

Figure 2: Accuracy and inference time on GSM8K with varying C, the number of ST tokens.

Experiments on Various Base Models. To further demonstrate the robustness of DART across different LLMs, we conduct the experiments using Qwen2.5-1.5B (Team, 2024) and GPT2 (Radford et al., 2019) as base models. For Qwen2.5-1.5B, we initialize the learning rate at 5e-4 and keep all other configurations as those used for Llama-3.2-1B. For GPT2, we set the initial learning rate to 2e-3 and train the model for 40 epochs. As shown in Table 5, DART consistently boosts the reasoning capabilities without notable latency.

	GPT2		Qwen2.5-1.5B	
Methods	Acc	IT	Acc	IT
No-CoT	16.2	18	33.3	95
DART	24.7	19	42.2	95

Table 5: Results on GPT2 and Qwen2.5-1.5B. Acc and IT indicate the Accuracy (%) and inference time (ms, on a Nvidia A10 GPU), respectively.

Methods	Accuracy
No-CoT	32.5
DART (w/ $\mathcal{L}_{distill}$ on z_N)	42.6
w/ $\mathcal{L}_{distill}$ on y_1	31.5
w/ $\mathcal{L}_{distill}$ on $[z_N; Y]$	33.7

Table 6: Accuracy (%) with various tokens used in $\mathcal{L}_{distill}$.

Different Choices of Alignment Token. To empirically demonstrate the rationality of applying $\mathcal{L}_{distill}$ on z_N, we perform the experiments with different alignment tokens. As shown in Table 6, employing other tokens like the answer tokens in $\mathcal{L}_{distll}$ hinders the effectiveness of alignment.