SETTLE: LEARNING WHEN TO STOP REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reasoning models often continue generating after their answers have settled. Settle learns when to stop from answer stability in completed traces. It trains the existing end-of-reasoning token while keeping other predictions close to the base model, and requires only ordinary decoding at inference. On MATH-500 with Qwen3-4B, Settle reduces token count by 40% with a 0.5-percentage-point decrease in accuracy. It gains 6.16 percentage points over supervised fine-tuning on the same traces shortened at their first stable answer, at nearly identical token counts. Its stopping score predicts whether a correct answer will remain correct. Settle extends the accuracy–token-count Pareto frontier of the evaluated stopping methods.

1 INTRODUCTION

Maximally performant models such as Qwen3, Nemotron, and DeepSeek-R1-Distill use intermediate reasoning to improve answer quality. This reasoning can also consume computation checking and restating an answer that no longer changes. An end-of-reasoning token marks the transition between these two stages. We call the recorded response, including its reasoning and final answer, a *trace*. Learning when to move from thinking to answering could save this computation while allowing difficult problems more reasoning.

A trace reveals earlier opportunities to make that transition. We take successively longer initial portions of the reasoning, called *prefixes*, and prompt the model to give a short answer from each one. Following Liu & Wang (2025), an answer is *stable* when it agrees with every later prompted answer, including the one prompted at the end of the recorded reasoning. A correct answer can still be unstable: in Figure 1, the answer reaches 107, changes to 106, and later returns to 107. These labels use the completed trace during training; the trained model decides from the reasoning generated so far.

Prior work reduces reasoning through supervised fine-tuning (SFT) on the shortest correct sampled solutions (Munkhbat et al., 2025) or a separate controller trained to interrupt a frozen model (Liu & Wang, 2025). Our stability-truncated SFT baseline trains on the same traces shortened at their first stable answer. Settle supervises the language model's own stopping decision, while regularizing its other predictions toward the base model.

We call this method Settle: it learns to stop when the model has settled on an answer, rather than at the first answer that happens to be correct. We train this distinction directly into the model's existing end-of-reasoning token, with stop targets at stable prefixes and continue targets at earlier ones. A Kullback–Leibler (KL) penalty keeps predictions at other positions close to the base model. The labels depend only on the model's own answers. Answer checks take place before training; at inference, the model makes the stopping decision through standard decoding.

On Qwen3-4B/MATH-500, Settle gains 6.16 natural-accuracy points over stability-truncated SFT at nearly identical average token counts. It also beats the regularized SFT baseline by 1.42 points while using 25.2% fewer tokens. Simply lowering the base model's token cap is less effective: Settle gains 7.03 forced-accuracy points over a 4,096-token cap while using 6.8% fewer tokens. These results distinguish learning when to stop from imitating shorter solutions or imposing a smaller budget. Settle extends the Pareto frontier among the evaluated stopping methods, with savings extending to Qwen3-8B and a separate MATH test sample. Its closing-token score also predicts whether a correct answer will persist at identical prefix lengths, where the base score is near chance. All stopping decisions use ordinary decoding.

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.

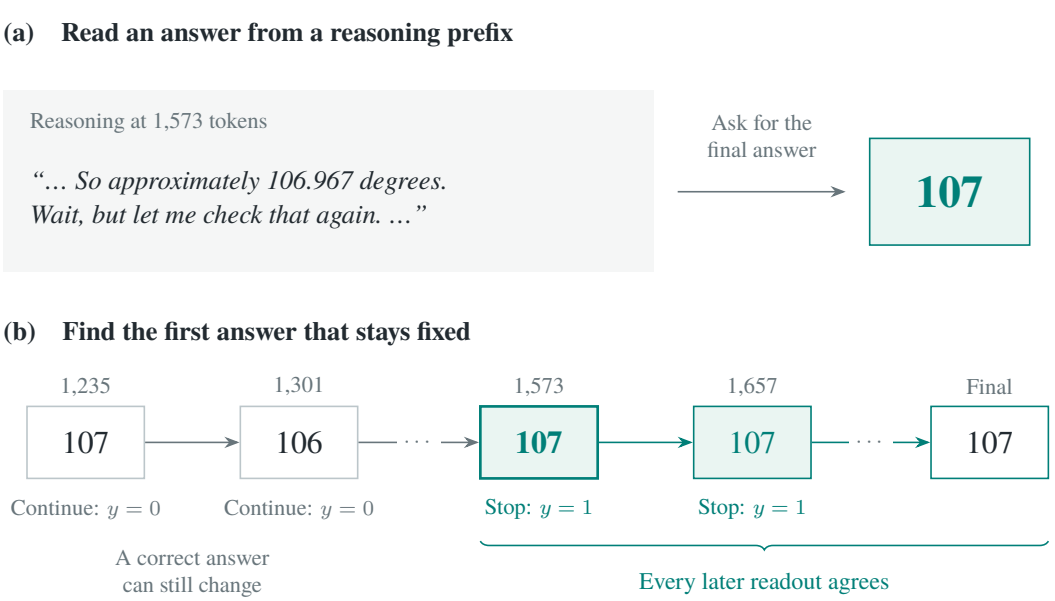

Figure 1: **From answer readouts to stopping targets.** A recorded Qwen3-4B trace solves a pie-chart problem whose answer is $107°$. (a) Prompting a saved prefix gives a short answer without changing the recorded continuation. (b) A first-correct-answer rule would stop at 1,235 tokens, but that answer changes later. Settle instead labels the stable suffix: from 1,573 tokens onward, every sampled answer agrees with the final readout. These boundaries receive stop targets; earlier boundaries receive continue targets. Ellipses mark omitted readouts; the final readout defines agreement and is not a training target.

2 Learning to stop from answer stability

Settle first labels the stable suffix of each recorded trace, then trains the model to recognize those stopping opportunities from its prefixes. Figure 1 shows the label construction on a real example.

Let x be a sampled response. A prefix $x_{<t}$ consists of its first t generated tokens, conditioned throughout on the original question. We choose candidate boundaries $t_1 < \cdots < t_K$ at paragraph breaks and at sentence endings before reflection phrases such as *Wait* or *Alternatively*. Appendix D specifies the spacing rule and the limit of 40 sampled boundaries per trace. The last prefix ends at T, immediately before the model's first end-of-reasoning tag (`</think>`), or at the end of the available response if no such tag occurs.

From a saved copy of each sampled prefix and of T, we append a closing tag and a final-answer prompt and greedily decode up to 16 tokens, following budget forcing (Muennighoff et al., 2025). These prompted answers are the *readouts* $a_1, \ldots, a_K, a_T$, with a_T the *terminal readout*. Each readout leaves the recorded continuation unchanged.

An answer is *stable at* t_j if every readout from t_j onward agrees with the terminal readout a_T. Let $E(a, b) \in \{0, 1\}$ indicate mathematical equivalence: normalized strings are compared first, followed by symbolic equivalence checking with Math-Verify (Kydlíček, 2025); a failed comparison returns zero. Following Liu & Wang (2025), the stop target is

$$y_j = \prod_{k=j}^{K} E(a_k, a_T), \qquad j = 1, \ldots, K. \tag{1}$$

Thus $y_j = 1$ exactly when $E(a_k, a_T) = 1$ for every $k \geq j$. If any later readout differs, $y_j = 0$, even when $a_j = a_T$. The terminal readout serves as the comparison answer; only earlier boundaries receive training targets. Because the criterion compares the model's own answers, a consistently wrong answer also forms a stable suffix. Stopping is desirable here too: the remaining recorded

reasoning consumes tokens without correcting the prompted answer. Evaluation uses fresh responses generated by the trained model.

Algorithm 1 in Appendix D computes the labels with K terminal-answer comparisons and a backward scan. Answer readouts are generated once, offline, and the labelled traces are reused across training epochs. Deployment requires only the model's prediction from the current prefix.

For models with a single-token closing tag, define $p_\theta(t) = \pi_\theta(\texttt{</think>} \mid x_{<t})$. Let $\mathcal{B}(x)$ be the retained, non-terminal training boundaries and $\mathcal{R}(x)$ the response positions. We minimize

$$\begin{aligned}\mathcal{L}(\theta) = & -\sum_x \sum_{j:t_j \in \mathcal{B}(x)} [w y_j \log p_\theta(t_j) + (1-y_j)\log(1-p_\theta(t_j))] \\ & + \lambda \sum_x \sum_{t \in \mathcal{R}(x)\setminus\mathcal{B}(x)} \mathrm{KL}(\pi_0(\cdot \mid x_{<t}) \,\|\, \pi_\theta(\cdot \mid x_{<t}))\,. \end{aligned} \tag{2}$$

The frozen reference π_0 is the base model. Increasing w raises the penalty for continuing after stability, while λ controls the KL penalty on other predictions. Both losses are summed over their positions. A trace therefore contributes one binary target per retained boundary and one reference-distribution target at each remaining response position.

At inference, the closing token competes with the other tokens in the model's output distribution. Sampling it ends reasoning and begins the final answer. This uses the same decoding procedure as the base model. Appendix D gives the boundary-loss derivative.

Full-sequence SFT learns when to stop as part of predicting each token. For a target continuation token $z \neq \texttt{</think>}$, let $q_\theta(z \mid x_{<t})$ be its probability conditional on not stopping. Its SFT loss decomposes as

$$\ell_{\mathrm{SFT}}(z,t) = -\log \pi_\theta(z \mid x_{<t}) = -\log(1-p_\theta(t)) - \log q_\theta(z \mid x_{<t}). \tag{3}$$

The first term teaches the model to continue; the second teaches which continuation token to generate. At the closing tag, SFT contributes $-\log p_\theta(t)$ and teaches termination. SFT thus learns a stopping point together with a particular sequence of reasoning and answer tokens.

Settle assigns a binary decision target at each labelled boundary. Before stability, the target is to continue; throughout the stable suffix, the target is to stop. This supplies several acceptable stopping opportunities from one trace. At other positions, the KL term uses the base model's full next-token distribution as the target. Labelled boundaries are excluded from that term so their closing-token probabilities can respond directly to the stop/continue labels. The experiments below compare this objective with sequence imitation and vary reference regularization within both procedures.

3 Experimental design

We study Qwen3-4B, Qwen3-8B (Yang et al., 2025a), Llama-3.1-Nemotron-Nano-8B-v1 (NVIDIA, 2025), and DeepSeek-R1-Distill-Qwen-7B (DeepSeek-AI et al., 2025). Rollouts are sampled at temperature 1.0 and top-p $= 1.0$ from 50 DAPO-Math-17K problems (Yu et al., 2025) selected for intermediate difficulty. Each model is trained on its own readouts. Known correct answers are used to select training problems and measure development accuracy. Stopping targets are constructed by comparing the model's answers. Unless noted, Qwen3-4B results average three independently trained models; Appendix I identifies the runs used throughout the evaluations.

We fit rank-32 attention-projection LoRA adapters (Hu et al., 2021) for three epochs with AdamW at 3×10^{-4}, then merge them into the model weights. Qwen3-4B uses 1,000 rollouts and 40,000 nonterminal readouts; one training run takes approximately 56 minutes on an H100. The main models use reference weight $\lambda = 0.2$, with stopping weights $w \in \{0.10, 0.25\}$. Appendix D gives the optimizer, label counts, and training context.

MATH-500 is the main benchmark for both Qwen3 sizes, using the same grading and overlap exclusions. We also evaluate the same Qwen3-4B models and stopping weight on GSM8K, OlympiadBench, GPQA-Diamond, AMC23, and Minerva. Appendix G gives sample counts and scoring details.

Evaluation uses thinking-mode chat templates, temperature 0.6, top-$p = 0.95$, and caps of 8,192 or 16,384 tokens. MATH-500 (Hendrycks et al., 2021) accuracy averages four samples on each of 498 problems, excluding two training overlaps; token means include all 500.

We report two accuracy measures. *Natural accuracy* grades the final answer produced during ordinary generation, counting an unclosed reasoning block as wrong. *Forced accuracy* additionally prompts responses that reach the token cap to give a final answer; responses that finish naturally are unchanged. Unless otherwise specified, token counts include these additional answer tokens. DEER's count also includes its trial answers, their short prompts, and generated tokens omitted from the final response. Input processing is excluded.

We select $w = 0.10$ from $\{0.10, 0.25\}$ by average natural accuracy on a 144-problem development set. We use this weight for the other models as well. A separate 500-problem sample from the MATH test set provides an additional evaluation; it is disjoint from the training set, development set, and MATH-500 (Appendix F).

Main base runs use sampling seeds 42–44; trained runs pair training seeds 0–2 with sampling seeds 42–44. We average samples and the three runs within each problem, then bootstrap whole problems with 20,000 draws. The intervals describe variation across evaluation problems for these models. Per-seed rows appear in Appendix I. The supplementary suite uses three trained models and one base run.

4 Results

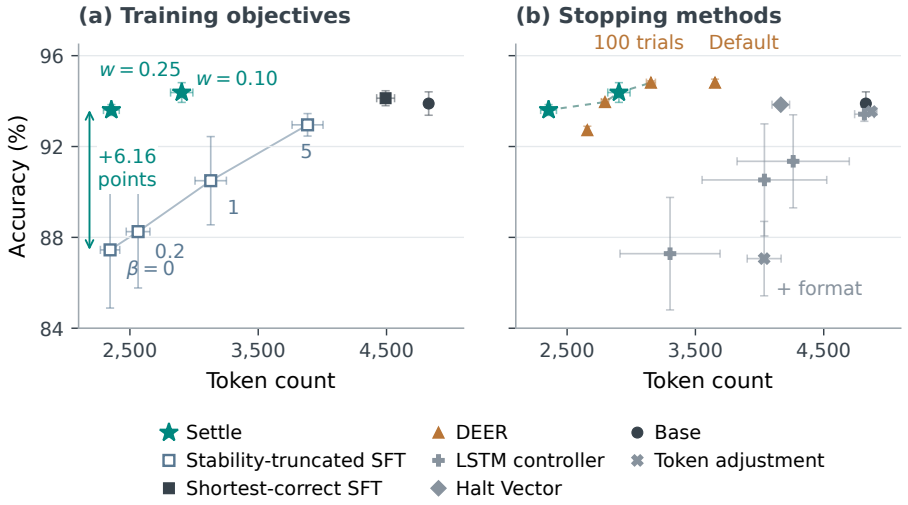

Figure 2: **Settle improves on imitation and extends the Pareto frontier.** Qwen3-4B, MATH-500, 16k; three-run means ± sample standard deviations. (a) β controls SFT regularization. (b) The dashed line joins nondominated means. Natural accuracy; token counts include DEER checks. Settle uses $\lambda = 0.2$. Baselines: shortest-correct SFT (Munkhbat et al., 2025), LSTM and token adjustment (Liu & Wang, 2025), Halt Vector (Jayabahu & Adeleke, 2026), and DEER (Yang et al., 2025b). Settings and scores appear in Appendix B.

4.1 Higher accuracy than imitating shorter solutions

Our first baseline, *stability-truncated SFT*, imitates Settle's source traces cut at their first stable answer. We append the answer readout and train every response token with cross-entropy, using all eligible traces, including incorrect ones. We compare this baseline with and without reference regularization; Settle uses $\lambda = 0.2$ unless varied explicitly. SFT training settings are selected on development data (Appendix A).

Table 1: **Reference regularization within each training procedure.** MATH-500, Qwen3-4B, 16k; three-run means ± sample standard deviations. Accuracy grades naturally generated final answers; token counts cover ordinary generation. SFT uses mean losses and Settle uses sums, so β and λ have different scales. Other settings stay fixed within each procedure (Appendix A).

Training procedure	Reference weight	Accuracy (%)	Token count
Stability-truncated SFT	$\beta = 0$	87.45 ± 2.56	$2,345 \pm 75$
Stability-truncated SFT	$\beta = 0.2$	88.25 ± 2.48	$2,564 \pm 92$
Stability-truncated SFT	$\beta = 1.0$	90.50 ± 1.94	$3,130 \pm 122$
Stability-truncated SFT	$\beta = 5.0$	92.96 ± 0.50	$3,884 \pm 121$
Settle, $w = 0.25$	$\lambda = 0$	0.03 ± 0.06	$6,289 \pm 6,642$
Settle, $w = 0.25$	$\lambda = 0.2$	93.61 ± 0.24	$2,355 \pm 62$

Table 2: **Sensitivity to Settle's reference weight.** MATH-500, Qwen3-4B, 16k; $w = 0.25$ and all other training settings fixed. Means ± sample standard deviations across three fits per coefficient. The sweep includes three new runs at the default $\lambda = 0.20$. Token counts cover ordinary generation.

λ	Natural accuracy (%)	Forced accuracy (%)	Token count
0.05	93.04 ± 0.15	93.07 ± 0.18	$2{,}057 \pm 23$
0.10	92.99 ± 0.23	93.01 ± 0.20	$2{,}167 \pm 28$
0.20	93.19 ± 0.33	93.31 ± 0.31	$2{,}338 \pm 45$
0.40	93.98 ± 0.48	94.14 ± 0.50	$2{,}695 \pm 73$

Both methods generate about 2,350 tokens per response, but Settle is substantially more accurate (Figure 2a). Settle at $w = 0.25$ reaches 93.61% natural accuracy, compared with 87.45% for stability-truncated SFT without regularization: a gain of 6.16 percentage points, with paired 95% interval $[4.70, 7.68]$. Their exact token counts are 2,355 and 2,345, a difference of just 0.4%. The gain remains at 6.17 points when unfinished responses are prompted to answer.

Shortest-correct SFT uses a different kind of demonstration: the shortest complete correct solution sampled for each problem (Munkhbat et al., 2025). It reaches 94.13% accuracy at 4,493 tokens. This baseline tests the benefit of selecting concise, correct solutions. Stability-truncated SFT tests the benefit of shortening Settle's source traces using the same answer-stability labels.

Adding a KL penalty strengthens stability-truncated SFT by keeping its predictions close to the base model (Table 1). At the largest tested penalty, $\beta = 5$, natural accuracy rises from 87.45% to 92.96%, while responses lengthen from 2,345 to 3,884 tokens.

Settle improves both accuracy and token count over this stronger SFT baseline. At $w = 0.10$, it reaches 94.38% accuracy with 25.2% fewer tokens, a paired gain of 1.42 points $[0.42, 2.49]$. The shorter Settle policy, $w = 0.25$, uses 39.4% fewer tokens and reaches 93.61% accuracy; its mean accuracy difference is $+0.65$ points $[-0.54, +1.84]$.

Removing Settle's KL penalty produces repeated closing tags or missing final answers: natural accuracy falls to 0.03%, and prompting unfinished responses to answer raises it only to 1.59%. The penalty preserves answer generation while the binary loss trains termination. Appendix A reports the full comparisons, including forced accuracy, paired intervals, and SFT restricted to the closing tag and final answer.

The savings persist across an eightfold range of positive KL weights (Table 2). Holding the stopping weight at $w = 0.25$, we train three models for each $\lambda \in \{0.05, 0.10, 0.20, 0.40\}$. All four settings use 44.2–57.4% fewer tokens than the base model, with natural accuracy between 92.99% and 93.98%.

Stronger regularization produces longer responses, with mean token count rising from 2,057 to 2,695 across this sweep. Relative to $\lambda = 0.20$, reducing the weight to 0.05 saves a further 12.0% of tokens $[10.50, 13.52]$, with an accuracy difference of -0.15 points $[-0.75, +0.45]$. Increasing it to 0.40 raises accuracy by 0.79 points $[0.08, 1.49]$ and token count by 15.3%. Thus the reference weight adjusts the accuracy–length tradeoff as well as preserving answer generation.

Table 3: **MATH-500 results with agreement labels,** $w = 0.10$**.** Forced accuracy in percent; changes in points with paired 95% intervals. Tokens are base / Settle means, including additional answer tokens. Each row averages three runs per model. "8k" and "16k" denote 8,192 and 16,384.

Model	Cap	Base	Settle	Change [95% interval]	Token count	Saved %
Qwen3-4B	8k	92.92	92.79	-0.13 [-0.95, +0.69]	4,172 / 2,650	36.5
Qwen3-4B	16k	95.23	94.73	-0.50 [-1.27, +0.27]	4,829 / 2,904	39.9
Qwen3-8B	16k	95.67	95.45	-0.22 [-0.94, +0.50]	5,112 / 3,135	38.7

Increasing the stopping weight w raises the penalty for continuing after the answer stabilizes and shortens responses. Raising it from 0.10 to 0.25 reduces mean token count from 2,904 to 2,356, including prompted answers for unfinished responses. Forced accuracy changes from 94.73% to 93.71%. Reference-answer targets give nearby results; they differ from agreement labels at only 689 of 40,000 sampled boundaries (Appendix A).

4.2 Extending the frontier with ordinary decoding

Figure 2b compares Settle with learned controllers, closing-token adjustments, and DEER's online answer checks. Both Settle settings are Pareto-optimal on Qwen3-4B/MATH-500: no compared method improves mean accuracy without using more tokens, or uses fewer tokens without lowering mean accuracy. We report natural final-answer accuracy averaged over three runs and count the tokens used by DEER's intermediate checks.

A long short-term memory (LSTM) controller trained on the same labels with the language model frozen (Liu & Wang, 2025) reaches 90.53% accuracy at 4,038 tokens. Settle at $w = 0.10$ gains 3.85 points $[2.76, 5.00]$ with 28.1% fewer tokens. Appendix B gives the development-selected threshold and full sweep.

Halt Vector moves a stopping intervention into the model weights (Jayabahu & Adeleke, 2026); our Qwen3-4B adaptation reaches 93.84% accuracy at 4,165 tokens. Think Token Adjustment changes the closing-token probability during inference (Liu & Wang, 2025). It reaches 93.54% at 4,872 tokens, or 87.06% at 4,036 with the released answer-format constraints. Settle has higher natural accuracy and uses fewer tokens than both.

PUMA selects exits using semantic redundancy and answer checks (Min et al., 2026). In a single-seed comparison on the same Qwen3-4B traces, PUMA-SFT reaches 93.07% at 2,803 tokens; Settle at $w = 0.25$ reaches 93.93% with 16.8% fewer tokens. Released PUMA replay reaches 93.42% at 3,433 tokens including checks (Appendix B.1).

DEER uses confidence in trial answers to decide when to exit (Yang et al., 2025b). These checks require extra generation even when reasoning continues; Settle generates no trial answers at inference. DEER's default and 100-trial settings both reach 94.81% accuracy, at 3,653 and 3,153 tokens. Tuning DEER on development data under the two Settle token budgets gives 93.96% at 2,795 tokens and 92.72% at 2,658 tokens (Appendix B).

Settle extends this frontier. At $w = 0.25$, it uses 11.4% fewer tokens than the 2,658-token DEER setting, with an accuracy difference of $+0.89$ points $[-0.03, +1.86]$. Against the 2,795-token setting, it saves 15.7% with a difference of -0.35 points $[-1.07, +0.38]$. Section 4.5 compares inference times.

4.3 Higher accuracy than a smaller token budget

Settle reduces token counts across Qwen3 sizes and inference budgets (Table 3). We compare these budgets using forced accuracy (Section 3). At a 16k cap, Qwen3-4B uses 39.9% fewer tokens, from 4,829 to 2,904, with an accuracy decrease of 0.50 points. At 8k, it saves 36.5% with a decrease of 0.13 points. Applying the same training rule and stopping weight to Qwen3-8B saves 38.7%, from 5,112 to 3,135 tokens, with a decrease of 0.22 points.

Learning when to stop outperforms simply lowering the base model's token cap. Across six tested caps (Figure 4, Appendix A), the 4,096-token cap gives 87.70% forced accuracy at 3,115 tokens

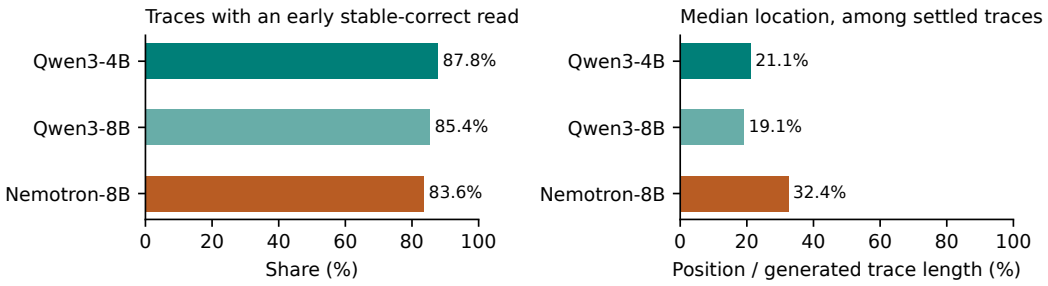

Figure 3: **Stable correct answers often precede completion.** MATH-500; 500 separately sampled traces per model, with an 8,192-token cap. Left: fraction with an early stable correct answer. Right: median first stable position as a fraction of trace length, among those traces. Appendix E gives full results and a repeated Qwen3-4B sample.

on average. Settle at $w = 0.10$ reaches 94.73% at 2,904 tokens: 7.03 points higher accuracy with 6.8% fewer tokens. It also has higher accuracy and lower mean token count than the 6,144- and 8,192-token-cap runs.

The difference is larger for shorter responses. A 2,560-token cap gives 80.77% accuracy at 2,314 tokens, compared with 93.71% at 2,356 for Settle ($w = 0.25$). The 16k base model retains the highest forced accuracy, but reducing its cap has a substantially steeper accuracy cost than learned stopping.

Earlier stopping can also improve the chance of producing a final answer within the budget. For Qwen3-4B at 16k, natural accuracy rises from 93.89% to 94.38%, while forced accuracy changes from 95.23% to 94.73%. Thus the natural-accuracy gain reflects improved completion within the budget; it accompanies a small decrease under forced scoring.

4.4 Answer stability and the learned stopping score

The base models often reach a stable correct answer well before completing their responses (Figure 3). On MATH-500, 87.8% of Qwen3-4B traces have an early correct answer that persists at every later readout. Among these traces, the first such readout occurs at a median of 21.1% of the generated length. Qwen3-8B and Nemotron show similar patterns. This diagnostic uses reference answers to assess correctness; the training labels require only agreement between readouts.

We next test whether Settle's stopping score predicts answer stability beyond the amount of reasoning already generated. We compare the base and trained models' closing-token probabilities on the same prefixes, alongside predictors that use only token count. The target is *stable correctness*: the current answer and every later readout must be correct. Reference answers define this diagnostic target; the training labels use only agreement between the model's own answers.

The score is evaluated before the next token is generated. Later readouts determine the evaluation target but are not supplied to the predictor. Scoring every model on the same recorded prefixes holds the reasoning text fixed while testing what the closing-token probability encodes. The generation experiments separately measure the accuracy and length of each model's own responses.

We compare prefixes at identical token counts, pairing a stably correct answer with one that is not. A second comparison considers only prefixes that already give the correct answer, testing whether the score predicts which answers will stay correct. We measure discrimination by the area under the receiver operating characteristic curve (AUROC); 0.5 indicates chance performance.

Across 17,899 boundaries from 498 MATH-500 traces, Settle reaches AUROCs of 0.831 and 0.838 for $w = 0.10$ and $w = 0.25$, respectively (Table 4). The base score reaches 0.532 and token-count predictors reach 0.618. Matching *identical* token counts leaves both Settle scores above 0.80, while the base score falls to 0.507. A token-count predictor necessarily ties, giving AUROC 0.5. Wider matching ranges give consistent results (Appendix C).

Table 4: **Predicting answer persistence beyond elapsed length and present correctness.** Overall and conditional AUROC [95% interval] on fixed Qwen3-4B prefixes. Overall uses 17,899 boundaries; matching token counts uses 12,106. The final column also requires the present readout to be correct, leaving 2,455 eligible boundaries. Intervals resample whole traces; each weight uses one agreement-trained model. † A token-count predictor assigns the same score to equal-length prefixes, giving conditional AUROC 0.5. Appendix C defines the estimands and reports all matching ranges.

Score	Overall	Same token count	Correct now, same token count
Base native score	0.532 [0.508, 0.556]	0.507 [0.484, 0.530]	0.495 [0.445, 0.547]
Elapsed tokens	0.618 [0.583, 0.655]	$0.500^{\dagger}$	$0.500^{\dagger}$
Time-only spline	0.618 [0.583, 0.654]	$0.500^{\dagger}$	$0.500^{\dagger}$
Settle, $w = 0.10$	0.831 [0.811, 0.851]	0.807 [0.782, 0.831]	0.695 [0.624, 0.764]
Settle, $w = 0.25$	0.838 [0.819, 0.857]	0.818 [0.793, 0.842]	0.706 [0.635, 0.774]

Among currently correct answers at identical token counts, Settle predicts which stay correct at every later readout, with AUROCs of 0.695 and 0.706. The base score is near chance at 0.495. Paired improvements are 0.200 $[0.112, 0.290]$ and 0.210 $[0.119, 0.305]$, showing that the score predicts persistence beyond present correctness and elapsed reasoning.

Settle predicts stability through the same closing-token score it uses to end generation.

4.5 Generalization and inference time

Settle saves 42.7% of tokens on a separate 500-problem MATH sample, with forced accuracy of 96.52% versus 96.33% for the base model. The same three Qwen3-4B models at $w = 0.10$ reduce mean token count from 3,803 to 2,179 at the 16k cap (Appendix F).

We keep the stopping weight fixed across five other benchmarks (Table 5). On GSM8K (Cobbe et al., 2021), Settle saves 43.2% of tokens with a 0.68-point forced-accuracy decrease. On OlympiadBench (He et al., 2024) and GPQA-Diamond (Rein et al., 2023), token counts fall by 36.3% and 13.2%, while accuracy rises by 3.76 and 0.34 points. On AMC23 and Minerva, token savings of 35.6% and 49.1% accompany forced-accuracy decreases of 2.6 and 3.7 points. The fixed stopping weight therefore transfers token savings more consistently than accuracy (Appendix G).

Table 5: **Token savings transfer across tasks.** Qwen3-4B, $w = 0.10$, 16k cap. Forced accuracy and token pairs are base / Settle. All rows average three trained models; MATH uses three base runs and the other tasks one. Sample sizes and protocols appear in Appendices F and G.

Benchmark	Accuracy (%)	Token count	Saved (%)
MATH, separate sample	96.33 / 96.52	3,803 / 2,179	42.7
GSM8K	95.1 / 94.5	2,053 / 1,167	43.2
OlympiadBench	66.6 / 70.4	9,024 / 5,750	36.3
GPQA-Diamond	55.6 / 55.9	8,565 / 7,433	13.2
AMC23	93.4 / 90.8	7,617 / 4,903	35.6
Minerva	48.2 / 44.5	6,536 / 3,327	49.1

On 100 MATH problems using one H100, batch-1 median latency falls from 13.25 to 7.66 seconds. At batch size 32, total time is 302 seconds for Settle, 324 for base, and 521 for DEER, with natural accuracies of 94%, 94%, and 95%. The batch-32 saving over base is about 7% of total time, showing that fewer generated tokens need not yield a proportional runtime reduction. Appendix H reports the timing procedure and all batch sizes.

5 Related work

Liu & Wang (2025) introduce the future-agreement labels used here and train an LSTM stopping predictor on hidden activations of a frozen language model. They also study online answer consis-

tency and inference-time closing-token adjustment. Settle builds on their labels to train the existing closing-token distribution, which makes the language model itself the stopping policy. Online methods obtain evidence through additional computation: DEER uses intermediate answer confidence (Yang et al., 2025b), while SABER tests stability with branch probes (Cheng et al., 2026). LYNX learns confidence-controlled exits (Akgül et al., 2025), and TERMINATOR adds a transformer layer and prediction head trained on first-answer positions (Nagle et al., 2026). Settle learns from offline answer agreement and uses the existing output distribution, without an extra inference module.

Budget forcing controls reasoning length directly (Muennighoff et al., 2025). Training methods can instead change the generated policy: self-training imitates the shortest correct sampled solution (Munkhbat et al., 2025), and reinforcement-learning methods such as ThinkPrune optimize efficient reasoning (Hou et al., 2025). ReasonMaxxer applies sparse offline corrections with a reference anchor (Akgül et al., 2026), motivating targeted policy updates without an online RL loop. Several methods also learn stopping behavior in the weights. Halt Vector internalizes a causal steering intervention in an attention-only adapter (Jayabahu & Adeleke, 2026). ESTAR trains a separate stop-proposal token through SFT and reinforcement learning, with an external classifier accepting proposals during inference (Wang et al., 2026). PUMA learns exit positions selected using semantic redundancy and answer verification through SFT, DPO, or GRPO (Min et al., 2026). Settle combines binary supervision of the existing stop token at multiple agreement-labelled boundaries with reference regularization elsewhere. The SFT controls use the same traces, with and without a reference penalty, to compare this decision loss with sequence imitation.

6 Limitations and future work

Our emphasis on mathematical reasoning follows prior work on concise generation (Munkhbat et al., 2025); additional model and benchmark evaluations test transfer. The calibration study motivates testing development-selected stopping weights across more tasks. Denser boundary sampling could improve stopping performance by locating the onset of stability more precisely. Longer rollouts and alternative readouts would extend label evidence; fresh on-policy traces could better match deployment prefixes.

Our regularization experiments hold other settings fixed. Jointly varying stopping pressure, regularization, supervision coverage, and training duration would map further accuracy–length tradeoffs. Interventions on the learned stop score could test its causal contribution. Further training runs and serving studies across hardware and batch sizes would clarify reproducibility and practical efficiency.

7 Conclusion

Settle learns when to stop from completed reasoning traces and acts through ordinary decoding. On Qwen3-4B/MATH-500, it saves 39.9% of tokens versus the base model, with a 0.50-point decrease in forced accuracy. It gains 6.16 natural-accuracy points over stability-truncated SFT at nearly identical token counts, and beats regularized SFT by 1.42 points with 25.2% fewer tokens. Against a 4,096-token cap, it gains 7.03 forced-accuracy points while using 6.8% fewer tokens. These comparisons favour learning stopping over sequence imitation and fixed budgets. Savings extend to Qwen3-8B and a separate MATH sample, and persist across an eightfold range of KL weights. The learned score predicts whether correct answers will persist even at identical token counts. The stopping policy is encoded in the merged model weights. Deployment uses the base model's vocabulary and decoding procedure, with no extra stopping head or online answer checks. Together, these findings support answer stability as supervision for efficient reasoning through the model's existing stopping action, with an accuracy–length tradeoff that depends on the model and task.

AI use statement

The authors declare that no generative AI or AI-assisted technologies were used in the preparation, writing, or data analysis of this manuscript.

Reproducibility statement

Appendix D specifies boundary construction, training, model selection, scoring, and bootstrap estimation. Appendix F documents the separate MATH evaluation sample, and Appendix I reports individual runs. Appendix C describes the native-score diagnostics. The remaining appendices give the full benchmark results, training and truncation controls, baseline comparisons, and serving measurements, including hardware and batch settings.

References

Ömer Faruk Akgül, Yusuf Hakan Kalaycı, Rajgopal Kannan, Willie Neiswanger, and Viktor Prasanna. LYNX: Learning Dynamic Exits for Confidence-Controlled Reasoning, December 2025. URL http://arxiv.org/abs/2512.05325. arXiv:2512.05325 [cs.CL].

Ömer Faruk Akgül, Rajgopal Kannan, Willie Neiswanger, and Viktor Prasanna. Rethinking RL for LLM Reasoning: It's Sparse Policy Selection, Not Capability Learning, May 2026. URL http://arxiv.org/abs/2605.06241. arXiv:2605.06241 [cs.CL].

Wanli Cheng, Haiya Xiang, Juntao Li, Hongling Wang, and Wenliang Chen. SABER: Stability-Aware Early Exit for LLM Reasoning via Adversarial Branch Probing, August 2026. URL http://arxiv.org/abs/2608.27963. arXiv:2608.27963 [cs.AI].

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. Training Verifiers to Solve Math Word Problems, November 2021. URL http://arxiv.org/abs/2110.14168. arXiv:2110.14168 [cs.LG].

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *Nature*, 645(8081): 633–638, September 2025. ISSN 0028-0836, 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://arxiv.org/abs/2501.12948. arXiv:2501.12948 [cs.CL].

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiad-

Bench: A Challenging Benchmark for Promoting AGI with Olympiad-Level Bilingual Multimodal Scientific Problems, June 2024. URL http://arxiv.org/abs/2402.14008. arXiv:2402.14008 [cs.CL].

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, November 2021. URL http://arxiv.org/abs/2103.03874. arXiv:2103.03874 [cs.LG].

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, April 2025. URL http://arxiv.org/abs/2504.01296. arXiv:2504.01296 [cs.CL].

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models, October 2021. URL http://arxiv.org/abs/2106.09685. arXiv:2106.09685.

Dylan Jayabahu and Tinuade Adeleke. The Halt Vector: Internalizing a Causal Steering Intervention for Efficient Reasoning, August 2026. URL http://arxiv.org/abs/2608.28859. arXiv:2608.28859 [cs.LG].

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient Memory Management for Large Language Model Serving with PagedAttention, September 2023. URL http://arxiv.org/abs/2309.06180. arXiv:2309.06180 [cs.LG].

Hynek Kydlíček. Math-Verify: Math Verification Library, 2025. URL https://github.com/huggingface/Math-Verify. Published: Software.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving Quantitative Reasoning Problems with Language Models, July 2022. URL http://arxiv.org/abs/2206.14858. arXiv:2206.14858 [cs.CL].

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. 17896–17907, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.904. URL https://aclanthology.org/2025.emnlp-main.904.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization, January 2019. URL http://arxiv.org/abs/1711.05101. arXiv:1711.05101 [cs.LG].

Mathematical Association of America. American Mathematics Competitions. URL https://maa.org/student-programs/amc/. Published: Official competition website.

Dehai Min, Giovanni Vaccarino, Huiyi Chen, Yongliang Wu, Gal Yona, and Lu Cheng. Stop When Reasoning Converges: Semantic-Preserving Early Exit for Reasoning Models, May 2026. URL http://arxiv.org/abs/2605.17672. arXiv:2605.17672 [cs.CL].

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Fei-Fei Li, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, March 2025. URL http://arxiv.org/abs/2501.19393. arXiv:2501.19393 [cs.CL].

Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. Self-Training Elicits Concise Reasoning in Large Language Models, June 2025. URL http://arxiv.org/abs/2502.20122. arXiv:2502.20122 [cs.CL].

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, May 2026. URL http://arxiv.org/abs/2603.12529. arXiv:2603.12529 [cs.LG].

NVIDIA. Llama-3.1-Nemotron-Nano-8B-v1 model card, 2025. URL https://huggingface.co/nvidia/Llama-3.1-Nemotron-Nano-8B-v1. Published: Hugging Face model repository.

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, November 2023. URL http://arxiv.org/abs/2311.12022. arXiv:2311.12022 [cs.AI].

Junda Wang, Zhichao Yang, Dongxu Zhang, Sanjit Singh Batra, and Robert E. Tillman. ESTAR: Early-Stopping Token-Aware Reasoning For Efficient Inference, February 2026. URL http://arxiv.org/abs/2602.10004. arXiv:2602.10004 [cs.AI].

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 Technical Report, May 2025a. URL http://arxiv.org/abs/2505.09388. arXiv:2505.09388 [cs.CL].

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. Dynamic Early Exit in Reasoning Models, September 2025b. URL http://arxiv.org/abs/2504.15895. arXiv:2504.15895 [cs.CL].

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO: An Open-Source LLM Reinforcement Learning System at Scale, May 2025. URL http://arxiv.org/abs/2503.14476. arXiv:2503.14476 [cs.LG].

A Stopping supervision and reference regularization

The first stable answer can supply either a shorter demonstration or a target for the stopping decision. Our central comparison uses the same traces to test these two learning procedures. Settle improves accuracy over stability-truncated SFT at nearly identical token counts; the additional controls examine how reference regularization and the choice of stopping targets contribute to this result.

For stability-truncated SFT, we cut each source response at its first future-agreement boundary and append the closing tag and answer readout. Cross-entropy supervises the entire target, with question tokens masked, no reference anchor, and no correctness filter. Of 1,000 responses, 993 have an eligible boundary and 976 fit the 8,192-token context. Adapters match Settle's rank and scaling. We use AdamW with zero weight decay, accumulation over eight responses, gradient clipping at 1, and a linear schedule with 10% warmup. Three seeds are trained at each learning rate, 10^{-4} and 3×10^{-4}; all three epochs are evaluated on 144 development problems using training seed 0.

To compare shorter policies, we select the highest development accuracy within 50%, 60%, or 75% of the base model's token count, breaking ties by lower count. The 50% budget selects epoch 1 at 3×10^{-4}; both larger budgets select epoch 2 at the same rate, which also has the highest accuracy overall. These choices are fixed for all three seeds before MATH-500 evaluation. Table 6 gives both durations, and epoch 2 enters the main comparison.

Table 6: **SFT on stability-truncated solutions.** MATH-500 at 16k, four responses per problem. Accuracy excludes the two overlapping problems; token means include all 500. All rows average three independently trained models. Epochs are selected on development data.

Training procedure	Natural %	Forced %	Token count
SFT: stability-truncated, epoch 1	80.30	80.44	2,250
SFT: stability-truncated, epoch 2	87.45	87.53	2,345
Settle, $w = 0.10$	94.38	94.73	2,904
Settle, $w = 0.25$	93.61	93.71	2,355

We vary the reference constraint within each learning procedure. Settle compares $\lambda = 0$ with $\lambda = 0.2$ at $w = 0.25$, keeping all 1,000 source traces, boundary labels, supervised positions, and 375 updates fixed. Stability-truncated SFT uses the same 976 targets, with reference KL added to the retained reasoning prefix. Let $\mathcal{C}$ denote the shortened completion, including its closing tag and answer, and $\mathcal{P} \subset \mathcal{C}$ its reasoning prefix. The per-response loss is

$$\mathcal{L}_{\text{SFT+KL}} = -\frac{1}{|\mathcal{C}|} \sum_{t \in \mathcal{C}} \log \pi_\theta(x_t \mid x_{<t}) + \frac{\beta}{|\mathcal{P}|} \sum_{t \in \mathcal{P}} \text{KL}(\pi_0(\cdot \mid x_{<t}) \,\|\, \pi_\theta(\cdot \mid x_{<t})) \,.$$

The KL term acts on the reasoning prefix, excluding the question and appended ending. We test $\beta \in \{0, 0.2, 1.0, 5.0\}$ with three seeds at the selected learning rate 3×10^{-4}. Training ends at the epoch-2 checkpoint, after 244 updates of the original three-epoch, 366-update schedule. The remaining optimizer and evaluation settings stay fixed. Because SFT averages each loss over positions whereas Settle sums them, the coefficients have different scales.

The two procedures use the same source traces and agreement labels; their losses, supervised positions, and training durations are specified above. Appendix B reports all scores.

The positive-weight sweep in Table 2 uses twelve additional fits: three training seeds at each $\lambda \in \{0.05, 0.10, 0.20, 0.40\}$, with $w = 0.25$ throughout. All settings use the same 1,000 labelled traces, 8,192-token training context, 375 updates, adapters, optimizer, and evaluation protocol. The sweep includes three new training runs at the default $\lambda = 0.20$. Figure 2 and Table 1 retain the original models. Each fit and evaluation uses one H200. Table 31 gives the individual runs.

Without KL regularization, 5,995 of 6,000 responses contain no extracted final answer. Table 30 reports the individual runs.

Table 8 compares the supervision targets and stopping rules. Reference-answer and agreement supervision use the main source traces. Ending-only SFT, first-positive-only training, and shuffled labels use a separate trace sample; the latter two also use a higher stopping weight.

Table 7: **Both Settle settings versus every SFT reference weight.** Settle-minus-SFT accuracy differences in percentage points, with paired 95% problem-bootstrap intervals. Token savings compare ordinary-generation means; negative values mean Settle uses more tokens. All comparisons use three trained models per setting and the interval procedure in Appendix D.

SFT β	Natural accuracy change	Forced accuracy change	Tokens saved (%)
Settle $w = 0.10$			
0	$+6.93\,[+5.52, +8.43]$	$+7.20\,[+5.79, +8.68]$	-23.81
0.2	$+6.12\,[+4.77, +7.56]$	$+6.21\,[+4.84, +7.65]$	-13.26
1.0	$+3.88\,[+2.71, +5.10]$	$+3.77\,[+2.58, +5.00]$	$+7.22$
5.0	$+1.42\,[+0.42, +2.49]$	$+0.87\,[-0.13, +1.92]$	$+25.25$
Settle $w = 0.25$			
0	$+6.16\,[+4.70, +7.68]$	$+6.17\,[+4.70, +7.70]$	-0.43
0.2	$+5.35\,[+3.93, +6.81]$	$+5.19\,[+3.78, +6.64]$	$+8.12$
1.0	$+3.11\,[+1.81, +4.42]$	$+2.74\,[+1.44, +4.05]$	$+24.74$
5.0	$+0.65\,[-0.54, +1.84]$	$-0.15\,[-1.32, +1.00]$	$+39.36$

Table 8: Training controls on MATH-500 at 16k. Accuracy uses the same scoring and overlap exclusions as the main evaluation. Tokens include additional answer tokens under forced scoring. Reference labels use known correct answers; agreement labels use the model's own answers.

Supervision	Runs	Natural %	Forced %	Token count
Agreement, $w = 0.10$	3	94.38	94.73	2,904
Reference, $w = 0.10$	3	93.96	94.38	2,961
Reference, $w = 0.25$	3	93.56	93.69	2,408
Agreement, $w = 0.25$	3	93.61	93.71	2,356
Base cut at 2,560	1	35.89	80.77	2,314
Shuffled labels	1	86.75	86.80	1,296
Shortest-correct SFT	3	94.13	95.30	4,493

A uniform token cap shortens reasoning without changing the model. Figure 4 compares this intervention with learned stopping; Table 9 gives the six caps, using Qwen3-4B, seed 42, four responses per problem, and the common protocol. The additional 2,560-token cap gives 80.77% forced accuracy at 2,314 mean tokens (Table 8).

Table 9: Qwen3-4B fixed-budget curve on MATH-500. Each row is one evaluation run.

Maximum tokens	Forced accuracy %	Mean token count
2,048	78.01	1,950
3,072	83.84	2,624
4,096	87.70	3,115
6,144	91.62	3,781
8,192	93.47	4,198
16,384	95.73	4,869

Following Munkhbat et al. (2025), shortest-correct SFT chooses the shortest complete correct response per training problem that fits the context. It imitates this response through its natural ending. This retains a complete solution, whereas stability-truncated SFT constructs a new ending at a labelled boundary.

Ending-only SFT retains each rollout through its first stable-correct nonterminal boundary; rollouts without one are excluded. Negative log-likelihood supervises the appended closing tag, final-answer prompt, forced answer, and end-of-response token, while reference KL with coefficient 0.2 constrains the retained prefix. Training uses the separate Qwen3-4B trace sample, an 8,192-token context, and seeds 0 and 1.

Table 10 uses the common evaluation protocol with seed 42 and four responses per problem: both models at 8k and training seed 0 at 16k.

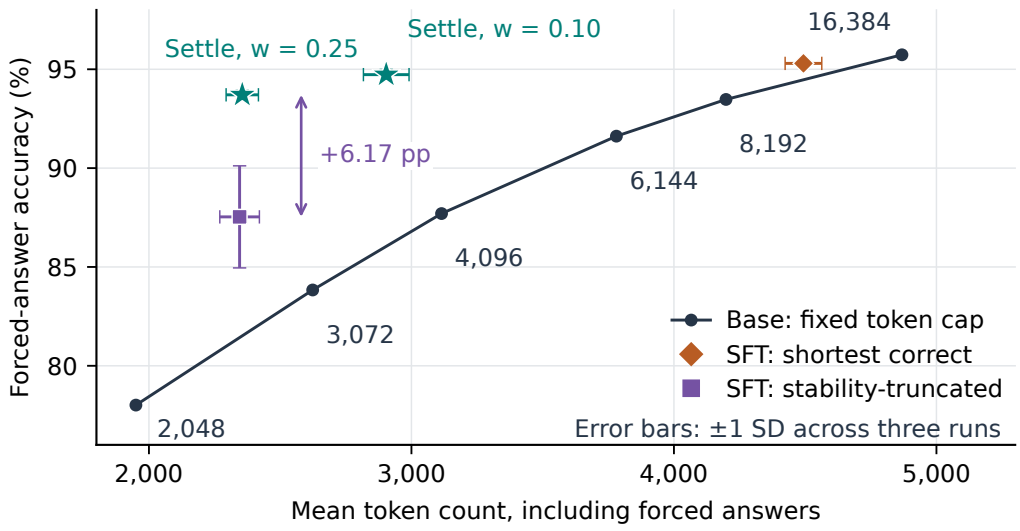

Figure 4: **Learning the stop action versus shorter budgets and demonstrations.** Labels on the base curve give maximum token budgets. Every point uses forced-answer accuracy and generated-token count, with the same scoring and overlap exclusions. Each base cap uses one run; Settle and both SFT controls show three-run means with sample-standard-deviation bars at a 16k cap. SFT on stability-truncated solutions uses the development-selected setting; the arrow gives its forced-accuracy difference from Settle at $w = 0.25$. The base curve uses a separate evaluation from Table 3; lines connect its evaluated caps.

Table 10: Ending-only SFT on MATH-500. Accuracy is natural, with no forced answer at the cap.

Maximum tokens	Training seed	Natural accuracy %	Token count
8,192	0	86.45	1,767
8,192	1	86.60	1,684
16,384	0	88.15	1,923

We isolate the placement of positive stopping supervision by concentrating each trace's total positive loss weight at its first stable boundary. Later positive boundaries receive zero stopping weight, while the negative targets and all positions receiving KL regularization stay unchanged. Thus both methods have the same positive supervision weight per trace. Both use the original 1,000 traces, training seed 0, $w = 0.10$, $\lambda = 0.2$, and 375 updates.

Table 11: Placement of positive stopping supervision, Qwen3-4B. Positive loss weight per trace, negative targets, and KL positions are identical. One training seed; four responses per MATH-500 problem, with evaluation seed 42 and a 16k cap. Token counts include prompted answers for capped responses.

Task	Positive supervision	Natural %	Forced %	Token count
MATH-500	First stable boundary	93.42	93.93	2,969
	All stable boundaries	94.18	94.58	2,890

On MATH-500, supervising all stable boundaries raises natural accuracy by 0.75 points, with paired 95% problem-bootstrap interval $[-0.15, +1.71]$, and uses 2.6% fewer tokens. This comparison holds supervision weight and regularization positions fixed.

A correct readout need not remain correct as reasoning continues. We test whether that future information improves stopping targets, using the same 1,000 traces, 39,649 positions, adapters, KL

positions, and optimizer. Persistent labels require correctness at the current and all later readouts; current-only labels use the present readout. The current-only comparison uses both the same stopping weight and reweighted positive and negative losses that preserve their total contribution.

At 16k, natural accuracy is 94.03% for persistent labels, 93.12% for current-only labels, and 93.32% after reweighting. Mean token counts are 2,406, 2,312, and 2,322; forced accuracy is 94.18%, 93.26%, and 93.46%, respectively. Persistent labels perform best at 16k. At 8k the ranking varies by condition.

We also test a fixed stopping threshold applied to the learned closing probability. On the separate MATH sample, a threshold of 0.5 applied to the reference-answer-trained model at $w = 0.25$ gives 94.67% forced accuracy, versus 96.33% for base (paired difference -1.67 points, 95% interval $[-2.90, -0.45]$).

B Comparisons with alternative stopping policies

Both Settle settings form part of the accuracy–token-count Pareto frontier in Figure 2. Table 12 gives the complete comparison under the common protocol, including the tokens used by DEER's intermediate checks (Appendix D).

Following Liu & Wang (2025), a one-layer LSTM with 128 hidden units predicts agreement from frozen final-layer activations at Settle's labelled prefixes. Validation F1, with problems separated between fitting and validation, selects training duration before refitting on the full pool with three seeds. The controller reads activations every 128 generated tokens in place of the original sentence schedule. At the first threshold crossing, it closes reasoning and generates a greedy answer of at most 256 tokens within the remaining budget. We evaluate by replaying traces, exposing each decision only to the available prefix activations. Token count includes the retained prefix and answer; the inserted answer prompt is counted separately.

We evaluate controller thresholds $0.5, 0.9, 0.95, 0.99, 0.995, 0.999, 0.9995, 0.9999$ on 144 development problems with four responses each. The highest natural accuracy within 50%, 60%, and 75% of base token count selects $\tau = 0.95, 0.99, 0.995$, respectively, with ties broken by lower count. We also test $\tau = 0.9999$, which has the highest development accuracy overall. These thresholds are fixed for all three trained controllers before MATH-500 evaluation; the main comparison uses the 60% budget.

A separate three-run evaluation checks variation in the base-model reference. With the same model, prompt, sampling, and scoring, it gives 93.44% natural accuracy and 94.95% forced accuracy at 4,877 tokens. The main base runs have $+0.45$ natural-accuracy points relative to these runs (paired 95% interval $[-0.18, +1.10]$) and 1.0% lower token count.

Table 13 varies ten settings of the released DEER procedure (Yang et al., 2025b), using sampling seed 42 and greedy trial answers capped at 20 tokens. The default combines confidence threshold 0.95, the 10-trial setting, and an 80% reasoning allowance. Variants change the threshold, use 100 trials, or reduce the allowance to 50%. The trial parameter limits checks at continuation boundaries; reaching a segment limit or ending without a closing tag can trigger additional checks. All checks count toward token use. The replicated default and 100-trial results use seeds 42–44 with the same model, prompt, policy, and accounting.

The lowest-count setting in this sweep reaches 94.33% accuracy at 2,784 tokens, including its intermediate checks.

We select DEER settings under Settle's development token budgets. We cross $\theta \in \{0.50, 0.95\}$ with $\rho \in \{0.50, 0.25, 0.125\}$ at 100 trials, allowing $\lfloor 16{,}384\rho \rfloor$ reasoning tokens and holding other settings fixed. Selection uses 144 development problems, four responses each and seed 42, and maximizes natural accuracy within Settle's mean counts: 5,470.28 for $w = 0.25$ (two development runs) and 6,509.89 for $w = 0.10$ (three runs). Counts include trial answers, added prompts, and discarded tokens. Ties favor lower count, then a fixed configuration order. The selected (θ, ρ) settings are $(0.50, 0.50)$ for $w = 0.10$ and $(0.95, 0.25)$ for $w = 0.25$. Table 15b gives all six candidates. Both selected policies are evaluated on MATH-500 with seeds 42–44.

Table 12: **Learning when to stop: training controls and existing stopping methods.** MATH-500, Qwen3-4B, 16k cap; three-run means ± sample standard deviations. Natural accuracy grades generated final answers; forced accuracy also prompts budget-limited responses to answer. Token counts cover ordinary generation before additional answer forcing, including intermediate checks for DEER. Bold marks the best mean within each panel. Unregularized SFT and controller settings are selected on development data; regularization controls keep other training settings fixed (Appendix A).

Method	Natural (%) ↑	Forced (%) ↑	Token count ↓
(a) Training objectives and reference anchoring			
Base (Yang et al., 2025a)	93.89 ± 0.52	95.23 ± 0.23	$4,828 \pm 28$
SFT: shortest correct (Munkhbat et al., 2025)	94.13 ± 0.33	$\mathbf{95.30} \pm 0.14$	$4,493 \pm 70$
SFT: stability-truncated	87.45 ± 2.56	87.53 ± 2.58	$\mathbf{2,345} \pm 75^{\dagger}$
SFT: stability-truncated, $\beta = 0.2$	88.25 ± 2.48	88.52 ± 2.49	$2,564 \pm 92$
SFT: stability-truncated, $\beta = 1.0$	90.50 ± 1.94	90.96 ± 1.83	$3,130 \pm 122$
SFT: stability-truncated, $\beta = 5.0$	92.96 ± 0.50	93.86 ± 0.63	$3,884 \pm 121$
Settle, $w = 0.25$, $\lambda = 0$	0.03 ± 0.06	1.59 ± 1.08	$6,289 \pm 6,642$
Settle, $w = 0.10$	$\mathbf{94.38} \pm 0.43$	94.73 ± 0.26	$2,904 \pm 87$
Settle, $w = 0.25$	93.61 ± 0.24	93.71 ± 0.24	$2,355 \pm 62^{\dagger}$
† Token counts differ by 0.4%; Settle gains 6.16 accuracy points over stability-truncated SFT.			
(b) Alternative stopping policies			
LSTM controller (Liu & Wang, 2025)	90.53 ± 2.47	91.23 ± 3.02	$4,038 \pm 486$
Halt Vector (Jayabahu & Adeleke, 2026)	93.84 ± 0.08	94.61 ± 0.08	$4,165 \pm 68$
Token adjustment (Liu & Wang, 2025)	93.54 ± 0.18	$\mathbf{94.93} \pm 0.18$	$4,872 \pm 42$
+ answer formatting	87.06 ± 1.64	87.88 ± 1.88	$4,036 \pm 132$
DEER, default (Yang et al., 2025b)	$\mathbf{94.81} \pm 0.15$	94.81 ± 0.15	$3,653 \pm 10$
DEER, 100 trials (Yang et al., 2025b)	$\mathbf{94.81} \pm 0.08$	94.81 ± 0.08	$3,153 \pm 35$
DEER, $\theta = 0.5$, $\rho = 0.5$ (Yang et al., 2025b)	93.96 ± 0.12	94.01 ± 0.08	$2,795 \pm 29$
DEER, $\theta = 0.95$, $\rho = 0.25$ (Yang et al., 2025b)	92.72 ± 0.17	92.75 ± 0.21	$2,658 \pm 15$
Settle, $w = 0.10$	94.38 ± 0.43	94.73 ± 0.26	$2,904 \pm 87$
Settle, $w = 0.25$	93.61 ± 0.24	93.71 ± 0.24	$\mathbf{2,355} \pm 62$
(c) Additional base and controller settings			
Base (repeat evaluation)	$\mathbf{93.44} \pm 0.29$	$\mathbf{94.95} \pm 0.25$	$4,877 \pm 26$
LSTM controller, $\tau = 0.95$	87.28 ± 2.48	87.68 ± 2.83	$\mathbf{3,302} \pm 390$
LSTM controller, $\tau = 0.995$	91.35 ± 2.05	92.17 ± 2.62	$4,262 \pm 437$
LSTM controller, $\tau = 0.9999$	93.42 ± 0.31	94.78 ± 0.45	$4,817 \pm 76$

Table 13: DEER settings on Qwen3-4B at 16k. θ is the confidence threshold, ρ the reasoning-budget fraction, and trials the checking parameter defined above. Accuracy is natural; token counts include checks. Extra counts trial answers, added trial prompts, and discarded generated tokens on MATH-500. Each row reports one run; replicated means appear in Table 12. MATH-500 uses the main overlap exclusions.

θ	Trials	ρ	MATH %	Token count	Extra
0.95	10	0.80	94.83	3,688	118
0.90	10	0.80	94.78	3,530	115
0.85	10	0.80	94.83	3,561	115
0.80	10	0.80	94.88	3,475	115
0.70	10	0.80	94.63	3,480	114
0.50	10	0.80	94.93	3,470	114
0.95	100	0.80	94.78	3,157	229
0.95	10	0.50	94.73	3,294	118
0.50	10	0.50	94.33	3,152	114
0.50	100	0.50	94.33	2,784	199

Table 14 reports all four paired comparisons between the selected DEER settings and the two Settle weights.

Table 14: **Settle versus development-selected DEER on MATH-500.** Accuracy differences are Settle minus DEER in percentage points, with paired 95% intervals computed as in Appendix D. Token savings compare ordinary means, including DEER checks; negative values mean Settle uses more tokens.

Settle weight	Natural difference	Forced difference	Tokens saved (%)
DEER $\theta = 0.50,\ \rho = 0.5$			
0.10	$+0.42\,[-0.18,\ +1.07]$	$+0.72\,[+0.08,\ +1.42]$	-3.88
0.25	$-0.35\,[-1.07,\ +0.38]$	$-0.30\,[-1.00,\ +0.42]$	$+15.73$
DEER $\theta = 0.95,\ \rho = 0.25$			
0.10	$+1.66\,[+0.82,\ +2.56]$	$+1.97\,[+1.09,\ +2.93]$	-9.26
0.25	$+0.89\,[-0.03,\ +1.86]$	$+0.95\,[+0.03,\ +1.94]$	$+11.37$

We adapt Halt Vector, the activation-reconstruction procedure of Jayabahu & Adeleke (2026), to Qwen3-4B. Its compressibility filter retains 48 responses from 24 problems in the shared training pool. We fix steering strength 25 and layer 23, using a proportional-depth layer choice. Attention adapters on layers 0–23 have rank 16 and scaling 32. Training uses 12 epochs, learning rate 10^{-4}, accumulation over eight responses, reconstruction weight 1, and cosine decay with 10% warmup. All three fits use these responses and settings, with Settle's base model, training pool, and answer readouts.

Table 12 gives the Qwen3-4B results. On R1-Distill-Qwen-7B, the base model used in the Halt Vector publication, three adapters average 92.30% natural and 92.42% forced accuracy at 2,887 tokens. Settle gives 91.03% natural and 91.62% forced accuracy at 2,523 tokens under forced scoring; DEER gives 89.36% natural accuracy at 2,329 tokens including checks.

The closing token can also be encouraged without training. Think Token Adjustment (Liu & Wang, 2025) adds $\alpha(\max_v z_v - \text{mean}_v\, z_v)$ to its logit until the first closing tag. We evaluate $\alpha \in \{0, 0.2, 0.4, 0.6, 0.8, 1.0\}$, along with $\alpha = 0.6$ using the released answer-format constraints: a newline and boxed-answer prefix after closing. Both $\alpha = 0.6$ variants enter the MATH-500 comparison regardless of development rank. No setting meets the 50%, 60%, or 75% development token budget, and plain $\alpha = 0.6$ has the highest development natural accuracy, so the selection rule adds no further settings.

Against plain token adjustment, Settle saves 40.4% of tokens, with a natural-accuracy difference of $+0.84$ points (paired 95% interval $[-0.08, +1.82]$) and slightly lower forced accuracy (94.73% versus 94.93%). This comparison supports an efficiency advantage, with the accuracy ordering depending on the scoring convention. Against the answer-format variant, Settle saves 28.1% of tokens and gains 7.31 natural-accuracy points $[+5.87, +8.84]$. Tables 12 and 15 report both scoring conventions and every development setting.

Table 15: **Full development sweeps.** All settings use the same 144 problems, four responses each, and sampling seed 42. SFT and controller evaluations use training seed 0. SFT entries specify learning rate and epoch; every SFT entry uses solutions truncated at their first stable answer. Accuracy is in percent. Panel (a) counts naturally generated tokens; base is $\alpha = 0$. Panel (b) includes DEER checks; θ is the confidence threshold and ρ the reasoning-budget fraction. The final column marks selection under each Settle development ceiling. These problems exclude the training pool.

(a) Training and closing-token baselines

Method and setting	Natural %	Forced %	Token count
Base (repeat evaluation)	77.95	86.11	10,130
LSTM controller, $\tau = 0.5$	30.38	30.38	2,169
LSTM controller, $\tau = 0.9$	52.08	52.08	3,703
LSTM controller, $\tau = 0.95$	56.60	56.60	4,190
LSTM controller, $\tau = 0.99$	64.24	64.41	5,510
LSTM controller, $\tau = 0.995$	68.06	68.23	6,095
LSTM controller, $\tau = 0.999$	71.70	73.61	7,603
LSTM controller, $\tau = 0.9995$	73.44	76.39	8,152
LSTM controller, $\tau = 0.9999$	76.39	82.12	9,322
SFT: stability-truncated, 10^{-4}, epoch 1	26.22	26.56	1,752
SFT: stability-truncated, 10^{-4}, epoch 2	66.32	68.06	5,694
SFT: stability-truncated, 10^{-4}, epoch 3	66.15	67.53	5,229
SFT: stability-truncated, 3×10^{-4}, epoch 1	47.92	48.44	3,337
SFT: stability-truncated, 3×10^{-4}, epoch 2	72.92	74.13	5,580
SFT: stability-truncated, 3×10^{-4}, epoch 3	71.53	72.92	5,183
Think Token Adjustment, $\alpha = 0.2$	77.95	86.11	10,136
Think Token Adjustment, $\alpha = 0.4$	78.12	86.11	10,133
Think Token Adjustment, $\alpha = 0.6$	80.56	86.28	10,127
Think Token Adjustment, $\alpha = 0.6$ (answer format)	78.30	83.68	9,100
Think Token Adjustment, $\alpha = 0.8$	78.65	86.46	10,108
Think Token Adjustment, $\alpha = 1$	79.86	87.85	10,103

(b) DEER at lower reasoning budgets

θ	ρ	Natural (%)	Token count	Selected ceiling
0.50	0.5	78.99	6,485	$w = 0.10$
0.50	0.25	70.66	5,187	—
0.50	0.125	63.02	4,194	—
0.95	0.5	84.03	6,698	—
0.95	0.25	73.09	5,453	$w = 0.25$
0.95	0.125	62.33	4,283	—

B.1 PUMA WITH THE SAME BACKBONE AND SOURCE TRACES

We compare with PUMA's released semantic-redundancy detector and answer-checking procedure (Min et al., 2026), using Qwen3-4B, the shared question prompt, and a 16,384-token cap. The released stopping configuration uses similarity threshold 0.35, confidence threshold 0.98, confidence tolerance 0.03, two consecutive agreeing checks, and a minimum of ten reasoning steps. The implementation replays completed traces to select an exit, then generates the final answer. We count the retained prefix, the generated answer, and all generated tokens and added prompts from checks used before the exit, including checks before the minimum stopping step. Replay supports the accuracy and token-count comparison; the online timing study in Appendix H evaluates base, Settle, and DEER.

For PUMA-SFT, we apply the same exit-labeling procedure to Settle's 1,000 source traces. Following the published filtering rule, we keep correct regenerated answers whose retained reasoning is below 60% of the source length. This yields 293 training examples. We use the published SFT learning rate 2×10^{-4}, three epochs, accumulation over 16 examples, and rank-64 LoRA with scaling 128 on all linear layers. We evaluate the final epoch. This adapts PUMA-SFT to the common backbone and data; the original experiment uses 12,000 problems and R1-Distill-7B.

Table 16: PUMA comparisons with Qwen3-4B. Accuracy pairs are natural / forced percentages. Four responses per MATH-500 problem, evaluation seed 42, and one training seed per learned method. MATH accuracy excludes the two training overlaps; token means cover all problems and include checks and prompted answers for capped responses. Base uses PUMA's source generations.

	MATH-500	
Method	Accuracy	Tokens
Base	93.32 / 94.93	4,904
PUMA, released replay	93.42 / 93.67	3,433
PUMA-SFT, common data	93.07 / 93.17	2,803
Settle, $w = 0.10$	94.18 / 94.58	2,890
Settle, $w = 0.25$	93.93 / 93.98	2,331

On MATH-500, Settle at $w = 0.10$ gains 1.10 points in natural accuracy over PUMA-SFT, with paired 95% problem-bootstrap interval $[-0.05, +2.31]$, at 3.1% more tokens. Its forced-accuracy gain is 1.41 points $[0.25, 2.61]$. Relative to PUMA's inference procedure, the natural-accuracy gain is 0.75 points $[-0.30, +1.86]$ with 15.8% fewer tokens.

At $w = 0.25$, Settle uses 16.8% fewer tokens than PUMA-SFT, a reduction of 471 tokens with paired 95% interval $[375, 563]$. Its natural-accuracy difference is $+0.85$ points $[-0.20, +1.96]$. Relative to released PUMA replay, it uses 32.1% fewer tokens, with a natural-accuracy difference of $+0.50$ points $[-0.65, +1.66]$. Both Settle weights are fixed settings from the main comparison.

C WHAT THE LEARNED STOPPING SCORE PREDICTS

We compare the closing-token score with elapsed-token predictors. The target is stable correctness: the current answer and every later answer are correct.

The native-score and elapsed-token comparisons use identical prefixes, positions, and stable-correct labels: 17,899 boundaries from 498 traces after removing the two training overlaps. The time-only predictor fits logistic regression to standardized cubic-spline features of $\log(1+t)$, with five training-quantile knots, inverse regularization strength 1, and constant extrapolation. It uses 12,000 training boundaries from 300 traces; complexity and regularization are fixed before evaluation. Each native score uses one trained model. Appendix I identifies the fits used in each analysis. Paired whole-trace bootstrap resamples use seed 0: 5,000 for overall and broad-range AUROC, and 2,000 for exact-token and current-correct analyses, following the convention in Appendix D.

We compare trained and base closing-token probabilities on the same prefixes and boundary indices; both use only the question and preceding reasoning, without the forced-answer prompt or readout. For $w = 0.10$ and $w = 0.25$, respectively, paired AUROC gains are 0.299 $[0.269, 0.329]$ and 0.306

$[0.277, 0.336]$ overall, and 0.300 $[0.266, 0.332]$ and 0.311 $[0.276, 0.343]$ at identical token counts. Among currently correct readouts at identical counts, the gains are 0.200 $[0.112, 0.290]$ and 0.210 $[0.119, 0.305]$.

The advantage also holds throughout the trace: native scores exceed the nonlinear time-only predictor in each of the five predefined token ranges (Table 17a), all of which contain both target classes. Across all boundaries, the paired AUROC improvements are 0.213 with 95% interval $[0.176, 0.250]$ for $w = 0.10$, and 0.221 with interval $[0.184, 0.257]$ for $w = 0.25$. This overall AUROC weights boundaries equally.

Pairs compare prefixes from different traces, since stable correctness is monotone within a trace.

Overall AUROC can reflect how far reasoning has progressed. We therefore compare prefixes at matched elapsed lengths. Let $\mathcal{P}_b$ and $\mathcal{N}_b$ be stable-correct and other boundaries in token-position group b. Conditional AUROC is

$$\mathrm{AUC}_{\mathrm{cond}}(s) = \frac{\sum_b \sum_{i \in \mathcal{P}_b} \sum_{j \in \mathcal{N}_b} \left[\mathbf{1}\{s_i > s_j\} + \frac{1}{2}\mathbf{1}\{s_i = s_j\} \right]}{\sum_b |\mathcal{P}_b|\,|\mathcal{N}_b|}.$$

Groups containing both classes contribute in proportion to their number of eligible positive–negative pairs. Table 17b reports all fixed 64-, 128-, and 256-token groups and exact integer-token matching. Time-only predictions are evaluated once per unique position and reused for all prefixes there, preserving exact ties.

Exact matching includes 12,106 boundaries at 2,315 positions from all 498 traces, forming 16,267 pairs. Each pair compares different traces. Time-only scores tie by construction, while the native scores distinguish stable-correct prefixes at the same elapsed position. Each bootstrap resample recomputes pair weights and the denominator.

Table 17: **Native-score prediction conditional on elapsed length.** AUROC [paired whole-trace bootstrap 95% interval] on 498 nonoverlapping MATH-500 traces. Panel (a) ranks boundaries within each broad range; panel (b) ranks pairs in the same position group. Exact matching admits only identical token counts. Every row uses the same fitted time-only predictor and one agreement-trained model per Settle column. All four matching rules include all 498 traces.

(a) Ranking within broad token ranges

Elapsed tokens	Boundaries	Time-only predictor	Settle $w = 0.10$	Settle $w = 0.25$
[0, 1024)	6,798	0.680 [0.657, 0.703]	0.842 [0.821, 0.863]	0.853 [0.831, 0.873]
[1024, 2048)	4,667	0.534 [0.508, 0.560]	0.824 [0.787, 0.858]	0.832 [0.795, 0.867]
[2048, 4096)	3,678	0.452 [0.421, 0.483]	0.766 [0.719, 0.814]	0.771 [0.723, 0.820]
[4096, 8192)	2,014	0.489 [0.442, 0.537]	0.743 [0.687, 0.798]	0.755 [0.700, 0.809]
[8192, 16384)	742	0.500 [0.500, 0.500]	0.681 [0.572, 0.782]	0.682 [0.575, 0.782]

(b) Ranking matched prefix pairs

Matching range	Boundaries	Pairs	Time-only predictor	Settle $w = 0.10$	Settle $w = 0.25$
64 tokens	17,843	1,024,057	0.506 [0.497, 0.516]	0.805 [0.781, 0.826]	0.816 [0.793, 0.838]
128 tokens	17,886	2,048,420	0.514 [0.504, 0.524]	0.805 [0.782, 0.827]	0.817 [0.793, 0.838]
256 tokens	17,899	4,100,706	0.522 [0.512, 0.532]	0.806 [0.783, 0.827]	0.817 [0.794, 0.839]
Exact token count	12,106	16,267	0.500 [0.500, 0.500]	0.807 [0.782, 0.831]	0.818 [0.793, 0.842]

To test persistence beyond present correctness, we restrict the comparison to prefixes whose current readout is correct. The subset contains 12,013 boundaries from 457 traces: 11,162 remain correct at every later readout, while 851 boundaries from 121 traces have a later incorrect readout.

Native scores still distinguish persistence within this subset (Table 18). Exact-token matching retains 2,455 boundaries from 425 traces at 585 positions, giving 2,015 pairs across traces. For uncertainty, we resample the original 498 traces, apply the correctness restriction, and recompute the statistic; all 2,000 draws contain valid comparisons.

Table 18: **Predicting whether a correct answer remains correct.** AUROC [95% whole-trace bootstrap interval] restricted to currently correct readouts. Overall uses 12,013 boundaries; exact-token matching uses 2,455. † Token-count predictors assign the same score to equal-length prefixes, giving conditional AUROC 0.5.

Score	Overall	Exact token count
Elapsed tokens	0.552 [0.479, 0.630]	$0.500^{\dagger}$
Time-only spline	0.551 [0.477, 0.630]	$0.500^{\dagger}$
Settle, $w = 0.10$	0.702 [0.645, 0.760]	0.695 [0.624, 0.764]
Settle, $w = 0.25$	0.715 [0.659, 0.771]	0.706 [0.635, 0.774]

D Training and evaluation protocol

Algorithm 1 Settle: hindsight supervision for the native stop token

Require: Base policy π_0, sampled rollouts, checker E, weights w, λ
for each recorded rollout x **do**
  Find candidate prefix boundaries $t_1 < \cdots < t_K$ and terminal point T.
  Generate short forced readouts $a_1, \ldots, a_K, a_T$.
  $r \leftarrow 1$
  for $j = K, K-1, \ldots, 1$ **do**
    $r \leftarrow r \cdot E(a_j, a_T)$; store target $y_j \leftarrow r$.
  end for
  Retain supervised boundaries within the training context.
end for
Fit a LoRA policy using the stop loss and reference KL in Equation 2.
Merge the adapter into the base model.
Ensure: Inference uses ordinary autoregressive decoding and the native `</think>` token.

We consider paragraph breaks and sentence endings followed by *Wait*, *But*, *Alternatively*, *Let me*, or *Hmm*. The first position must follow at least 32 generated tokens, and successive positions must be at least 16 tokens apart. If more than 40 positions remain, we retain at most 40 spaced uniformly through the trace. The terminal readout uses the prefix before the first closing tag, or the full available response when no closing tag occurs. The closing tag `</think>` is a single token for Qwen3 and R1-Distill-Qwen-7B. Nemotron splits the tag into multiple tokens; its adaptation supervises the first token, `</`, and generates the rest through ordinary decoding.

Following budget forcing (Muennighoff et al., 2025), at each position we append `</think>`, a `**Final Answer**` line, and the start of a boxed expression, then generate at most 16 tokens greedily. The same prompt extracts an answer when evaluation reaches the token cap. Ordinary evaluation uses each model's thinking-mode chat template with an instruction to put the final answer in a box. We grade the final boxed expression after the reasoning block with Math-Verify (Kydlíček, 2025).

We compare answers by their mathematical content. Normalization removes LaTeX delimiters, spacing commands, and other formatting differences; unequal normalized strings are tested for symbolic equivalence with Math-Verify. Parsing failures count as nonmatches. All 40,000 intermediate Qwen3-4B readouts are nonempty. The final readout determines the agreement targets but is excluded from supervised stopping positions.

For the reference-answer comparison, a position is *stable-correct* when its answer and all later readouts are correct, *stable-wrong* when all are incorrect, and *unstable* otherwise. Stable-wrong answers may still change from one incorrect value to another. The two training rules assign the following labels:

Reference-answer category	Agreement: stop	Agreement: continue
Stable-correct	20,089	0
Unstable	0	18,264
Stable-wrong	689	958

The two rules agree at 39,311 of 40,000 positions (98.28%). Reference-answer supervision assigns stop only to stable-correct positions, whereas agreement supervision also labels 689 consistently wrong positions as stop. These account for 3.32% of positive targets across 58 traces. Restricting training to the 8,192-token context leaves 39,649 supervised positions. Agreement and stable-correct labels describe the recorded continuation up to the 8,192-token cap, reached by 45.9% of Qwen3-4B training traces.

The stopping weight determines the penalty for continuing at a positive boundary. We select it by natural accuracy on 144 development problems. Reference-answer supervision considers $w \in \{0.10, 0.15, 0.25\}$ and selects 0.25. Agreement supervision considers $\{0.10, 0.25\}$ and selects 0.10. Appendix I records the development estimates and contributing fits. This agreement weight is carried to the other models without further tuning.

Table 19 gives the Qwen3-4B settings for attention-projection LoRA (Hu et al., 2021), trained on DAPO-Math-17K problems (Yu et al., 2025) with AdamW (Loshchilov & Hutter, 2019) and a frozen reference model. Training uses only Equation 2: the stopping and KL terms are summed over their respective positions without normalization. Gradients are averaged over eight accumulated examples.

Table 19: Qwen3-4B training settings.

Item	Setting
Problem pool	50 DAPO-Math-17K problems
Rollouts	20 per problem, 1,000 total
Rollout sampling	Temperature 1.0, top-p 1.0
Generation / training length limit	8,192 / 8,192 tokens
Readout budget	16 greedy tokens per read
LoRA rank / alpha / dropout	32 / 64 / 0
Target projections	Query, key, value, output
Epochs / optimizer updates	3 / 375
Batch / gradient accumulation	1 / 8
Optimizer / learning rate	AdamW / 3×10^{-4}
Weight decay / gradient clip	0 / 1.0
Precision	bfloat16, activation checkpointing
Stop loss weight / KL coefficient	0.10 / 0.20
KL direction	$\mathrm{KL}(\pi_0 \| \pi_\theta)$
Training seeds	0, 1, 2
Deployment	Adapter merged into base weights

The learning rate is 3×10^{-4} times

$$\min\left(1, \frac{s+1}{\max(1, \lfloor 0.1U \rfloor)}\right) \max\left(0, 1 - \frac{s}{\max(U, 1)}\right),$$

where s is the update step and $U = 375$. The schedule combines a warmup factor over the first 10% of updates with a linear decay factor over all updates.

Qwen3-4B training takes 3,366–3,374 seconds per run on an H100; one training-set readout pass takes 1,922 seconds. These timings exclude data selection, tuning, and evaluation. The collected traces are reused across training runs.

The loss directly changes the competition between closing the reasoning block and producing another token. If z_t is the closing-token logit and $p = p_\theta(t)$, its boundary-loss derivative is

$$\frac{\partial \ell_j}{\partial z_{t_j}} = (1 - y_j)p - w y_j (1 - p). \tag{4}$$

Stop targets push the closing-token logit upward; continue targets push it downward. The weight w scales the former, and inference uses the ordinary vocabulary softmax.

The scoring conventions in Section 3 apply throughout. For forced scoring, every response cut off by the token budget receives an answer prompt, including those whose reasoning block has already closed. Token means include these extra answer tokens unless a table specifies ordinary-generation counts. DEER's counts also include trial answers, their prompts, and discarded generation. Input processing is excluded; Appendix H reports measured inference time.

MATH-500 accuracy excludes problems 292 and 475 because they overlap the training data; token means use all 500 problems. MATH-500 uses four responses per problem.

We first average responses within each problem, also averaging across runs for replicated comparisons, then draw 20,000 paired problem bootstrap samples using NumPy's default generator with seed 0. The 2.5th and 97.5th percentiles give pointwise 95% intervals without adjustment for multiple comparisons. These intervals describe variation across problems for the evaluated fits; single-fit comparisons use the same procedure and do not estimate training-run variability. The diagnostics resample whole traces, with counts specified in Appendix C.

E When answers stabilize

Early stable answers define an opportunity for learned stopping. Table 20 measures that opportunity in separately sampled base-model traces: a retrospective rule chooses the first position whose answer remains correct at every later readout. When there is no such position, it uses the final answer, which may be obtained by a forced readout. The resulting accuracy and token counts describe what can be recovered from these collected traces.

Table 20: When answers stabilize in sampled reasoning traces. Settle is the percentage with an early stable-correct answer; median is the first such position as a fraction of trace length among those that settle. Accuracy and trace-token pairs are ordinary completion / retrospective stopping. "Repeat" denotes a separate Qwen3-4B MATH-500 sample.

Evaluation	Traces	Settle %	Median	Accuracy %	Trace tokens
Q3-4B MATH-500 8k	500	87.8	0.21	85.6 / 94.4	4,151 / 1,704
Q3-4B MATH-500 8k repeat	500	86.6	0.21	83.0 / 93.0	4,185 / 1,724
Q3-8B MATH-500 8k	500	85.4	0.19	82.0 / 93.2	4,396 / 1,776
Nemotron MATH-500 8k	500	83.6	0.32	90.0 / 93.0	2,937 / 1,535
Q3-4B MATH-500 16k	500	87.6	0.20	94.0 / 95.4	4,809 / 2,065

F Evaluation on separate MATH problems

On 500 further test problems, Settle reduces mean token count by 41.5% at 8k and 42.7% at 16k, with small positive mean changes in forced accuracy (Table 21). This sample from MATH (Hendrycks et al., 2021) excludes the training and development problems and all MATH problems already included in the benchmark evaluations.

Table 21: Separate 500-problem MATH test sample, Qwen3-4B. Forced accuracy, paired 95% intervals, and base / Settle token means, averaged over three runs.

Cap	Base	Settle	Change [95% interval]	Token count	Saved %
8k	95.45	95.87	+0.42 [-0.68, +1.55]	3,493 / 2,045	41.5
16k	96.33	96.52	+0.18 [-0.83, +1.22]	3,803 / 2,179	42.7

At 8k, natural accuracy rises by 4.08 points, indicating that earlier completion contributes to the larger natural-scoring gain.

G Accuracy and token counts across models and tasks

We next examine how the accuracy–token-count tradeoff changes across models and tasks while keeping the stopping weight at $w = 0.10$. Table 22 reports all four models on MATH-500 under both natural and forced scoring. Nemotron uses the first-token adaptation in Appendix D; the other models have a single-token closing tag.

Table 22: Results across four models. Accuracy pairs are base / Settle, in percent; forced-accuracy changes are percentage points. Each model uses three base runs and three trained runs. Token savings include additional answer tokens under forced scoring.

Model	Set	Natural	Forced	Forced change [95% CI]	Saved %
Qwen3-4B	MATH	93.89 / 94.38	95.23 / 94.73	-0.50 [-1.27, +0.27]	39.9
Qwen3-8B	MATH	93.71 / 94.66	95.67 / 95.45	-0.22 [-0.94, +0.50]	38.7
Nemotron-Nano-8B	MATH	94.61 / 92.99	94.81 / 93.06	-1.76 [-2.51, -1.04]	33.3
R1-Distill-Qwen-7B	MATH	92.82 / 91.03	93.56 / 91.62	-1.94 [-2.68, -1.22]	31.6

With the same stopping weight across tasks, Settle reduces mean token count on every benchmark in Table 23, with savings from 13.2% to 49.1%. The table gives the accompanying accuracy changes. The suite covers GSM8K (Cobbe et al., 2021); AMC23 (Mathematical Association of America); Minerva (Lewkowycz et al., 2022); OlympiadBench (He et al., 2024); and GPQA-Diamond (Rein et al., 2023). Each comparison uses one base run and three trained models with a common evaluation seed. Problem and response counts differ from the main evaluation and are listed in the table. The accompanying MATH-500 evaluation uses one response per problem, includes both training overlaps, and gives 95.2% / 94.9% base / Settle forced accuracy.

Table 23: Additional benchmarks, Qwen3-4B at 16k. Forced accuracy and token pairs are base / Settle. N denotes problems and S responses per problem. Results average three trained models against one base run.

Set	N	S	Accuracy %	Change	Token count	Saved %
GSM8K	1319	1	95.1 / 94.5	-0.7	2,053 / 1,167	43.2
AMC23	40	8	93.4 / 90.8	-2.6	7,617 / 4,903	35.6
Minerva	272	1	48.2 / 44.5	-3.7	6,536 / 3,327	49.1
OlympiadBench	674	1	66.6 / 70.4	+3.8	9,024 / 5,750	36.3
GPQA-Diamond	198	1	55.6 / 55.9	+0.3	8,565 / 7,433	13.2

G.1 Calibrating stopping pressure on harder problems

We train Qwen3-4B at $w \in \{0.01, 0.025, 0.05, 0.10\}$ on the original 1,000 traces, keeping $\lambda = 0.2$, the 375 training updates, and the remaining settings fixed. Each weight uses training seed 0. We choose the weight on 128 harder DAPO development problems, excluded from training and the evaluation benchmarks. The rule selects the lowest token count among settings whose forced accuracy is within one percentage point of the base model. Four responses per development problem select $w = 0.025$; the test comparisons below use this fixed development-selected weight.

Table 24: Development accuracy for Qwen3-4B stopping-pressure calibration. One training seed per weight and four responses per development problem, with evaluation seed 42 and a 16k cap. † The development-selected weight.

Method	Forced accuracy (%)
Base	85.55
Settle, $w = 0.01$	86.13
Settle, $w = 0.025^{\dagger}$	84.57
Settle, $w = 0.05$	84.18
Settle, $w = 0.10$	80.47

Table 25: The same Qwen3-4B weight sweep on MATH-500 and AMC23. Accuracy pairs are natural / forced percentages. Four responses per MATH problem and eight per AMC problem, evaluation seed 42, and one training seed per weight. MATH base uses the source generations from Table 16; all rows use the shared prompts, 16k cap, and scoring. † The weight selected on hard development problems.

	MATH-500		AMC23	
Method	Accuracy	Token count	Accuracy	Token count
Base	93.32 / 94.93	4,904	87.81 / 92.81	7,507
Settle, $w = 0.01$	93.22 / 94.83	4,789	90.62 / 94.06	7,392
Settle, $w = 0.025^{\dagger}$	93.98 / 95.13	4,410	89.38 / 91.88	6,896
Settle, $w = 0.05$	94.58 / 95.28	3,643	88.44 / 90.62	5,897
Settle, $w = 0.10$	94.18 / 94.58	2,890	89.06 / 89.69	4,822

On AMC23, the development-selected $w = 0.025$ reduces tokens by 8.1%, with a forced-accuracy difference of -0.94 points $[-3.44, +0.94]$. The token reduction is 611, with paired 95% interval $[338, 914]$.

G.2 Training on harder problems and reference-answer labels

We cross two training pools with agreement or reference-answer labels, keeping $w = 0.10$, $\lambda = 0.2$, 375 updates, and the remaining training settings fixed. Both pools contain 1,000 traces. The original pool uses 20 traces on each of 50 problems. The broader pool retains ten traces per original problem and adds two traces on each of 250 harder DAPO problems. We sample the added problems from a preliminary eight-response evaluation of 5,000 DAPO questions, retaining those with forced accuracy in $(0, 0.5]$ and excluding training, development, and benchmark overlaps. Qwen3-4B achieves 30.8% natural accuracy on the added traces under the 8k generation cap, compared with 54.0% on the original pool.

Reference-answer targets require every subsequent readout to be correct; all other boundaries receive continue targets. Agreement targets instead require agreement with the terminal readout. Within each pool, both conditions use identical traces, sampled positions, and KL masks. After the training-context limit, the label rules differ at 632 of 39,649 original-pool boundaries and 1,915 of 39,553 broader-pool boundaries. We train all four conditions with seeds 0–2, paired with evaluation seeds 42–44.

Table 26: Training-pool and label-rule comparison on Qwen3-4B, with $w = 0.10$ throughout. Means $\pm$ sample standard deviations across three training runs. Four responses per MATH problem, with a 16k cap. Token counts include prompted answers for capped responses.

Pool	Labels	Natural accuracy (%)	Forced accuracy (%)	Token count
MATH-500				
Original	Agreement	94.21 ± 0.15	94.66 ± 0.14	$2{,}943 \pm 49$
Original	Reference	94.31 ± 0.28	94.85 ± 0.16	$2{,}961 \pm 109$
Broader	Agreement	94.01 ± 0.21	94.38 ± 0.17	$3{,}073 \pm 73$
Broader	Reference	94.08 ± 0.46	94.98 ± 0.39	$3{,}169 \pm 164$

H Inference time under batching

We measure inference time on one H100 80GB using vLLM 0.28 (Kwon et al., 2023), with 100 MATH problems, one response per problem, and a 16k cap. Each setting is timed once after model loading. Settle uses the Qwen3-4B model trained with seed 1. Base and Settle use ordinary generation; DEER uses the procedure in Appendix B. Batch sizes 32 and 100 process fixed groups, with no new requests arriving during generation.

Table 27: Inference time for 100 MATH problems on one H100. Total time covers all requests; median time per request is reported only for batch size 1. Accuracy is natural accuracy on these requests. Token counts here cover responses only; the accuracy–cost comparisons above also count the additional work of DEER's intermediate checks.

Method	Batch	Total time (s)	Median (s)	Accuracy %	Response tokens
Base	1	1999.7	13.25	92	4,426
Base	32	323.7	–	94	4,214
Base	100	160.5	–	94	4,365
Settle	1	1165.2	7.66	91	2,602
Settle	32	301.9	–	94	2,621
Settle	100	120.0	–	90	2,670
DEER	1	1340.7	≈ 7.0	95	2,884
DEER	32	520.7	–	95	3,142
DEER	100	223.3	–	–	2,837

At batch size 1, Settle reduces median request time from 13.25 to 7.66 seconds; DEER's median is approximately 7 seconds at coarser timing resolution. For batches of 32 and 100, Settle completes the workload 1.72 and 1.86 times as fast as DEER. Table 27 gives accuracy alongside time for each setting. Its token column covers final responses; the policy comparisons additionally count DEER's intermediate checks.

I Results across individual runs

All tables in this section report individual runs at a 16,384-token response cap. MATH-500 accuracy uses 498 nonoverlapping problems; token means cover all 500. Inference-only methods use sampling seeds 42–44. Learned methods pair training seeds 0–2 with sampling seeds 42–44. For ordinary generation, the MATH-500 evaluator generates four responses per prompt with child seeds $r, \ldots, r+3$ for evaluation seed r, so runs 42–44 use overlapping seed ranges. DEER derives separate seeds for each sample and decoding stage. All responses from a problem remain together in the bootstrap.

Run accounting. The $w = 0.10$ main evaluation uses the second seed-0 fit plus seeds 1 and 2; the native-score analysis uses the first seed-0 fit with identical settings. Substituting the first fit changes three-run mean natural/forced accuracy by $+0.12/+0.30$ points at 8k and $-0.05/-0.07$ at 16k, with token changes below 0.3% at both caps. The $w = 0.25$ results use one fit per seed. In development selection, agreement weights 0.10 and 0.25 reach approximately 80.3% and 79.0% natural accuracy;

the latter estimate averages two trained models. One regularizer-removal evaluation used disjoint batches across three GPUs with the same per-response seeds.

Table 28: **Individual runs across models.** Accuracy pairs are natural / forced; token means include additional answer tokens under forced scoring.

Method	Seed	MATH-500	
		Accuracy (%)	Token count
Qwen3-4B			
Base	42	94.33 / 95.28	4,816
Base	43	94.03 / 95.43	4,809
Base	44	93.32 / 94.98	4,862
Settle	0	94.63 / 94.88	2,847
Settle	1	94.63 / 94.88	3,004
Settle	2	93.88 / 94.43	2,861
Qwen3-8B			
Base	42	93.32 / 95.53	5,102
Base	43	93.93 / 95.73	5,056
Base	44	93.88 / 95.73	5,177
Settle	0	94.68 / 95.58	3,201
Settle	1	94.18 / 95.03	3,055
Settle	2	95.13 / 95.73	3,150
Nemotron-Nano-8B			
Base	42	94.63 / 94.88	3,227
Base	43	94.78 / 94.88	3,234
Base	44	94.43 / 94.68	3,242
Settle	0	92.67 / 92.57	2,191
Settle	1	93.07 / 93.22	2,153
Settle	2	93.22 / 93.37	2,128
R1-Distill-Qwen-7B			
Base	42	92.87 / 93.83	3,743
Base	43	92.37 / 92.97	3,659
Base	44	93.22 / 93.88	3,657
Settle	0	91.16 / 91.77	2,611
Settle	1	90.81 / 91.47	2,518
Settle	2	91.11 / 91.62	2,440

Table 29: **Individual stopping-baseline runs.** Token counts cover ordinary generation. Epoch-2 unregularized truncated SFT appears in Table 30, with both token conventions.

Method and setting	Run	Natural (%)	Forced (%)	Tokens
Base (repeat evaluation)	0	93.22	94.83	4,905
	1	93.78	95.23	4,853
	2	93.32	94.78	4,874
Halt Vector	0	93.78	94.53	4,116
	1	93.83	94.63	4,137
	2	93.93	94.68	4,244
LSTM controller, $\tau = 0.95$	0	84.44	84.44	2,859
	1	88.45	88.96	3,457
	2	88.96	89.66	3,591
LSTM controller, $\tau = 0.99$	0	87.75	87.80	3,484
	1	92.47	93.47	4,392
	2	91.37	92.42	4,237

Continued on next page

Table 29: Individual stopping-baseline runs (continued)

Method and setting	Run	Natural (%)	Forced (%)	Tokens
LSTM controller, $\tau = 0.995$	0	89.11	89.26	3,767
	1	93.12	94.33	4,595
	2	91.82	92.92	4,424
LSTM controller, $\tau = 0.9999$	0	93.17	94.33	4,730
	1	93.78	95.23	4,853
	2	93.32	94.78	4,868
SFT: stability-truncated, 3×10^{-4}*, epoch 1*	0	63.81	63.81	1,544
	1	88.20	88.40	2,488
	2	88.91	89.11	2,718
Think Token Adjustment, $\alpha = 0.6$	0	93.52	95.08	4,900
	1	93.72	94.98	4,892
	2	93.37	94.73	4,824
Think Token Adjustment, $\alpha = 0.6$ *(answer format)*	0	86.14	86.80	3,988
	1	86.09	86.80	3,934
	2	88.96	90.06	4,185

Table 30: **Individual reference-regularization and DEER runs.** Ordinary token counts precede additional answer forcing; the final column includes it. DEER counts include intermediate checks.

Sampling seed	Natural (%)	Forced (%)	Token count	
			Ordinary	With forced answers
Truncated SFT, $\beta = 0$				
42	89.36	89.41	2,347.7	2,347.8
43	84.54	84.59	2,268.7	2,268.8
44	88.45	88.60	2,419.5	2,419.6
Truncated SFT, $\beta = 0.2$				
42	89.81	90.06	2,591.2	2,591.3
43	85.39	85.64	2,460.7	2,460.8
44	89.56	89.86	2,639.1	2,639.2
Truncated SFT, $\beta = 1.0$				
42	91.67	92.12	3,258.2	3,258.3
43	88.25	88.86	3,014.6	3,014.8
44	91.57	91.92	3,116.3	3,116.5
Truncated SFT, $\beta = 5.0$				
42	93.52	94.58	4,018.3	4,018.6
43	92.72	93.57	3,851.1	3,851.3
44	92.62	93.42	3,784.0	3,784.2
Settle, $w = 0.25$*,* $\lambda = 0$				
42	0.00	0.55	1,353.2	1,353.9
43	0.00	2.71	13,840.2	13,845.4
44	0.10	1.51	3,672.5	3,675.3
Settle, $w = 0.25$*,* $\lambda = 0.2$				
42	93.52	93.62	2,303.3	2,303.4
43	93.88	93.98	2,423.3	2,423.4
44	93.42	93.52	2,339.7	2,339.8
DEER, default				
42	94.68	94.68	3,641.5	3,641.6
43	94.98	94.98	3,661.4	3,661.4
44	94.78	94.78	3,655.1	3,655.2

Continued on next page

Table 30: Individual reference-regularization and DEER runs (continued)

Sampling seed	Natural (%)	Forced (%)	Token count	
			Ordinary	With forced answers
DEER, 100 trials				
42	94.88	94.88	3,131.9	3,131.9
43	94.83	94.83	3,133.8	3,133.8
44	94.73	94.73	3,193.6	3,193.7
DEER, $\theta = 0.50, \rho = 0.5$				
42	94.03	94.08	2,805.5	2,805.5
43	94.03	94.03	2,762.9	2,762.9
44	93.83	93.93	2,817.2	2,817.3
DEER, $\theta = 0.95, \rho = 0.25$				
42	92.82	92.82	2,650.7	2,650.7
43	92.82	92.92	2,647.4	2,647.5
44	92.52	92.52	2,674.6	2,674.6

Table 31: **Individual Settle reference-weight sensitivity runs.** Qwen3-4B, $w = 0.25$. These twelve fits use the same source traces and training settings as the main models, including three new runs at $\lambda = 0.20$. Token counts cover ordinary generation.

λ	Training seed	Sampling seed	Natural (%)	Forced (%)	Token count
0.05	0	42	92.87	92.87	2,040.1
0.05	1	43	93.17	93.22	2,048.3
0.05	2	44	93.07	93.12	2,082.6
0.10	0	42	93.12	93.12	2,135.9
0.10	1	43	92.72	92.77	2,177.0
0.10	2	44	93.12	93.12	2,188.3
0.20	0	42	93.52	93.57	2,289.9
0.20	1	43	93.17	93.37	2,378.4
0.20	2	44	92.87	92.97	2,345.1
0.40	0	42	94.53	94.63	2,702.6
0.40	1	43	93.78	94.18	2,764.3
0.40	2	44	93.62	93.62	2,618.3