LEARNING TO REDIRECT AND STOP: SELF-TEACHING CONTROL FOR REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) have demonstrated strong reasoning capabilities through extended chain-of-thought generation, yet they may take suboptimal directions at critical reasoning transitions or continue reasoning after a correct answer has already emerged. We introduce Self-teaching Decision and Exit for Reasoning (SeDER), a lightweight framework that adaptively controls both the direction and length of reasoning while keeping the underlying language model frozen. At its core, a decision-token head selectively intervenes at uncertain reasoning transitions. Using successful rollouts generated by the base model itself, we construct counterfactual alternatives and evaluate them based on their compatibility with successful continuations and stability under short stochastic rollouts. The resulting utility differences yield weighted pairwise preferences for training the decision-token head to rerank plausible reasoning operators, without requiring a stronger teacher, a separate process reward model, or reinforcement learning. We further incorporate a lightweight stopping head that terminates reasoning once a sufficient solution has been reached. Experiments across multiple reasoning benchmarks with DeepSeek-R1-Distill-Qwen-1.5B and 7B demonstrate consistent accuracy improvements while substantially reducing reasoning length. SeDER achieves the highest accuracy among the evaluated baselines across all benchmarks, while ablation studies validate the effectiveness of its utility construction and selective intervention strategy.

1 INTRODUCTION

Large language models (LLMs) (Jaech et al., 2024; Guo et al., 2025; Chen et al., 2026; Zhang et al., 2025b) have achieved increasingly strong performance on complex reasoning tasks by generating extended chains of intermediate reasoning before producing a final answer (Wei et al., 2022; Kojima et al., 2022). However, recent studies show that reasoning models can take suboptimal reasoning directions, leading to incorrect intermediate steps, unnecessary detours, or redundant reasoning (Shojaee et al., 2026; Sui et al., 2025; Jiang et al., 2025a; Kang et al., 2025a). Such failures often originate from a relatively small number of consequential transitions, particularly at positions associated with high uncertainty or planning-related tokens (Ling et al., 2026; Zhu et al., 2026; Shen et al., 2026). Once an unfavorable transition is taken, subsequent reasoning may remain locally coherent while following a globally incorrect or redundant trajectory. As illustrated in Figure 1(a), replacing the token *Therefore* with *But* redirects the subsequent reasoning toward redundant thinking. Conversely, intervening at only a small fraction of such decision points can improve reasoning outcomes. These observations motivate us to identify uncertain transitions at which the model is choosing how to reason next, and to intervene only when that choice is likely to matter.

However, identifying an important decision point is only half of the problem. Once a model is uncertain between several plausible reasoning directions, how should the better direction be determined? Existing approaches often rely on additional supervision or computation, including process reward models (Lightman et al., 2024; Liao et al., 2025), external verifiers or stronger teacher models (Ling et al., 2026), and multi-branch exploration during inference (Wang et al., 2022; Sun et al., 2024). Reinforcement-learning-based methods can also be effective, but typically require substantially greater training cost and complexity (Aggarwal & Welleck, 2025; Arora & Zanette, 2026). This raises a natural question: can a reasoning model learn to improve its own reasoning decisions without a stronger teacher, a separate guiding model, or reinforcement learning?

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

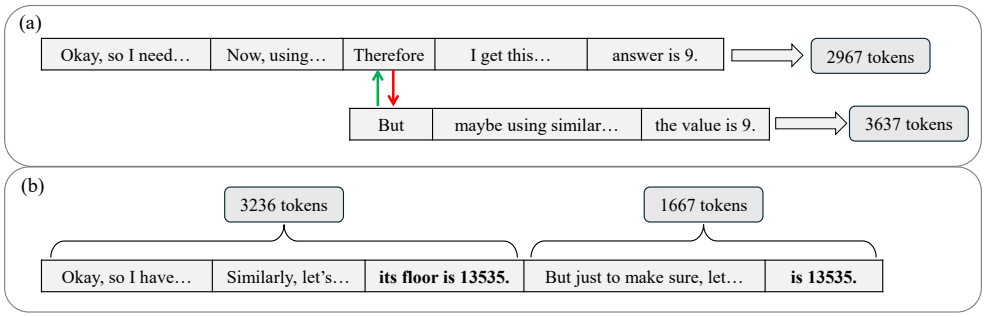


Figure 1: (a) A change in the decision token at a critical reasoning transition leads to unnecessary reasoning. (b) The reasoning model continues reasoning after the correct answer has emerged.

We address this question by exploiting the model's own successful reasoning trajectories. At uncertain reasoning transitions, a successful rollout provides a reference decision whose plausible alternatives can be examined counterfactually using the same base model. We compare these alternatives using local compatibility and stability signals, convert the resulting evidence into pairwise preferences, and train a lightweight decision-token head to guide inference. Importantly, this supervision requires neither a stronger teacher nor a separate reward model: the model learns to improve its reasoning decisions from its own successful behavior.

Reasoning quality is only one dimension of effective deliberation. As illustrated in Figure 1(b), reasoning models may also continue generating after the essential answer has already appeared, resulting in redundant verification and, in some cases, harmful self-correction (Zhang et al., 2025a; Nagle et al., 2026). We therefore complement decision-token guidance with an early-stopping module inspired by prior work on learning optimal exit points in chain-of-thought reasoning. We train a lightweight stopping head to detect when further reasoning is no longer necessary and apply it during inference. Together, the two modules address complementary aspects of reasoning control: the decision-token head determines which direction to take at uncertain transitions, while the stopping head determines when further reasoning should terminate. We refer to the resulting framework as SeDER (Self-teaching Decision and Exit for Reasoning). Our main contributions are as follows:

Self-teaching decision-token guidance. We introduce a lightweight mechanism that selectively reranks reasoning operators at low-confidence transitions while keeping the base language model frozen. Unlike approaches based on stronger-teacher distillation, cross-model delegation, or external process reward models, the decision head requires no stronger or separate guiding model.

Self-derived counterfactual preference learning. We construct supervision from the base model's own successful rollouts by comparing the local consequences of alternative reasoning operators. Compatibility with a successful continuation and stability under short stochastic rollouts are combined into a local utility, which is converted into weighted pairwise preferences for training the decision head.

Selective control of reasoning direction and length. We combine decision-token intervention with a lightweight stopping module that detects when sufficient reasoning has already been completed, providing complementary control over both the direction and duration of reasoning without modifying the underlying language model.

2 Related Work

Inference-Time Dynamic Reasoning and Test-Time Scaling. Test-time scaling improves LLM reasoning by allocating additional computation during inference without updating the underlying model. A straightforward paradigm is parallel sampling, including self-consistency and Best-of-N, which generates multiple reasoning trajectories and aggregates or selects promising solutions (Wang et al., 2022; Snell et al., 2024). More structured approaches explicitly explore alternative reasoning paths, such as Tree of Thoughts (ToT) (Yao et al., 2023) and Monte Carlo Tree Search (MCTS)-based methods (Hao et al., 2023; Zhou et al., 2023; Qi et al., 2025). Another line of work employs

process reward models (PRMs) to guide inference (Sun et al., 2024; Liao et al., 2025; Liu et al., 2025). More recently, confidence- and uncertainty-aware methods dynamically allocate test-time computation according to model confidence, entropy, or reasoning progress, selectively expanding, pruning, or terminating reasoning paths (Ding et al., 2025; Li et al., 2025; Fu et al., 2024; Xu et al., 2025b; Scalena et al., 2025; Zhang et al., 2025d). Moreover, NCoTS (Ling et al., 2026) formulates reasoning as a search over discrete reasoning operators and uses learned potential and progress estimators to select promising reasoning directions, whereas GUARD (Zhu et al., 2026) detects high-risk transitions through entropy and performs targeted short-horizon branching.

Training-Based Efficient Reasoning. A growing body of work improves reasoning efficiency through post-training, primarily using reinforcement learning (RL) or supervised fine-tuning (SFT). RL-based approaches design correctness-conditioned or adaptive length rewards to improve the trade-off between reasoning accuracy and generation length (Xu et al., 2025a; Team et al., 2025; Yeo et al., 2025; Aggarwal & Welleck, 2025; Arora & Zanette, 2026; Zhang et al., 2025e; Dumitru et al., 2025; Zhang et al., 2025c; Hou et al., 2026; Liu et al., 2026; Lou et al., 2025; Jiang et al., 2026; Dai et al., 2026; Fang et al., 2026; Gao et al., 2026; Tu et al., 2026; Cheng et al., 2026; Nie et al., 2026). SFT-based approaches construct concise reasoning demonstrations and fine-tune models to internalize more efficient reasoning behaviors. Representative strategies include compressing existing long reasoning traces (Kang et al., 2025b; Xia et al., 2025), training models to skip unnecessary reasoning steps (Liu et al., 2024), learning from explicitly controlled token budgets (Han et al., 2025), and selecting short correct trajectories from sampled solutions as supervision (Munkhbat et al., 2025). Collectively, these approaches improve reasoning efficiency by modifying the model's post-training objective or training distribution.

Adaptive Early Stopping for Reasoning. Another line of work improves inference efficiency by terminating reasoning once further computation is deemed unnecessary. Training-free methods monitor signals such as model confidence or answer consistency during generation and trigger early exit when a predefined criterion is satisfied (Fu et al., 2024; Wang et al., 2025; Mao et al., 2025; Yang et al., 2026). Alternatively, learning-based approaches train lightweight probes or classifiers to predict when reasoning can be safely terminated (Zhang et al., 2025a; Liu & Wang, 2025; Wu et al., 2025; Jiang et al., 2025b; Yang et al., 2025b; Nagle et al., 2026).

3 METHOD

3.1 OVERVIEW AND KEY REASONING POINTS

We view a reasoning trace as a sequence of tokens whose trajectory is shaped by a small number of consequential transition points. Our method targets two complementary failure modes at such points: *which reasoning direction to take next* and *when to stop reasoning*. We therefore augment a frozen base language model with two lightweight modules: (i) a *decision-token head* that reranks plausible reasoning operators at uncertain transitions, and (ii) a *stopping head* that predicts when a correct answer has already emerged and further reasoning is unnecessary. Both heads are trained from rollouts generated by the base model itself; no stronger teacher model is required. We refer to this framework as SeDER (Self-teaching Decision and Exit for Reasoning).

Let a problem be q and a generated reasoning trace be $\boldsymbol{x} = (x_1, \ldots, x_N)$. We define a set of reasoning operators $\mathcal{O}$ (e.g., `Therefore`, `But`, `Wait`); surface-form variants that differ only in capitalization or leading whitespace are mapped to the same canonical operator (the full operator set and mapping are given in Appendix A.1). Following Ling et al. (2026), we treat these reasoning operators as important control points that help guide the reasoning. At position t, an operator token is treated as an *uncertain decision point* when its generation probability is below a threshold,

$$p_\theta(x_t \mid q, x_{<t}) < \tau_{\text{conf}}, \tag{1}$$

optionally together with a small margin between the two most likely canonical operators. Relatedly, Liu et al. (2025) partitions reasoning traces at low-confidence tokens and demonstrates the effectiveness of this approach. We focus the decision-token head on these low-confidence operator transitions, where a local change in reasoning direction is most likely to matter.

3.2 DECISION-TOKEN HEAD

Reference rollouts and candidate operators. For each problem, we sample multiple reasoning rollouts from the base model, retain the correct rollouts, and choose the shortest correct rollout as the reference trajectory $\boldsymbol{x}^\star$. For every uncertain decision point t in this trajectory, the sampled operator is denoted $o_t^+ \in \mathcal{O}$. We aggregate the base-model probability mass of surface-form tokens into canonical operators and select up to K highest-probability alternative operators whose probability exceeds τ_{alt}. This yields a candidate set $\mathcal{C}_t$ containing o_t^+ and plausible counterfactual alternatives.

Counterfactual local utility. For each candidate operator $o \in \mathcal{C}_t$, we replace the reference operator at position t with o and evaluate the resulting local continuation using two signals: *alignment* with the successful reference continuation and *stability* of short stochastic continuations.

Let $\tilde{x}_{\le t}^{(o)} = (x_1^\star, \ldots, x_{t-1}^\star, o)$ be the counterfactual prefix, and let $\boldsymbol{b}_t = (x_{t+1}^\star, \ldots, x_{t+L}^\star)$ denote the next L tokens of the successful reference trajectory. The alignment score is the average teacher-forced log-likelihood of this continuation under the same base model:

$$A_t(o) = \frac{1}{|\boldsymbol{b}_t|} \sum_{\ell=1}^{|\boldsymbol{b}_t|} \log p_\theta\left(b_{t,\ell} \mid q, \tilde{x}_{\le t}^{(o)}, b_{t,<\ell}\right). \tag{2}$$

A larger $A_t(o)$ means that replacing the original operator by o remains compatible with the continuation of a successful reasoning trace.

To measure stability, we sample B short branches of L tokens after the counterfactual prefix $\tilde{x}_{\le t}^{(o)}$. Let $p_{o,r}^{(\ell)}$ be the next-token distribution at step $\ell \in \{1, \ldots, L\}$ in branch r. For branches r and s,

$$\mathrm{JS}\left(p_{o,r}^{(\ell)}, p_{o,s}^{(\ell)}\right) = \frac{1}{2} D_{\mathrm{KL}}\left(p_{o,r}^{(\ell)} \middle\| m\right) + \frac{1}{2} D_{\mathrm{KL}}\left(p_{o,s}^{(\ell)} \middle\| m\right), m = \frac{1}{2}\left(p_{o,r}^{(\ell)} + p_{o,s}^{(\ell)}\right). \tag{3}$$

We average this divergence over valid local steps and all branch pairs and negate it:

$$S_t(o) = -\frac{1}{|\mathcal{P}_t(o)|} \sum_{(r,s,\ell) \in \mathcal{P}_t(o)} \mathrm{JS}\left(p_{o,r}^{(\ell)}, p_{o,s}^{(\ell)}\right), \tag{4}$$

so that a larger $S_t(o)$ indicates more stable local behavior. The candidate utility is

$$U_t(o) = \alpha S_t(o) + \beta A_t(o), \tag{5}$$

where α and β are configurable coefficients.

Preference construction. For each alternative $o^- \in \mathcal{C}_t \setminus \{o_t^+\}$, we compute

$$\Delta U_t(o^-) = U_t(o_t^+) - U_t(o^-). \tag{6}$$

We use this utility gap as the preference weight, $w_t(o^-) = \Delta U_t(o^-)$, which measures the strength of preference for the reference operator over the alternative. We emit the pair $(x_{<t}^\star, o_t^+, o^-, w_t)$ only when $w_t > m_{\text{pair}}$. Thus, training focuses on counterfactual comparisons for which the reference operator has sufficiently stronger local evidence. Figure 2 illustrates the preference construction step and the role of the utility gap.

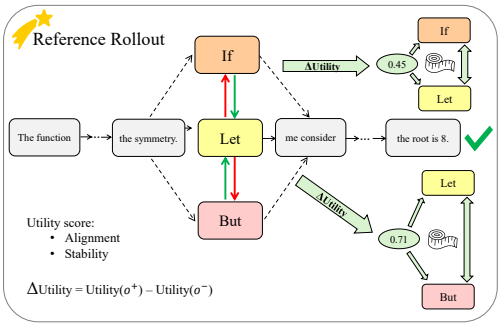


Figure 2: Construction of decision-token pairs.

Decision-token head training. The base language model is frozen. To control cost and emphasize the recent reasoning state, we retain at most the most recent M tokens of the pre-decision prefix,

$$\bar{\boldsymbol{x}}_{<t} = \mathrm{suffix}_M(\boldsymbol{x}_{<t}^\star). \tag{7}$$

Let $h_t \in \mathbb{R}^d$ denote the final non-padding hidden state obtained from the frozen model on $(q, \bar{\boldsymbol{x}}_{<t})$; this is the causal state immediately before the operator at position t is generated. The decision-token head is a bias-free linear map $W \in \mathbb{R}^{|\mathcal{O}| \times d}$,

$$s_\phi(o \mid q, \bar{\boldsymbol{x}}_{<t}) = W_o^\top h_t. \tag{8}$$

Rows of W are initialized by averaging the corresponding output-head vectors of all surface-form token IDs mapped to each canonical operator (Appendix A.1). Only W is optimized.

For a preference pair (o^+, o^-, w), we use a weighted pairwise logistic loss,

$$\mathcal{L}_{\text{decision}} = \frac{1}{B_{\text{tr}}} \sum_{i=1}^{B_{\text{tr}}} w_i \, \zeta\big(m_{\text{tr}} - s_\phi(o_i^+) + s_\phi(o_i^-)\big), \tag{9}$$

where $\zeta(z) = \log(1 + \exp(z))$ is the softplus function and m_{tr} is an optional training margin. For $m_{\text{tr}} = 0$, Eq. 9 is equivalent to maximizing $\sigma(s_\phi(o^+) - s_\phi(o^-))$. The weight w is given by the utility gap in Eq. 6, so pairs with larger utility differences receive greater emphasis during training. Hence the head learns a context-dependent preference over reasoning operators.

3.3 Stopping Head

The stopping head addresses a complementary question: whether the model has already reached a correct answer and should terminate its internal reasoning. Inspired by Nagle et al. (2026), we train a classifier to detect when reasoning can terminate early. Specifically, we generate multiple rollouts and retain only those whose final answers are correct. For each successful rollout, we locate the earliest token index $t^\star$ at which the generated prefix already contains an answer matching the gold answer under the task-specific answer-matching procedure. We then assign token-level labels

$$y_t = \mathbb{I}[t \geq t^\star], \tag{10}$$

within the reasoning span; tokens outside that span are masked from the loss.

The frozen base model produces token representations $H = (h_1, \ldots, h_T)$. The stopping module consists of a decoder block g_ψ, initialized by copying the final decoder block of the base model, followed by a binary linear classifier (w_s, b_s):

$$\tilde{h}_t = g_\psi(H)_t, \qquad z_t = w_s^\top \tilde{h}_t + b_s, \qquad p_{\text{stop},t} = \sigma(z_t). \tag{11}$$

The copied block and classifier are trainable while the base model remains frozen. With a binary mask m_t indicating valid reasoning tokens, the loss is masked binary cross-entropy,

$$\mathcal{L}_{\text{stop}} = \frac{\sum_t m_t \, \text{BCEWithLogits}(z_t, y_t)}{\sum_t m_t}. \tag{12}$$

Long traces are split into bounded-length windows for training; details are given in Appendix A.2.

3.4 Inference

During autoregressive generation, the base model first proposes a provisional next token $\tilde{x}_t$ from the current sampling distribution conditioned on $(q, x_{<t})$. We invoke the decision-token head only when this proposal is an uncertain reasoning transition. Specifically, let $\tilde{o}_t$ denote the canonical operator associated with $\tilde{x}_t$, and let $\hat{p}_t(\tilde{x}_t)$ be its proposal probability. The token is eligible for intervention if $\tilde{o}_t \in \mathcal{O}$ and

$$\hat{p}_t(\tilde{x}_t) < \tau_{\text{conf}}, \qquad \Delta_t^{\text{op}} = P_{\text{LM},t}^{(1)} - P_{\text{LM},t}^{(2)} \leq \tau_{\text{margin}}, \tag{13}$$

where $P_{\text{LM},t}^{(1)}$ and $P_{\text{LM},t}^{(2)}$ are the two largest canonical-operator probability masses under the base model. Importantly, the provisional token is not appended before reranking. We retain at most the most recent M tokens of the pre-decision prefix,

$$\bar{\boldsymbol{x}}_{<t} = \text{suffix}_M(\boldsymbol{x}_{<t}). \tag{14}$$

so that the decision-token head uses the same pre-decision context length used during training.

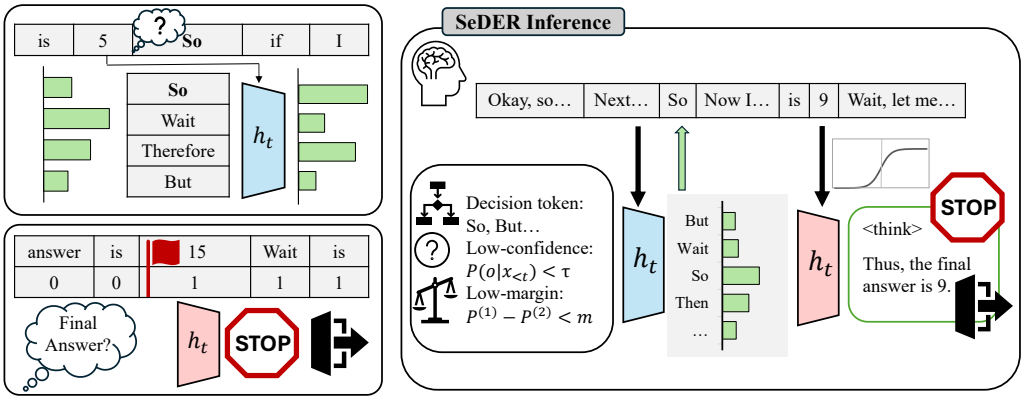


Figure 3: Overview of SeDER. The decision-token head learns operator preferences, while the stopping head detects when the correct answer first emerges. During inference, they guide uncertain reasoning transitions and enable early exit from reasoning.

Let $P_{\mathrm{LM}}(o \mid q, x_{<t})$ be the base-model probability of canonical operator o, obtained by summing the full-vocabulary probability mass over its surface-form token IDs. We retain the top-K canonical operators as the candidate set and compute the decision-token-only reranking score

$$R_t(o) = \log P_{\mathrm{LM}}(o \mid q, x_{<t}) + \lambda_{\mathrm{decision}}\, s_\phi(o \mid q, \bar{\boldsymbol{x}}_{<t}), \tag{15}$$

where $s_\phi(o \mid q, \bar{\boldsymbol{x}}_{<t})$ is defined in Eq. 8, $\lambda_{\mathrm{decision}}$ controls the contribution of the learned decision-token score, and we set $\lambda_{\mathrm{decision}} = 1$ in all experiments. The final operator is greedily selected or sampled from these reranked scores and only then appended to the sequence. Thus the learned decision-token preference modifies the base model's operator distribution only at uncertain decision points, while the original generation is left unchanged elsewhere.

The stopping head operates only while the model is inside its reasoning span. After each generated token, we evaluate the probability $p_{\mathrm{stop},t}$ defined in Eq. 11 on a bounded recent context and form a binary vote $v_t = \mathbb{I}[p_{\mathrm{stop},t} \geq \tau_s]$. To avoid premature termination from an isolated high score, we trigger stopping only when a sufficient number of recent votes are positive. When the criterion fires, we explicitly append the tokenization of the end-of-reasoning delimiter `</think>` to the generated sequence. This closes the reasoning span, disables further stop-head intervention for that span, and returns generation to the frozen base model, which continues autoregressively to produce the final answer. Hence the stopping head performs an *early exit from reasoning*, rather than terminating response generation altogether. Figure 3 provides an overview of SeDER, illustrating how the decision-token and stopping heads are trained and applied to control reasoning during inference.

4 Experiments

4.1 Experimental Setup

Training. We train the decision-token and stopping heads for DeepSeek-R1-Distill-Qwen-1.5B and DeepSeek-R1-Distill-Qwen-7B (Guo et al., 2025; Yang et al., 2025a). We randomly sample 5,000 questions from the DeepScaleR-Preview dataset (Luo et al., 2025) and generate four rollouts per question to construct the decision-token preference pairs and stopping labels described in Section 3. For data construction, we set the local branch length in the utility function of Eq. 5 to $L = 32$, with both the alignment and stability coefficients set to 1. We use a confidence threshold of 0.2 and a margin threshold of 0.3 for identifying uncertain decision points as defined in Eq. 1. We train both heads on the resulting data and use a prefix window of $M = 512$ tokens for the decision-token head, following Eq. 7. Data generation and training are performed on four NVIDIA H200 GPUs. Additional training details are provided in Appendix B.1.

Evaluation. We evaluate our method on four mathematical reasoning benchmarks with varying levels of difficulty: MATH-500 (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), AMC23,

and AIME24. We generate four rollouts per problem for MATH-500 and GSM8K and eight rollouts for AMC23 and AIME24. We use a sampling temperature of 0.6 and top-p of 0.95. The maximum generation length is 16,384 tokens for MATH-500 and GSM8K and 32,768 tokens for AMC23 and AIME24. We use a prefix window of $M = 512$ tokens as defined in Eq. 14, together with a confidence threshold of 0.2 and a margin threshold of 0.3 in Eq. 13. Under these settings, uncertain decision tokens account for approximately 2% of all generated tokens. This is consistent with the 2% rate adopted by AdaptiveStep (Liu et al., 2025), which motivates this choice by relating it to the proportion of deliberate, effortful thinking in human cognition reported by Kahneman (2011). We set the stopping threshold to 0.7, using a relatively conservative value to reduce the risk of premature termination and prioritize reasoning accuracy. We report Pass@1 accuracy (Acc.) and average reasoning-trace length. Additional evaluation details are provided in Appendix B.2.

Baselines. We compare our method against six baselines: the original DeepSeek-R1-Distill models, supervised fine-tuning (SFT), AdaptThink (Zhang et al., 2025c), LASER (Liu et al., 2026), LC-R1 (Cheng et al., 2026), and NCoTS (Ling et al., 2026). For SFT, we use the same rollouts employed to construct our decision-token training data, selecting the shortest correct rollout as the training target. Additional SFT training details are provided in Appendix B.3.

4.2 Main Results

Table 1 compares the accuracy and reasoning-trace length of our method with the baselines for both DeepSeek-R1-Distill-Qwen-1.5B and 7B. Our method SeDER consistently achieves the highest accuracy across all benchmarks for both model sizes. The improvement over the original model is generally larger for the 1.5B model than for the 7B model, likely because the stronger 7B model leaves less room for improvement. We also observe larger gains on more challenging benchmarks such as AIME24 and AMC23 than on the relatively easier GSM8K.

Our method also produces shorter reasoning traces than the original model across all benchmarks and both model sizes. The degree of reduction varies across datasets: it is particularly substantial on MATH-500 and GSM8K, reaching approximately 50% on GSM8K with the 7B model, while the reduction is smaller on AIME24. Our traces remain longer than those of several efficiency-oriented baselines. This is partly because we use a fixed, conservative stopping threshold of 0.7 across all models and benchmarks to avoid aggressive early exits that may hurt accuracy. Moreover, unlike RL-based approaches such as AdaptThink, LASER and LC-R1, which directly modify the model's reasoning behavior through post-training, our method keeps the base model frozen and relies only on lightweight auxiliary heads, substantially reducing training overhead. The additional inference overhead introduced by SeDER is small: the auxiliary heads contain only a tiny fraction of the parameters of the base model, decision-token interventions occur sparsely, and the stopping head can further offset this cost by terminating reasoning early.

4.3 Ablation Study

We conduct ablation studies covering both data construction and inference strategies. Since the decision-token head is the primary component of our method, we focus mainly on its design choices. We report results for the 1.5B model on MATH-500 and AMC23 in the main text, with 7B results provided in Appendix C.

Utility score. As described in Section 3.2, the utility score used to construct preference pairs consists of alignment and stability components. We evaluate the contribution of each component by setting its coefficient to zero in Eq. 5 while keeping all other training and inference settings unchanged. Table 2 reports the resulting accuracy and reasoning length. Removing either component reduces accuracy, with a larger degradation when the alignment score is removed. This suggests that alignment with the successful reference continuation provides the primary preference signal, while stability serves as a complementary measure of the reliability of alternative reasoning directions.

Prefix window size. We use a 512-token look-back prefix window for the decision-token head during both training and inference, as defined in Eqs. 7 and 14. To examine the effect of this choice, we compare window sizes of 128, 512, and 1,024 tokens. As shown in Table 3, a window size of 512 achieves the best overall performance. Both shorter and longer contexts lead to modest accuracy

Table 1: Comparison of reasoning performance and generation length across benchmarks. SeDER consistently achieves the highest accuracy across all benchmarks for both models.

Method	MATH-500		AMC23		AIME24		GSM8K	
	Acc	Length	Acc	Length	Acc	Length	Acc	Length
DeepSeek-R1-Distill-Qwen-1.5B								
Original	83.9	5042	71.6	8202	28.9	15956	83.4	2294
SFT	80.5	4645	65.6	8386	30.8	20466	83.1	2048
AdaptThink	82.0	1782	67.0	3510	31.0	6679	83.1	480
LASER	85.6	2736	75.9	4162	31.5	6589	82.3	1064
LC-R1	82.5	2233	61.7	3947	23.6	7122	82.7	841
NCoTS	81.5	3180	67.5	3768	33.3	6897	85.4	954
Ours (SeDER)	**87.1**	3166	**77.2**	4504	**36.9**	9192	**87.4**	1377
DeepSeek-R1-Distill-Qwen-7B								
Original	92.6	4017	88.4	6433	53.1	13414	91.6	1842
SFT	91.6	3442	89.1	6953	57.5	13623	92.3	1484
AdaptThink	92.0	1875	87.5	3287	55.6	8599	90.9	309
LASER	92.2	2317	89.7	3733	54.4	6320	93.0	968
LC-R1	90.4	1568	79.1	3453	52.7	6087	88.1	450
NCoTS	91.9	3012	81.5	3459	51.0	7238	92.6	899
Ours (SeDER)	**93.7**	2617	**92.2**	4508	**59.7**	11396	**93.8**	947

Table 2: Ablation study of the utility score. Original denotes the base model, while Ours (SeDER) denotes the full setting used in the main results experiments.

Dataset	Metric	Original	Ours (SeDER)	No stability	No alignment
MATH-500	Acc	83.9	**87.1**	86.9	85.6
	Length	5042	3166	3116	3155
AMC23	Acc	71.6	**77.2**	75.0	72.5
	Length	8202	4504	5047	4664

decreases, with the degradation being more pronounced at 128 tokens. This suggests that a moderate context window provides sufficient information about the recent reasoning state without requiring a substantially longer history.

Inference strategy. We further examine how the choice of intervention positions affects performance. In our default strategy, the decision-token head intervenes only at low-confidence, high-uncertainty operator positions, while the stopping head determines when to terminate reasoning. We compare against three variants. *Intervene-All* applies the decision-token head to nearly all reasoning-operator positions by relaxing both the confidence and margin thresholds to 0.9. *Intervene-Step* instead applies the head at decision tokens following reasoning-step boundaries marked by the delimiter "\n\n", following prior work (Yang et al., 2025c; Ling et al., 2026). Finally, *No-Stop* disables the stopping head while retaining our default decision-token intervention strategy.

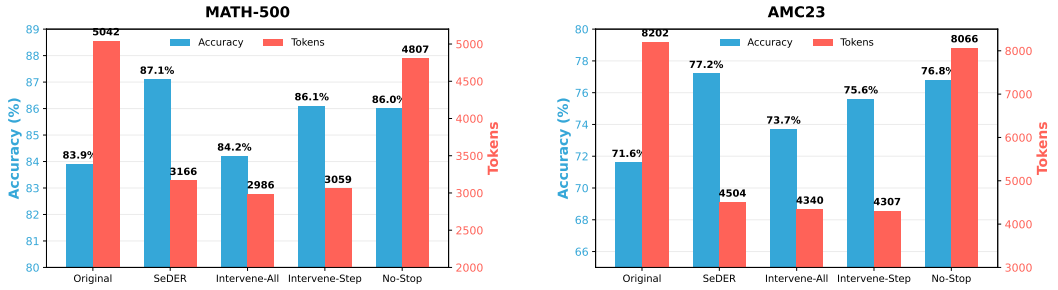
Figure 4 compares these variants with our default method. Confidence- and uncertainty-based intervention achieves the highest accuracy. Intervene-Step performs slightly worse, whereas Intervene-All results in a larger accuracy drop. The first two strategies intervene at similar frequencies, approximately 2% of generated tokens, while Intervene-All increases the intervention rate to approximately 7%. This suggests that indiscriminate intervention at additional operator positions can be counterproductive. Disabling the stopping head has little effect on accuracy but substantially increases reasoning length, demonstrating its role in reducing redundant reasoning.

4.4 TRANSFERABILITY

We evaluate whether our method generalizes beyond the distribution of the training data. We apply the same trained decision-token and stopping heads, without additional training, to GPQA Diamond, an out-of-domain benchmark containing multiple-choice questions in physics, chemistry, and biology (Rein et al., 2024). We use the same evaluation settings as above, with four rollouts per problem

Table 3: Ablation study of the decision-token head prefix window size. Original denotes the base model, and SeDER-512 denotes SeDER with a 512-token prefix window.

Dataset	Metric	Original	SeDER-128	SeDER-512	SeDER-1024
MATH-500	Acc	83.9	86.2	**87.1**	86.8
	Length	5042	3235	3166	3193
AMC23	Acc	71.6	75.9	**77.2**	76.6
	Length	8202	5111	4504	4567



(a) MATH-500 (b) AMC23

Figure 4: Ablation study of different inference strategies. Blue bars show accuracy, while red bars show the average reasoning-trace length.

and a maximum generation length of 16,384 tokens. We compare against the original model, SFT, LASER, and NCoTS. Table 4 reports accuracy and reasoning-trace length. For the 1.5B model, our method achieves slightly lower accuracy than LASER, whereas for the 7B model it achieves the highest accuracy. These results suggest that the learned reasoning control can transfer beyond the training domain, although the smaller model appears more sensitive to distribution shift.

Table 4: Transferability results on GPQA Diamond. 1.5B and 7B denote DeepSeek-R1-Distill-Qwen-1.5B and DeepSeek-R1-Distill-Qwen-7B, respectively.

Model	Metric	Original	SFT	LASER	NCoTS	SeDER
1.5B	Acc	33.8	31.9	**35.0**	33.5	34.4
	Length	7300	6953	6000	6017	6247
7B	Acc	44.4	42.7	46.3	42.5	**46.8**
	Length	6600	6785	3789	3992	4808

5 Conclusion

We introduced SeDER, a lightweight self-teaching framework for controlling both the direction and length of reasoning. A decision-token head redirects uncertain reasoning transitions, while a stopping head detects when sufficient reasoning has been completed. Both are trained from the base model's own successful rollouts while keeping the underlying model frozen. Across multiple mathematical reasoning benchmarks, SeDER improves accuracy while reducing reasoning length.

Future work includes jointly modeling decision and stopping behavior within a unified reasoning-control mechanism, rather than using separate heads. Another promising direction is to move beyond discrete surface-form decision tokens and explore reasoning control directly in latent representations. Extending SeDER to broader reasoning domains and larger models is also of interest.

AI USE STATEMENT

In this work, we used generative AI tools for design or provide feedback on research methodology or experiments, implement methods. We have not used generative AI tools for generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, clean and reformat dataset, support qualitative and thematic data analysis, interpret results, and assist in the writing of proofs, assist with translation are not applicable to this work. Additionally, we used generative AI tools for create or modify scientific figures or images, suggest experimental parameters, create or edit software code, draft parts of a research paper, edit a research paper to improve readability. We have reviewed all AI-assisted work. For example, LLM-generated code was verified and tested for correctness by the author. Feedback and suggestions provided by LLMs on research methodology and experimental design were reviewed and validated by the author before adoption. Experimental parameters proposed by LLMs were evaluated and selected by the author based on empirical results. In addition, sections of the paper drafted or revised with the assistance of LLMs were carefully proofread and edited by the author. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

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

Daman Arora and Andrea Zanette. Training language models to reason efficiently. *Advances in Neural Information Processing Systems*, 38:60770–60808, 2026.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *Science China Information Sciences*, 69(6):161101, 2026.

Zhengxiang Cheng, Dongping Chen, Mingyang Fu, and Tianyi Zhou. Optimizing length compression in large reasoning models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3232–3250, 2026.

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

Muzhi Dai, Chenxu Yang, and Qingyi Si. S-grpo: Early exit via reinforcement learning in reasoning models. *Advances in Neural Information Processing Systems*, 38:48178–48204, 2026.

Yifu Ding, Wentao Jiang, Shunyu Liu, Yongcheng Jing, Jinyang Guo, Yingjie Wang, Jing Zhang, Zengmao Wang, Ziwei Liu, Bo Du, et al. Dynamic parallel tree search for efficient llm reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11233–11252, 2025.

Razvan-Gabriel Dumitru, Darius Peteleaza, Vikas Yadav, and Liangming Pan. Conciserl: Conciseness-guided reinforcement learning for efficient reasoning models. In *EMNLP (Findings)*, pp. 17099–17123, 2025.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. Thinkless: Llm learns when to think. *Advances in neural information processing systems*, 38:151268–151295, 2026.

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

Jiaxuan Gao, Shu Yan, Qixin Tan, Shusheng Xu, Wei Fu, Zhiyu Mei, Kaifeng Lyu, Yi Wu, et al. How far are we from optimal reasoning efficiency? *Advances in Neural Information Processing Systems*, 38:10916–10946, 2026.

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

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24842–24855, 2025.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. Reasoning with language model is planning with world model. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 8154–8173, 2023.

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

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. Thinkprune: Pruning long chain-of-thought of LLMs via reinforcement learning. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=V51gPu1uQD.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Gangwei Jiang, Yahui Liu, Zhaoyi Li, Wei Bi, Fuzheng Zhang, Linqi Song, Ying Wei, and Defu Lian. What makes a good reasoning chain? uncovering structural patterns in long chain-of-thought reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 6501–6525, 2025a.

Guochao Jiang, Guofeng Quan, Zepeng Ding, Ziqin Luo, Dixuan Wang, and Zheng Hu. Flashthink: An early exit method for efficient reasoning. *arXiv preprint arXiv:2505.13949*, 2025b.

Lingjie Jiang, Xun Wu, Shaohan Huang, Qingxiu Dong, Zewen Chi, Li Dong, Xingxing Zhang, Tengchao Lv, Lei Cui, and Furu Wei. Think only when you need with large hybrid-reasoning models. *Advances in Neural Information Processing Systems*, 38:159355–159382, 2026.

Daniel Kahneman. *Thinking, Fast and Slow*. Farrar, Straus and Giroux, New York, 2011.

Liwei Kang, Yue Deng, Yao Xiao, Zhanfeng Mo, Wee Sun Lee, and Lidong Bing. First try matters: Revisiting the role of reflection in reasoning models. *arXiv preprint arXiv:2510.08308*, 2025a.

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. C3ot: Generating shorter chain-of-thought without compromising effectiveness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 24312–24320, 2025b.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

Peiji Li, Kai Lv, Yunfan Shao, Yichuan Ma, Linyang Li, Xiaoqing Zheng, Xipeng Qiu, and Qipeng Guo. Fastmcts: A simple sampling strategy for data synthesis. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 24405–24422, 2025.

Baohao Liao, Yuhui Xu, Hanze Dong, Junnan Li, Christof Monz, Silvio Savarese, Doyen Sahoo, and Caiming Xiong. Reward-guided speculative decoding for efficient LLM reasoning. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 37555–37572. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/liao25f.html.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Guoming Ling, Zhongzhan Huang, Yupei Lin, Junxin Li, Shanshan Zhong, Hefeng Wu, and Liang Lin. Neural chain-of-thought search: Searching the optimal reasoning path to enhance large language models. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 22900–22933, 2026.

Tengxiao Liu, Qipeng Guo, Xiangkun Hu, Cheng Jiayang, Yue Zhang, Xipeng Qiu, and Zheng Zhang. Can language models learn to skip steps? *Advances in Neural Information Processing Systems*, 37:45359–45385, 2024.

Wei Liu, Ruochen Zhou, Yiyun Deng, Yuzhen Huang, Junteng Liu, Yuntian Deng, Yizhe Zhang, and Junxian He. Learn to reason efficiently with adaptive length-based reward shaping. In *International Conference on Learning Representations*, volume 2026, pp. 43279–43308, 2026.

Xin Liu and Lu Wang. Answer convergence as a signal for early stopping in reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 17907–17918, 2025.

Yuliang Liu, Junjie Lu, Chaofeng Qu, Zhaoling Chen, Zefan Cai, Jason Klein Liu, Chonghan Liu, Yunhui Xia, Li Zhao, Jiang Bian, Chuheng Zhang, Wei Shen, and Zhouhan Lin. AdaptiveStep: Automatically dividing reasoning step through model confidence. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 39016–39031. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/liu25aq.html.

Chenwei Lou, Zewei Sun, Xinnian Liang, Meng Qu, Wei Shen, Wenqi Wang, Yuntao Li, Qingping Yang, and Shuangzhi Wu. Adacot: Pareto-optimal adaptive chain-of-thought triggering via reinforcement learning. *arXiv preprint arXiv:2505.11896*, 2025.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. Notion page, 2025.

Minjia Mao, Bowen Yin, Yu Zhu, and Xiao Fang. Early stopping chain-of-thoughts in large language models. *arXiv preprint arXiv:2509.14004*, 2025.

Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. Self-training elicits concise reasoning in large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 25127–25152, 2025.

Alliot Nagle, Jakhongir Saydaliev, Dhia Garbaya, Michael Gastpar, Ashok Vardhan Makkuva, and Hyeji Kim. Terminator: Learning optimal exit points for early stopping in chain-of-thought reasoning. *arXiv preprint arXiv:2603.12529*, 2026.

Shuaiyi Nie, Wenyuan Zhang, Linhao Yu, Tianmeng Yang, Yao Chen, Weichong Yin, Yu Sun, Hua Wu, Tingwen Liu, et al. Attnpo: Attention-guided process supervision for efficient reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 39728–39748, 2026.

Zhenting Qi, Mingyuan Ma, Jiahang Xu, Li Lyna Zhang, Fan Yang, and Mao Yang. Mutual reasoning makes smaller llms stronger problem-solver. In *International Conference on Learning Representations*, volume 2025, pp. 20788–20807, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Daniel Scalena, Leonidas Zotos, Elisabetta Fersini, Malvina Nissim, and Ahmet Üstün. Eager: Entropy-aware generation for adaptive inference-time scaling. *arXiv preprint arXiv:2510.11170*, 2025.

Changshuo Shen, Leheng Sheng, Yuxin Chen, An Zhang, and Xiang Wang. Reasoning can be restored by correcting a few decision tokens. *arXiv preprint arXiv:2605.16874*, 2026.

Parshin Shojaee, Iman Mirzadeh, Maxwell Horton, Samy Bengio, Mehrdad Farajtabar, et al. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. *Advances in Neural Information Processing Systems*, 38:108018–108059, 2026.

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

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, et al. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang, Jiahao Qiu, Ming Yin, Mengdi Wang, Peter Bartlett, and Andrea Zanette. Fast best-of-n decoding via speculative rejection. *Advances in Neural Information Processing Systems*, 37:32630–32652, 2024.

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

Songjun Tu, Jiahao Lin, Qichao Zhang, Xiangyu Tian, Linjing Li, Xiangyuan Lan, and Dongbin Zhao. Learning when to think: Shaping adaptive reasoning in r1-style models via multi-stage rl. *Advances in Neural Information Processing Systems*, 38:15181–15207, 2026.

Xi Wang, James McInerney, Lequn Wang, and Nathan Kallus. Entropy after `<\Think>` for reasoning model early exiting. *ArXiv*, abs/2509.26522, 2025. URL `https://api.semanticscholar.org/CorpusID:281681359`.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

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

Menghua Wu, Cai Zhou, Stephen Bates, and Tommi Jaakkola. Thought calibration: Efficient and confident test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 14302–14316, 2025.

Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. Tokenskip: Controllable chain-of-thought compression in llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 3351–3363, 2025.

Fengli Xu, Qianyue Hao, Chenyang Shao, Zefang Zong, Yu Li, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, et al. Toward large reasoning models: A survey of reinforced reasoning with large language models. *Patterns*, 6(10), 2025a.

Zicheng Xu, Xiuyi Lou, Guanchu Wang, Yu-Neng Chuang, Feng Luo, Guangyao Zheng, Alexander S Szalay, Zirui Liu, and Vladimir Braverman. Dts: Enhancing large reasoning models via decoding tree sketching. *arXiv preprint arXiv:2511.00640*, 2025b.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weipinng Wang. Dynamic early exit in reasoning models. In *International Conference on Learning Representations*, volume 2026, pp. 88170–88210, 2026.

Rubing Yang, Huajun Bai, Song Liu, Guanghua Yu, Runzhi Fan, Yanbin Dang, Jiejing Zhang, Kai Liu, Jianchen Zhu, and Peng Chen. Specexit: Accelerating large reasoning model via speculative exit. *arXiv preprint arXiv:2509.24248*, 2025b.

Wang Yang, Xiang Yue, Vipin Chaudhary, and Xiaotian Han. Speculative thinking: Enhancing small-model reasoning with large model guidance at inference time. *arXiv preprint arXiv:2504.12329*, 2025c.

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

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

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning models know when they're right: Probing hidden states for self-verification. *arXiv preprint arXiv:2504.05419*, 2025a.

Duzhen Zhang, Zhong-Zhi Li, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Xiuyi Chen, Yingying Zhang, et al. From system 1 to system 2: A survey of reasoning large language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025b.

Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. Adaptthink: Reasoning models can learn when to think. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 3716–3730, 2025c.

Jinghan Zhang, Xiting Wang, Fengran Mo, Yeyang Zhou, Wanfu Gao, and Kunpeng Liu. Entropy-based exploration conduction for multi-step reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 3895–3906, 2025d.

Xiaoyun Zhang, Jingqing Ruan, Xing Ma, Yawen Zhu, Haodong Zhao, Hao Li, Jiansong Chen, Ke Zeng, and Xunliang Cai. When to continue thinking: Adaptive thinking mode switching for efficient reasoning. *arXiv preprint arXiv:2505.15400*, 2025e.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning acting and planning in language models. *arXiv preprint arXiv:2310.04406*, 2023.

Wei Zhu, Jian Zhang, Lixing Yu, Kun Yue, and Zhiwen Tang. Dissecting failure dynamics in large language model reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8893–8914, 2026.

A Additional Method Details

A.1 Canonical Operator Mapping

Following Ling et al. (2026), the decision-token head operates over a predefined set of canonical reasoning operators. Our full operator set is

$$\begin{aligned}\mathcal{O} = \{&\texttt{Thus}, \texttt{Therefore}, \texttt{So}, \texttt{Then}, \texttt{Hence},\\ &\texttt{But}, \texttt{However}, \texttt{Wait}, \texttt{Hmm}, \texttt{Option}, \texttt{Maybe},\\ &\texttt{Alternatively}, \texttt{Alternative}, \texttt{Another},\\ &\texttt{If}, \texttt{Since}, \texttt{Given}, \texttt{Because}, \texttt{Let},\\ &\texttt{Now}, \texttt{First}, \texttt{Alright}, \texttt{The}, \texttt{I}, \texttt{Simplify}, \texttt{Yes}\}.\end{aligned}$$

Surface forms that differ only in leading whitespace or capitalization are mapped to the same canonical operator, except for the explicitly case-sensitive operators `I` and `First`. Each canonical operator may therefore correspond to multiple token IDs in the base vocabulary. During candidate construction and inference, their base-model probabilities are summed to obtain the canonical-operator probability. For initialization of the decision-token head, the output-head vectors of all token IDs mapped to the same canonical operator are averaged to initialize the corresponding row of W.

A.2 Stopping-Head Details

Answer-arrival detection. To construct stopping labels, we retain successful rollouts and identify the earliest token at which the generated reasoning prefix contains an answer equivalent to the gold answer. We first scan likely answer boundaries, including every 16 tokens and tokens containing common answer-boundary characters or markers such as `boxed`, `answer`, and `final`. At each candidate position, we extract and normalize the answer from the prefix and compare it with the gold answer using exact, multiple-choice, numeric, and symbolic matching. If no candidate boundary matches, we fall back to scanning all tokens. Once a matching region is found, we scan from the beginning of the reasoning span to locate the exact earliest matching token $t^\star$. Tokens before $t^\star$ receive label 0 and tokens from $t^\star$ onward receive label 1, with the loss restricted to the reasoning span.

Windowed training. Long reasoning traces are partitioned into windows of at most 2,048 tokens with a stride of 1,024 tokens. A window covering the transition around $t^\star$ is additionally retained when necessary so that the continue-to-stop transition is represented during training. Padding tokens and tokens outside the reasoning span are excluded from the stopping loss. The base model and decision-token head remain frozen while the copied decoder block and binary stopping classifier are optimized.

Inference and temporal voting. At inference time, the stopping head is evaluated after each generated token within the reasoning span using at most the most recent 2,048 tokens. A token produces a positive stop vote when $p_{\text{stop},t} \geq \tau_s$. We use a voting window of five recent predictions and trigger an early exit when at least three are positive. In our experiments, we set $\tau_s = 0.7$. Once triggered, SeDER appends the tokenization of `</think>`, closes the reasoning span, and returns generation to the frozen base model for final-answer generation.

B Additional Experimental Setup Details

B.1 Training

We use the same training data construction procedure described in Section 3. For each model, we sample 5,000 questions from DeepScaleR-Preview and generate four rollouts per question, with a maximum generation length of 6,000 tokens per rollout. We use a local branch length of $L = 32$, set both the alignment and stability coefficients to 1, and use confidence and margin thresholds of 0.2 and 0.3, respectively. This produces 42,676 decision-token preference pairs for DeepSeek-R1-Distill-Qwen-1.5B and 39,117 pairs for DeepSeek-R1-Distill-Qwen-7B.

For decision-token-head training, we use a prefix window of $M = 512$, a pairwise margin of $m_{\text{tr}} = 0$, a learning rate of 1×10^{-4}, weight decay of 0.01, and a per-GPU batch size of 8. Training is performed on four GPUs, corresponding to an effective batch size of 32, for 16 epochs. This gives approximately 21K and 20K optimization steps for the 1.5B and 7B models, respectively. The resulting pairwise preference accuracy is approximately 75% for the 1.5B model and 81% for the 7B model.

For stopping-head training, we use a maximum window length of 2,048 tokens with a stride of 1,024, yielding 29,164 and 27,716 training windows for the 1.5B and 7B models, respectively. We use a learning rate of 1×10^{-4}, weight decay of 0.01, and a per-GPU batch size of 2. Training is performed on four GPUs, corresponding to an effective batch size of 8, for 4 epochs. The resulting token-level classification accuracy is approximately 82% for both models.

Table 5: Ablation study of the utility score for the 7B model. Original denotes the base model, while Ours (SeDER) denotes the full setting used in the main results experiments.

Dataset	Metric	Original	Ours (SeDER)	No stability	No alignment
MATH-500	Acc	92.6	**93.7**	**93.7**	92.9
	Length	4017	2617	2658	2563
AMC23	Acc	88.4	92.2	**92.3**	91.5
	Length	6433	4508	4529	4477

B.2 Evaluation

We evaluate SeDER on four mathematical reasoning benchmarks: MATH-500 (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), AMC23, and AIME24. We generate four rollouts per problem for MATH-500 and GSM8K and eight rollouts for AMC23 and AIME24. We use a sampling temperature of 0.6 and top-p of 0.95. The maximum generation length is 16,384 tokens for MATH-500 and GSM8K and 32,768 tokens for AMC23 and AIME24.

For decision-token intervention, we use a prefix window of $M = 512$, a confidence threshold of 0.2, and an operator-margin threshold of 0.3. Under these settings, uncertain decision tokens account for approximately 2% of all generated tokens. We set the stopping threshold to 0.7 and use a majority-vote criterion over recent stop-head predictions, as described in Appendix A.2.

We additionally observe that decision-token intervention can occasionally lead to degenerate generations, such as repetitive token or phrase loops. Such cases are rare (approximately 1.5% of generations on MATH-500). When a repetition loop is detected, we discard the affected rollout and fall back to generation from the original frozen model. We report Pass@1 accuracy and average reasoning-trace length over the resulting outputs.

B.3 SFT Baseline

For the SFT baseline, we select the shortest correct rollout for each training problem, resulting in 2,966 training trajectories for the 1.5B model and 3,437 for the 7B model. For both model sizes, we use a learning rate of 1×10^{-5}, weight decay of 0.1, and a per-GPU batch size of 1. Training is performed on four GPUs with 8 gradient-accumulation steps, corresponding to an effective batch size of 32, for 4 epochs.

C Ablation Study

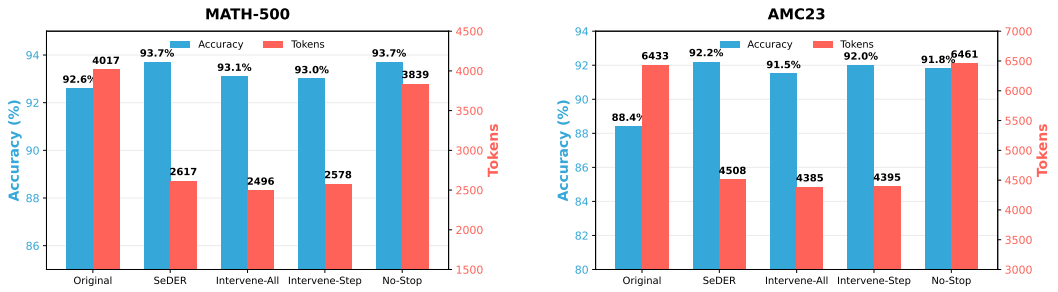
We conduct ablation studies covering both data construction and inference strategies. Since the decision-token head is the primary component of our method, we focus mainly on its design choices. We report results for the 7B model on MATH-500 and AMC23 in this section.

Utility score. As described in Section 3.2, the utility score used to construct preference pairs consists of alignment and stability components. We evaluate the contribution of each component by setting its coefficient to zero in Eq. 5 while keeping all other training and inference settings unchanged. Table 5 reports the resulting accuracy and reasoning length. The results for the 7B model show a similar overall pattern. Removing the alignment component leads to a modest decrease in accuracy, whereas removing the stability component has a smaller effect and can even slightly improve accuracy on AMC23. This suggests that alignment with the successful reference continuation provides the primary preference signal, while stability serves as a complementary measure of the reliability of alternative reasoning directions.

Prefix window size. We use a 512-token look-back prefix window for the decision-token head during both training and inference, as defined in Eqs. 7 and 14. To examine the effect of this choice, we compare window sizes of 128, 512, and 1,024 tokens. As shown in Table 6, a window size of 512 achieves the best overall performance. Both shorter and longer contexts lead to modest accuracy decreases.

Table 6: Ablation study of the decision-token head prefix window size for the 7B model. Original denotes the base model, and SeDER-512 denotes SeDER with a 512-token prefix window.

Dataset	Metric	Original	SeDER-128	SeDER-512	SeDER-1024
MATH-500	Acc	92.6	93.6	**93.7**	93.3
	Length	4017	2634	2617	2615
AMC23	Acc	88.4	91.6	**92.2**	91.4
	Length	6433	4415	4508	4417



(a) MATH-500 (b) AMC23

Figure 5: Ablation study of different inference strategies for the 7B model. Blue bars show accuracy, while red bars show the average reasoning-trace length.

Inference strategy. We further examine how the choice of intervention positions affects performance. In our default strategy, the decision-token head intervenes only at low-confidence, high-uncertainty operator positions, while the stopping head determines when to terminate reasoning. We compare against three variants. *Intervene-All* applies the decision-token head to nearly all reasoning-operator positions by relaxing both the confidence and margin thresholds to 0.9. *Intervene-Step* instead applies the head at decision tokens following reasoning-step boundaries marked by the delimiter "`\n\n`", following prior work (Yang et al., 2025c; Ling et al., 2026). Finally, *No-Stop* disables the stopping head while retaining our default decision-token intervention strategy.

Figure 5 compares these variants with our default method. Confidence- and uncertainty-based intervention performs slightly better than the other intervention strategies, although the differences are small for the 7B model. Disabling the stopping head has little effect on accuracy but substantially increases reasoning length, demonstrating its role in reducing redundant reasoning.

D Qualitative Analysis

In this section, we present a qualitative comparison between SeDER and the original DeepSeek-R1-Distill-Qwen-1.5B model (Table 7). SeDER identifies the first occurrence of a correct answer and injects `<think>` to terminate further reasoning, resulting in a reasoning length of only 545 tokens. In contrast, although the original model reaches the correct answer relatively early, it continues with multiple rounds of reflection and verification, leading to a substantially longer reasoning trace of 2602 tokens.

Table 7: Qualitative comparison of reasoning traces. SeDER reduces reasoning length by exiting immediately after the first correct answer is produced. The early-exit decision in SeDER and the subsequent reflection and verification process in the original model are highlighted in red.

Question: Find the product of $6_8 \cdot 7_8$. Express your answer in base 8.

SeDER (545 tokens): <think> Okay, so I need to multiply two octal numbers: 6 base 8 and 7 base 8, and then express the product in base 8. Hmm, octal means base 8, right? So each digit represents a power of 8. I remember that in base 10, whenI multiply numbers, I can do it the usual way, but here I need to be careful becausethe numbers are in base 8. Maybe I should convert them to base 10 first, multiply them, and then convertthe result back to base 8? That seems like a plan. Let me start by converting 6_8 to base 10. Since it's a single digit, 6 in base 8 is just 6 in base 10. Similarly, 7_8 is 7_{10}. So, 6 times 7 is 42 in base 10. Nowthe problem is, how do I convert 42 back to base 8? I think I need to divide 42 by 8 and find out how many times 8 goes into 42, and then take the remainder. So, 42 divided by 8 is 5 with a remainder of 2. So, 5 times 8 is 40, and adding the remainder gives me 42. So, 42 in base 10 is 52 in base 8 because $5 \times 8 + 2 = 42$. Therefore, 6_8 times 7_8 is 52_8. Wait, let<think> To find the product of $6_8 \cdot 7_8$ and express the answer in base 8, follow these steps. Convert each octal number to base 10: $6_8 = 6_{10}$ and $7_8 = 7_{10}$. Multiply the base-10 equivalents: $6_{10} \cdot 7_{10} = 42_{10}$. Convert the product back to base 8: $42 \div 8 = 5$ with remainder 2, so $42_{10} = 52_8$. Thus, the product of $6_8 \cdot 7_8$ in base 8 is $\boxed{52_8}$.

Original (2602 tokens): <think> Okay, so I need to multiply two numbers that are in base 8, which is octal, right? The numbers are 6_8 and 7_8, and I need to find their product and express the answer also in base 8. Hmm, I think I should start by converting them to base 10 because I'm more familiar with multiplying in base 10, and then I can convert the result back to base 8. But wait, maybe I can do it directly in base 8? I'm not sure if that's easier, but let me think. First, let me recall how multiplication works in different bases. In base 10, we multiply each digit and carry over the numbers as needed. In base 8, the same principles apply, but with digits from 0 to 7. So, perhaps I can perform the multiplication as if I'm in base 10 and then convert the result to base 8. That might be straightforward. Alright, let's write down the multiplication: $6_8 \times 7_8$. In base 10, 6_8 is equal to 6 in base 10, and 7_8 is also 7 in base 10. So, 6 multiplied by 7 is 42 in base 10. Now, I need to convert 42 back to base 8. To convert 42 to base 8, I can divide 42 by 8 and find the quotient and remainder. 42 divided by 8 is 5 with a remainder of 2. So, that means $42_{10} = 52_8$ because $5 \times 8 + 2 = 42$. Therefore, the product of 6_8 and 7_8 is 52_8. Wait, let me verify this because I want to make sure I didn't make a mistake. Maybe I should try ... which is 52_8. So, that's the same as before. Wait, but let me double-check. Maybe I can ... the product of 6_8 and 7_8 is 52_8. Wait, but let me double-check. Maybe I can think ... So, the product is 52_8. Alternatively, maybe I can think of ... Thus, the product of $6_8 \cdot 7_8$ in base 8 is $\boxed{52_8}$.