FREE TO IGNORE: CORNERMAN DECODING FOR DISTILLED REASONERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Distilling long chain-of-thought (CoT) traces from stronger teachers offers a practical route to improving reasoning in small language models. These traces demonstrate both solution content and the organization of reasoning. Yet our analysis shows that students can produce useful intermediate reasoning without using it effectively to advance their solutions. This motivates guidance on how students develop, revise, and complete solutions during generation. We introduce Cornerman Decoding, which pairs a frozen student with a learned local adviser. The adviser observes only the latest student paragraph and either remains silent or suggests continuing, switching approaches, checking a claim, or finishing, without supplying new solution steps or answers. The student retains the full problem and reasoning history and is free to follow or ignore the advice. Across reasoning benchmarks, we evaluate the same frozen 1.5B adviser with 1.5B and 7B students fine-tuned on complete CoT traces. Compared with decoding the same students without advice, Cornerman improves macro-average accuracy by 5.5 and 1.0 percentage points, respectively, while reducing mean student-generated tokens by 45.2% and 21.1%. Paired case studies further illustrate how guidance can help students reach correct answers that do not appear in their uncoached trajectories.

1 INTRODUCTION

Large reasoning models (LRMs), such as OpenAI o1 (OpenAI, 2024) and DeepSeek-R1 (Guo et al., 2025), have demonstrated the value of long chain-of-thought (CoT) generation for solving complex problems. Distillation extends these advances to small language models (SLMs) by training them on teacher-generated reasoning traces, as illustrated by the DeepSeek-R1 distilled family. Related work, including LIMO (Ye et al., 2025) and s1 (Muennighoff et al., 2025), further demonstrates the effectiveness of supervised fine-tuning (SFT) on carefully curated reasoning demonstrations. These trajectories provide two intertwined forms of supervision: the content of a solution and the organization of the reasoning process (Wen et al., 2026). Beyond individual solution steps, they can demonstrate how to develop an approach, revisit doubtful claims, explore alternatives, and bring a solution to completion. The promise of long-CoT distillation is therefore to transfer not only solution content, but also patterns of reasoning that students can reuse on new problems.

However, learning from teacher demonstrations does not ensure that students can use these behaviors appropriately in their own reasoning. During SFT, a student learns to continue teacher-provided reasoning prefixes; at inference, it must decide how to proceed from the history it has generated itself (Agarwal et al., 2024). Our analysis of student trajectories (Section 2) reveals a gap between producing useful reasoning and using it effectively: students can struggle to build on partial progress, leave conflicting conclusions unresolved, or continue reasoning without submitting an answer they have already obtained. These observations suggest that improving reasoning is not only a matter of supplying additional solution content. It also calls for support in deciding how to develop, revise, and complete a solution as generation unfolds.

One way to provide this support is to guide the model's choice of reasoning behaviors during generation. Prompt Intervention (PI; Yang et al., 2026a) steers reasoning through behavior-specific prompts and, in its dynamic variant, selects among candidate continuations. ACTS (Xia et al., 2026) trains a controller that uses the full reasoning history and remaining budget to select a strategy and a steering phrase that opens the student's next reasoning step. The controller can also terminate

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.

the thinking phase. These approaches demonstrate the value of explicit structural guidance, while assigning decisions about continuation or termination to an external mechanism. Yet the student is already developing its own solution, and the value of an intervention depends on how it relates to that ongoing work. Encouraging a different approach, for example, can help escape an unproductive route, but can also interrupt useful reasoning that has not yet been completed.

This motivates guidance that informs the student's next move without determining it. The student's latest paragraph provides local evidence of how its reasoning is progressing and where a reminder might help. Recognizing such an opportunity does not necessarily require reconstructing the full solution; deciding how to respond, however, may require the original problem and earlier reasoning. We therefore ask whether an adviser can learn what structural advice, if any, a student's current paragraph warrants, while leaving the student to decide how to continue.

To address this question, we introduce Cornerman Decoding, which pairs a frozen student with a learned local adviser. We fine-tune the adviser on annotated paragraphs drawn from students' own reasoning trajectories. Rather than labeling which reasoning behavior a paragraph exhibits, the supervision specifies whether a reminder is warranted and what it should encourage. The adviser learns to remain silent or offer one of four suggestions: continue useful unfinished work, switch approaches, check an existing claim, or finish the solution. During generation, the adviser observes only the latest paragraph. Its suggestions are inserted as short structural reminders, without supplying new solution steps or answers. The student retains the original problem and all its generated reasoning, continuing along a single decoding trajectory without external selection among candidate continuations.

We evaluate Cornerman with 1.5B and 7B students fine-tuned on complete teacher reasoning traces, using the same frozen 1.5B adviser at both scales. Across reasoning benchmarks, Cornerman improves macro-average accuracy by 5.5 and 1.0 percentage points, respectively, compared with decoding the same students without advice, while reducing mean student-generated tokens by 45.2% and 21.1%. Paired case studies further illustrate how guidance can support both submitting answers already obtained and revising the reasoning to reach correct answers absent from the corresponding uncoached trajectories. These observations connect the aggregate improvements to the development, revision, and completion of students' own solutions.

Our contributions are threefold:

- We introduce Cornerman Decoding, which provides local, non-binding structural guidance to a frozen student while preserving its full reasoning context and control over continuation.
- We formulate adviser training around explicit judgments of whether and how to intervene in student-generated reasoning. Paragraph-level supervision teaches the adviser both when to remain silent and which structural suggestion to offer.
- We demonstrate higher macro-average accuracy alongside shorter student-generated output at two model scales, and examine the roles of the learned policy and student–adviser interaction through ablations and paired case studies.

2 Motivation and Observations

An incorrect final response does not necessarily mean that the student never reached a correct answer. We analyze a Full-CoT student: DeepSeek-R1-Distill-Qwen-1.5B further fine-tuned on complete teacher reasoning traces. Among its audited MATH500 failures under greedy decoding (Appendix C.1), 49.1% contain a correct answer that is not submitted, while 61.3% exhibit repetition loops; these categories overlap. These findings reveal a gap between producing useful reasoning and carrying it through to a successful solution.

The Fibonacci example in Figure 1(a) illustrates how this gap can arise during checking. Asked to compute $F_{100} \bmod 4$, the student identifies a period of six and correctly obtains three. It then revisits the cycle's starting position, introduces an indexing error, and obtains two. Although it explicitly recognizes the contradiction, it ultimately submits two. The student therefore exhibits checking behavior, but fails to use it to resolve conflicting conclusions.

Decisions to continue or change approaches likewise depend on the progress already made. In our adviser-training examples, one student derives $c \leq 16$ before writing, "Wait, maybe another

approach," providing a reason to encourage further work on the unfinished route. Another repeats a numerical identity and concludes, "Hmm, not helpful," suggesting that a different route may be worth exploring. Steady progress, meanwhile, may warrant no reminder. These contrasts motivate supervising what assistance, if any, a paragraph calls for, rather than merely labeling the behavior it exhibits. Complete examples and annotations are provided in Appendix A.4.

Identifying a need for guidance, however, does not mean knowing how to resolve the underlying difficulty. A paragraph may expose conflicting conclusions, while determining which is correct may require the question and earlier reasoning. We therefore separate local guidance from full-context reasoning: the adviser uses the latest paragraph to suggest how to proceed, while the student retains the complete context to develop the solution. Because the adviser's view may omit relevant information, its suggestions remain non-binding; the student decides whether and how to use them.

3 CORNERMAN DECODING

We implement this division of responsibilities through Cornerman Decoding (Figure 1). The adviser is trained on student-generated paragraphs to judge what structural reminder, if any, is warranted, rather than simply identify the reasoning behaviour a paragraph exhibits. During generation, it observes only the latest paragraph and either remains silent or offers a reminder. The frozen student retains the full problem and reasoning history and generates its own continuation, deciding whether and how to use the advice.

3.1 LOCAL STRUCTURAL ADVICE

We use the student's latest paragraph as the unit of observation. A paragraph can reveal conflicting claims or useful work that the student is about to abandon, providing a reason to offer a reminder without establishing how the underlying problem should be solved. This distinction motivates the adviser's limited role: it identifies a possible need for structural guidance from local evidence, while the student retains the full context needed to resolve the difficulty and determine how to continue.

The adviser chooses among five verdicts: PASS, STEP, ALT, CHECK, and END (Table 1). Useful progress does not by itself warrant STEP: a student already pursuing it needs no additional reminder and receives PASS. STEP instead supports useful, unfinished work that the student appears ready to abandon, whereas ALT suggests changing routes when the current approach has stalled. The distinction depends on whether the approach still offers productive work, not simply whether the student expresses uncertainty. CHECK suggests reconsidering a specific claim, while END suggests submitting an answer when further reasoning no longer appears useful. PASS also applies when the paragraph provides insufficient evidence for intervention.

Non-PASS verdicts are expressed as short first-person reminders that suggest a next move within the student's ongoing reasoning (Table 1). STEP, ALT, and END use fixed messages; CHECK combines a fixed reminder with a verbatim quotation from the student's paragraph. The quotation identifies what to reconsider without supplying a correction. None of these messages provides new solution steps or answers. END likewise suggests submitting an answer rather than directly terminating generation.

3.2 LEARNING THE ADVISER

We train the adviser on student-generated paragraphs to reflect the reasoning states it will encounter during decoding. The dataset contains approximately 2,400 labelled examples from successful and unsuccessful Full-CoT student trajectories. The problems used to collect adviser-training paragraphs are disjoint from the MATH500 evaluation set. Each annotation identifies what assistance the visible paragraph warrants, without relying on the original question, unseen earlier reasoning, or the trajectory's eventual correctness. This restriction aligns the annotation task with the adviser's local view at inference. CHECK annotations additionally specify a verbatim span to reconsider. Labels are produced using the DeepSeek API, with a sample reviewed by the authors.

We initialize Cornerman from Qwen2.5-1.5B-Instruct (Qwen et al., 2025) and train it with full-parameter SFT. Given a student paragraph and a fixed task instruction, the adviser learns to output either PASS or a non-silent verdict with its corresponding first-person reminder. For CHECK, the

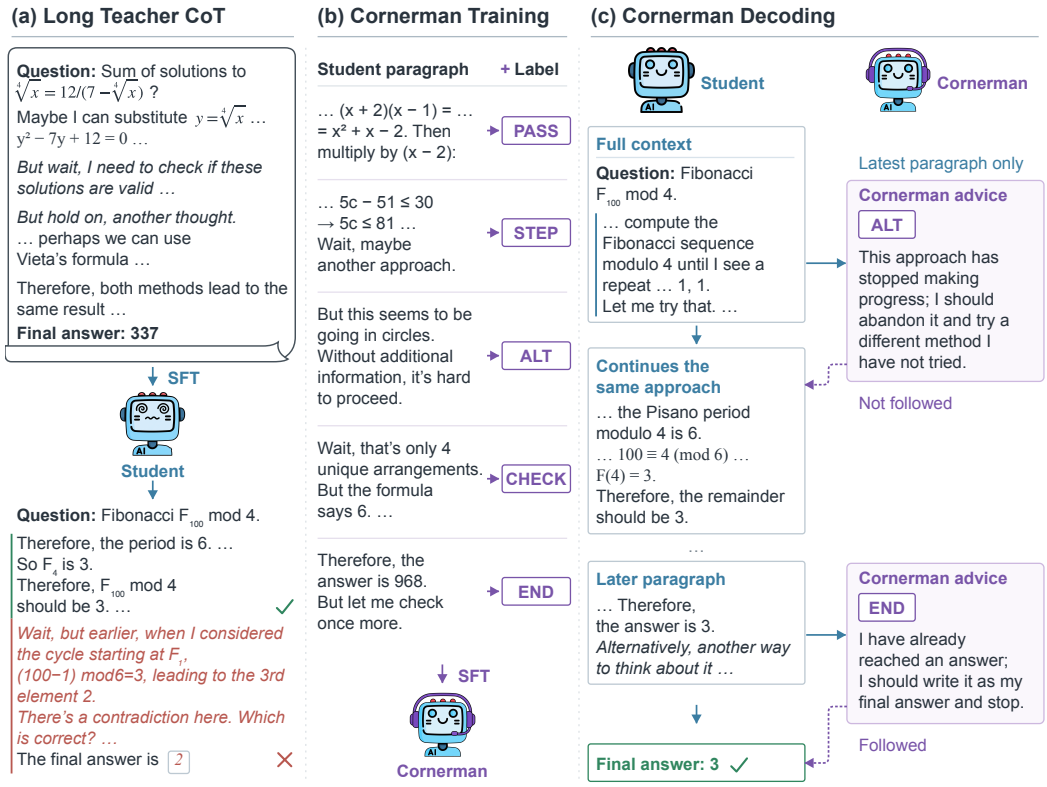


Figure 1: Overview of Cornerman Decoding. (a) The teacher trace demonstrates checking and exploring alternative approaches. The student example shows a correct result being revised into an incorrect answer. (b) Student paragraphs are annotated according to what structural guidance they need, including no intervention, and used to train the adviser. (c) A schematic interaction shows the student continuing its approach despite an ALT suggestion and later following END to submit its answer. Cornerman observes only the latest paragraph, while the student retains the full problem and reasoning history. Purple boxes show advice inserted as first-person reminders. Ellipses indicate omitted text and interactions.

reminder includes the span selected during annotation. Training updates only the adviser; the Full-CoT student remains frozen. The complete instruction and optimization settings are provided in Appendices A.2 and B.1, respectively.

3.3 DECODING WITH ADVICE

At inference, both models remain frozen. Paragraph tags learned during Full-CoT training identify boundaries for consulting the adviser. At each consultation, the adviser receives the latest paragraph with its tags removed. For a valid non-PASS reply, we remove any previous advice and insert its first-person reminder, preserving all student-generated reasoning. A PASS or invalid reply inserts nothing and leaves any existing reminder unchanged.

The student resumes generation with the original problem, its accumulated reasoning, and any retained reminder. The interaction continues until the student ends its response, exhausts its generation budget, or triggers a runtime stopping safeguard, including segment, round, and context limits. Algorithm 1 summarizes the decoding procedure; input preprocessing and runtime details are specified in Appendix A.5.

Table 1: Cornerman's advisory interface. PASS inserts no message. For CHECK, <QUOTE> is replaced with a verbatim span from the observed student paragraph.

Verdict	When guidance is warranted	Inserted message
PASS	Ordinary progress or insufficient evidence for intervention	No message.
STEP	Useful, unfinished work is being abandoned	This line is making progress; I should keep pushing it before trying anything else.
ALT	The current approach has stalled without useful progress	This approach has stopped making progress; I should abandon it and try a different method I have not tried.
CHECK	A specific visible claim warrants reconsideration	Before continuing, I should re-examine this claim I just made: "<QUOTE>"
END	An answer has been reached and continued reasoning adds nothing useful	I have already reached an answer; I should write it as my final answer and stop.

Algorithm 1 Cornerman Decoding

Require: Problem x; frozen student S_θ; frozen adviser C_ϕ; student token budget B
Ensure: Student-generated response

Set the fixed system instruction s to "Please reason step by step, and put your final answer within \boxed{}."
Initialize student context using its chat template with system instruction s, user problem x, and an assistant continuation beginning with <think>; no advice is present.
Initialize remaining student token budget $b \leftarrow B$.
while $b > 0$ and no stopping safeguard is triggered **do**
    Continue S_θ to a paragraph boundary, chunk limit, or termination, generating at most b new tokens; append student text.
    $b \leftarrow b - n$, where n is the number of student tokens generated in this iteration.
    if the student terminates, $b = 0$, or a stopping safeguard is triggered **then**
        return the student's generated response
    end if
    Extract the tag-free local observation w from the latest student paragraph.
    Query $C_\phi(w)$ for a verdict v and its corresponding reply.
    if the reply is valid and $v \neq$ PASS **then**
        Remove old advice, if any; append the new reminder and preserve all student reasoning.
    end if
end while
return the student's generated response

4 EXPERIMENTS

4.1 EVALUATION SETTING

We evaluate DeepSeek-R1-Distill-Qwen students at 1.5B and 7B (Guo et al., 2025). *Base* denotes the released distilled checkpoint, and *Full-CoT* denotes the student further fine-tuned on complete teacher traces from LIMO (Ye et al., 2025). *Cornerman Decoding* adds advice to the same Full-CoT student without updating its parameters. We use one frozen 1.5B adviser for both student scales. Training and decoding details are provided in Appendices B and A.5.

Our evaluation follows the six-benchmark suite used by SLA (Wang et al., 2026): MATH500 (Hendrycks et al., 2021; Lightman et al., 2024), AMC23, AIME24, GPQA-Diamond (Rein et al., 2024), Minerva (Lewkowycz et al., 2022), and the English text-only mathematics subset of OlympiadBench (He et al., 2024). We add GSM8K (Cobbe et al., 2021), AIME25, and the public validation split of CommonsenseQA (CSQA; Talmor et al., 2019), yielding nine benchmarks spanning mathematical, scientific, and commonsense reasoning. No benchmark-specific adviser training is performed.

Table 2: Single-run greedy accuracy and mean student output length across nine benchmarks. Each scale compares the released Base checkpoint, its Full-CoT fine-tuned counterpart, and Cornerman Decoding with that Full-CoT student. Overall is an equal-weight average across benchmarks.

	Base		Full-CoT		Cornerman Decoding	
Benchmark	Acc. (%) ↑	Tokens ↓	Acc. (%) ↑	Tokens ↓	Acc. (%) ↑	Tokens ↓
1.5B student						
GSM8K	74.9	2,908	76.6	1,634	**77.4**	**947**
MATH500	65.0	11,838	78.8	7,822	**84.4**	**4,128**
Minerva	26.8	14,231	26.1	16,588	**32.7**	**5,817**
AMC23	42.5	18,626	65.0	13,078	**65.0**	**8,391**
AIME24	20.0	25,522	26.7	23,018	**33.3**	**17,257**
AIME25	16.7	27,816	10.0	26,408	**23.3**	**16,265**
OlympiadBench	33.5	20,103	43.9	16,765	**48.7**	**9,259**
GPQA-Diamond	31.3	25,442	26.3	24,712	**33.3**	**11,712**
CSQA	40.8	4,347	40.3	7,042	**44.9**	**1,377**
Overall	39.1	16,759	43.7	15,230	**49.2**	**8,350**
7B student						
GSM8K	**89.8**	1,912	86.1	667	86.7	**621**
MATH500	87.4	5,621	93.0	3,911	**93.0**	**3,328**
Minerva	41.5	7,979	39.0	5,507	**42.6**	**4,098**
AMC23	80.0	9,442	87.5	7,358	**87.5**	**6,353**
AIME24	36.7	20,730	53.3	14,175	**56.7**	**10,317**
AIME25	33.3	21,008	**40.0**	14,756	36.7	**12,163**
OlympiadBench	56.7	13,549	**63.6**	8,758	61.7	**7,150**
GPQA-Diamond	40.9	16,998	40.4	8,712	**46.5**	**6,275**
CSQA	**66.3**	1,310	62.0	969	62.2	**816**
Overall	59.2	10,950	62.8	7,201	**63.7**	**5,680**

The main results use greedy decoding with a maximum of 32,768 student-generated tokens per problem. We score submitted final answers using mathematical equivalence for mathematical tasks and option matching for multiple-choice tasks; missing or invalid answers receive no credit. Appendix B.3 specifies the extraction rules. Overall accuracy is the unweighted mean of the nine benchmark accuracies, and overall length is the unweighted mean of their mean student output lengths. Tokens count student-generated output, excluding adviser output and inserted messages. They measure output length rather than total system computation. We examine sampled decoding separately in Section 4.6.

4.2 Accuracy and output length

Table 2 compares Base, Full-CoT, and Cornerman Decoding. For the 1.5B student, Full-CoT training raises macro-average accuracy from 39.1% to 43.7%. Adding Cornerman raises it further to 49.2%, while reducing mean student output from 15,230 to 8,350 tokens, a reduction of 45.2%. These gains are obtained with the Full-CoT student's parameters held fixed: the additional component provides structural advice during generation rather than new solution demonstrations.

At 7B, the same adviser increases macro-average accuracy from 62.8% to 63.7%, while reducing mean student output from 7,201 to 5,680 tokens (21.1%). On MATH500, the 7B student retains 93.0% accuracy with shorter output, whereas the 1.5B student improves from 78.8% to 84.4%. The two scales thus illustrate complementary effects: a larger accuracy gain for the smaller student and a smaller aggregate gain accompanied by shorter output for the larger student.

The benefits also extend beyond mathematical questions. On GPQA-Diamond, Cornerman raises accuracy from 26.3% to 33.3% at 1.5B and from 40.4% to 46.5% at 7B. On CSQA, the 1.5B student improves from 40.3% to 44.9%, with mean output falling from 7,042 to 1,377 tokens. These results show that the same structural interface can support different kinds of reasoning without task-specific advice messages.

Table 3: Comparison at 1.5B on a common four-benchmark subset. Acc. is final-answer accuracy (%); tokens count student-generated output. Overall gives equal weight to each benchmark for both metrics. ACTS and DEER use the Full-CoT student; JET and LC-R1 involve reasoner post-training.

	MATH500		AMC23		AIME24		GPQA-Diamond		Overall	
Method	Acc. ↑	Tokens ↓	Acc. ↑	Tokens ↓	Acc. ↑	Tokens ↓	Acc. ↑	Tokens ↓	Acc. ↑	Tokens ↓
Full-CoT	78.8	7,822	65.0	13,078	26.7	23,018	26.3	24,712	49.2	17,158
ACTS	81.0	**2,513**	**70.0**	**4,509**	33.3	**11,461**	27.8	**3,847**	53.0	**5,583**
DEER	73.8	4,504	57.5	9,425	23.3	15,809	30.8	16,884	46.4	11,656
JET	83.6	3,317	**70.0**	5,524	30.0	15,846	30.3	14,421	53.5	9,777
LC-R1	81.0	4,743	57.5	9,838	13.3	22,114	29.3	20,298	45.3	14,248
Cornerman Decoding	**84.4**	4,128	65.0	8,391	**33.3**	17,257	**33.3**	11,712	**54.0**	10,372

Table 4: Learned-policy results and reference conditions on MATH500 at 1.5B. Speak rate is the fraction of eligible consultations that insert a message. The trained row is the Cornerman result in Table 2. Appendix B.3 defines the controls and their messages.

Condition	Accuracy (%) ↑	Student tokens ↓	Speak rate (%)
Full-CoT (decoded alone)	78.8	7,822	—
Silent framework	77.4	5,397	0.0
Untrained adviser	77.6	4,386	68.0
Sham advice	77.4	5,191	79.0
Trained adviser	**84.4**	**4,128**	33.5

4.3 Comparison with reasoning-time interventions

We compare Cornerman with ACTS (Xia et al., 2026), DEER (Yang et al., 2026b), JET (Han et al., 2026a), and LC-R1 (Cheng et al., 2026) on MATH500, AMC23, AIME24, and GPQA-Diamond at 1.5B. These methods approach the accuracy–length trade-off through external strategy control, confidence-based early exit, or reasoner post-training. ACTS and DEER are applied to the Full-CoT student; JET and LC-R1 update the reasoner, whereas Cornerman trains a separate adviser and leaves the student frozen.

Cornerman attains 54.0% macro-average accuracy on this subset, compared with 49.2% for Full-CoT, and reduces mean student output from 17,158 to 10,372 tokens. It achieves the highest accuracy on MATH500 and GPQA-Diamond and matches ACTS on AIME24, while ACTS and JET score higher on AMC23. JET is the closest competitor in overall accuracy, reaching 53.5% with 9,777 tokens on average; Cornerman's 0.5 percentage point advantage is accompanied by 6.1% longer output in this evaluation. ACTS provides greater output compression, averaging 5,583 tokens at 53.0% accuracy. Whereas JET post-trains the reasoner, Cornerman trains a separate adviser and obtains its gains through local, non-binding advice, leaving the Full-CoT student's parameters unchanged and the student in control of its solution.

4.4 What does the learned policy contribute?

A shorter output does not by itself establish the value of learned advice. Table 4 places Cornerman alongside three reference conditions: a silent framework that segments generation without inserting advice, an untrained adviser, and unrelated sham messages. Unlike standalone Full-CoT decoding, the silent framework also imposes a segment-count limit, which can terminate generation before the shared token budget is exhausted. Its shorter outputs therefore do not by themselves indicate more efficient reasoning. The silent condition serves as a control for assessing advice within the segmented decoding framework. The silent framework already shortens output, while the untrained adviser generates 4,386 tokens on average, close to Cornerman's 4,128, but reaches only 77.6% accuracy rather than 84.4%. Frequent intervention alone is therefore not sufficient to reproduce the observed accuracy. Cornerman speaks at 33.5% of consultations, considerably less often than the untrained and sham conditions. The results support learning when and what to suggest, including when to remain silent.

Table 5: Sampled decoding with the 1.5B student. Per-benchmark accuracies (%) are reported as mean (avg@6) ± sample standard deviation across decoding seeds 1–6, with temperature 0.6, top-$p = 1.0$, and model parameters held fixed. Overall accuracy and mean student tokens give equal weight to all nine benchmarks.

Benchmark	Base	Full-CoT	Cornerman Decoding
GSM8K	**77.5** ± 0.3	75.2 ± 1.3	76.7 ± 0.4
MATH500	84.8 ± 0.8	84.1 ± 0.7	**85.2** ± 0.6
Minerva	31.3 ± 1.6	33.2 ± 1.0	**34.6** ± 1.0
AMC23	**72.9** ± 2.9	69.6 ± 1.9	68.3 ± 2.0
AIME24	31.7 ± 1.8	31.7 ± 2.8	**35.0** ± 4.1
AIME25	20.0 ± 6.7	24.4 ± 1.7	**25.0** ± 5.5
OlympiadBench	49.2 ± 0.5	49.9 ± 1.3	**50.1** ± 0.3
GPQA-Diamond	30.1 ± 1.1	27.0 ± 1.1	**32.6** ± 1.1
CSQA	37.3 ± 0.1	**40.9** ± 1.6	40.5 ± 1.3
Overall accuracy (%)	48.3	48.4	**49.8**
Mean student tokens	7,757	9,115	**6,383**

4.5 How advice enters the trajectory

Effect of the first suggestion. We next examine whether the first suggestion alone improves the student's final answer. At the first non-silent intervention, we create two continuations from the same reasoning prefix: one retains the suggestion, while the other removes it. Both use the same student checkpoint, and neither receives further advice. Across 1,260 paired replays drawn from the seven mathematical benchmarks, retaining the suggestion yields 649 correct answers (51.5%), compared with 669 (53.1%) when it is removed. On this replay subset, the first suggestion does not improve accuracy in isolation. The result distinguishes the effect of a single reminder from that of the full interaction, in which advice remains available as reasoning develops. Appendix C.4 gives the subset composition and results.

Selective uptake of advice. Students do not simply execute Cornerman's verdicts. In our trajectory audit, frequency-weighted adherence in the immediately following paragraph is 58% for the 1.5B student and 60% for the 7B student. CHECK is followed more consistently than ALT or END: students more readily revisit a flagged claim than switch approaches or submit an answer when prompted. In particular, an END suggestion often does not lead to immediate answer submission. The adviser's judgment that a solution appears complete therefore remains distinct from the student's decision to conclude. These observations illustrate the non-binding interaction in practice: Cornerman proposes a structural action, but does not directly enact it. Appendix C.3 reports the annotation criteria and verdict-level counts.

Figure 1(c) schematically illustrates this distinction. The student does not follow an ALT suggestion, instead continuing its Fibonacci calculation and reaching the correct remainder, 3. Later, when it considers another approach despite already having the answer, it follows END and submits 3. These different responses make the division of responsibilities concrete: Cornerman offers guidance on how to proceed, while the student performs the calculation and retains control over its continuation. Further paired examples and complete coached trajectories appear in Appendices D and E.

4.6 Sampled decoding

Cornerman yields a modest accuracy gain and substantially shorter outputs under sampled decoding. Table 5 reports avg@6 results for the 1.5B student over decoding seeds 1–6, with model parameters held fixed. Across the nine benchmarks, Cornerman increases macro-average accuracy from 48.4% to 49.8%, while reducing mean student-generated output from 9,115 to 6,383 tokens, a reduction of 30.0%. The smaller accuracy gain than under greedy decoding may reflect overlapping benefits of sampling and structural guidance: sampling already raises the uncoached Full-CoT baseline from 43.7% to 48.4%, potentially alleviating some of the reasoning difficulties that advice addresses. Sampling settings and metric definitions are provided in Appendix C.5.

5 Related work

Learning reasoning behaviours. Reasoning distillation transfers capabilities through teacher-generated solution traces, as demonstrated by the DeepSeek-R1 distilled family (Guo et al., 2025). Related work, including LIMO (Ye et al., 2025) and s1 (Muennighoff et al., 2025), highlights the effectiveness of SFT on carefully curated reasoning demonstrations. Beyond learning from textual traces, Reasoning Scaffolding (Wen et al., 2026) explicitly represents their structure through semantic signals and jointly trains next-signal prediction and signal-conditioned generation. This formulation makes the organization of teacher reasoning an explicit supervision target. Cornerman builds on reasoning distillation but supervises a different judgment: what assistance a student-generated paragraph calls for, including when no intervention is needed.

Self-correction and refinement. Research on self-correction investigates whether models can improve their own responses through reconsideration and revision. Prompting alone does not reliably improve reasoning accuracy without external feedback (Huang et al., 2024). SCoRe (Kumar et al., 2025) addresses this difficulty through multi-turn reinforcement learning on self-generated correction trajectories. PASR (Han et al., 2026b) extends refinement to ongoing generation, learning whether, when, and how to refine a developing response. These approaches train the reasoner itself to improve its revisions. Cornerman instead learns when to offer a structural reminder, leaving the student to determine whether and how the underlying reasoning should change.

Efficient reasoning. Research on efficient reasoning seeks to reduce unnecessary generation while preserving solution quality. One line of work trains models to produce shorter reasoning trajectories. JET (Han et al., 2026a) combines truncated rollouts with quality-controlled length rewards. ARLCP (Yu et al., 2026) coordinates reflection and length penalties, while LC-R1 (Cheng et al., 2026) uses length and compression rewards to discourage redundant thinking. Another approach controls termination during inference without additional training: DEER (Yang et al., 2026b) evaluates confidence in trial answers at reasoning transition points to determine when to exit. Cornerman addresses a complementary question: what assistance is useful during ongoing reasoning, where guidance may support further progress or reconsideration as well as completion.

External steering. Steering methods intervene during generation to influence which reasoning behaviour comes next. Prompt Intervention (PI; Yang et al., 2026a) inserts behaviour-specific prompts; its dynamic variant compares prompted continuations with the model's natural continuation and selects a branch using model-derived scores. ACTS (Xia et al., 2026) learns a controller that observes the full interaction history and remaining budget, selects a strategy, and supplies the opening phrase of the next reasoning step. Its `conclude` action can terminate the thinking phase. Cornerman instead offers optional structural advice based on the latest student paragraph. The adviser neither selects among candidate continuations nor directly terminates reasoning, leaving the student to interpret the suggestion and generate its own continuation using the full context.

6 Discussion and conclusion

We introduced Cornerman Decoding to help distilled students use their learned reasoning behaviours more effectively during generation. Trained on student-generated paragraphs, a local adviser learns what structural assistance is warranted, while the student retains the complete context and control of its continuation. Across nine benchmarks, Cornerman improves macro-average accuracy and reduces student output length for both 1.5B and 7B students, with gains also observed under sampled decoding at 1.5B. These findings support structural advice as a way to help students develop, revise, and complete their own solutions without the adviser supplying new solution steps or answers. Future work should examine transfer across model families, extend task coverage, and quantify end-to-end inference cost.

AI USE STATEMENT

We used generative AI tools to assist with method implementation, experimental-design discussions, data cleaning and formatting, qualitative trajectory analysis, interpretation of results, translation, and manuscript polishing. AI tools also supported literature search and analysis and figure preparation. The DeepSeek API assisted with annotation of the Cornerman training data, and the authors manually reviewed a sample of the labels. The core method was conceived by the authors. The authors manually reviewed key code implementations, checked all AI-assisted trajectory analyses and result interpretations against the original records, and reviewed the manuscript. We take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We provide training, decoding, and evaluation code in the supplementary archive and at `https://anonymous.4open.science/r/cornerman-decoding/`. Appendix A specifies the advice messages, annotation criteria, complete prompts, and decoding procedure; Appendix B documents training settings and evaluation protocols. Appendix C provides supplementary analyses, and Appendices D and E provide illustrative cases and complete coached trajectories.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024. URL `https://proceedings.iclr.cc/paper_files/paper/2024/file/5be69a584901a26c521c2b51e40a4c20-Paper-Conference.pdf`.

Zhengxiang Cheng, Dongping Chen, Mingyang Fu, and Tianyi Zhou. Optimizing length compression in large reasoning models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3232–3250, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.146. URL `https://aclanthology.org/2026.acl-long.146/`.

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, 2021. URL `https://arxiv.org/abs/2110.14168`.

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, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, 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, Mingxu Zhou, 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, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, 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, 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 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645 (8081):633–638, September 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL https://doi.org/10.1038/s41586-025-09422-z.

Jinyi Han, Ying Huang, Ying Liao, Haiquan Zhao, Zishang Jiang, Xinyi Wang, Xikun Lu, Guanghao Zhou, Sihang Jiang, Jiaqing Liang, Weikang Zhou, Zeye Sun, Fei Yu, and Yanghua Xiao. Your Models Have Thought Enough: Training Large Reasoning Models to Stop Overthinking. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 141341–141365, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/e4ebe9ad8be5a585d01e250716e81ded-Paper-Conference.pdf.

Jinyi Han, Xinyi Wang, Haiquan Zhao, Tingyun Li, Zishang Jiang, Sihang Jiang, Jiaqing Liang, Xin Lin, Weikang Zhou, Zeye Sun, Fei Yu, and Yanghua Xiao. A Stitch in Time Saves Nine: Proactive Self-Refinement for Language Models. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 151015–151039, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/f501955a5d98997ea4cc622089cba7ca-Paper-Conference.pdf.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.211. URL https://aclanthology.org/2024.acl-long.211/.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset. In J. Vanschoren and S. Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/be83ab3ecd0db773eb2dc1b0a17836a1-Paper-round2.pdf.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large Language Models Cannot Self-Correct Reasoning Yet. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/8b4add8b0aa8749d80a34ca5d941c355-Paper-Conference.pdf.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, JD Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. Training Language Models to Self-Correct via Reinforcement Learning. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 54523–54549, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/871ac99fdc5282d0301934d23945ebaa-Paper-Conference.pdf.

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. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 3843–3857. Curran Associates, Inc., 2022. doi: 10.52202/068431-0278. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/18abbeef8cfe9203fdf9053c9c4fe191-Paper-Conference.pdf.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's Verify Step by Step. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/aca97732e30bcf1303bc22ac3924fd16-Paper-Conference.pdf.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20275–20321, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1025. URL https://aclanthology.org/2025.emnlp-main.1025/.

OpenAI. Learning to reason with LLMs, September 2024. URL https://openai.com/index/learning-to-reason-with-llms/. September 12, 2024.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 Technical Report, 2025. URL https://arxiv.org/abs/2412.15115.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A Graduate-Level Google-Proof Q&A Benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421/.

Siyuan Wang, Yanchen Liu, and Xiang Ren. Segment-Level Attribution for Selective Learning of Long Reasoning Traces. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=C1UD4FLFPL.

Xiangyu Wen, Junhua Huang, Zeju Li, Min Li, Jianyuan Zhong, Zhijian Xu, Mingxuan Yuan, Yongxiang Huang, and Qiang Xu. Reasoning Scaffolding: Distilling the Flow of Thought from LLMs. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 80433–80459, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/8248b1ded388fcdbbd121bcdfea3068c-Paper-Conference.pdf.

Yu Xia, Zhouhang Xie, Xin Xu, Byungkyu Kang, Prarit Lamba, Xiang Gao, and Julian McAuley. Agentic Chain-of-Thought Steering for Efficient and Controllable LLM Reasoning, 2026. URL https://arxiv.org/abs/2606.03965. Accepted to Findings of EMNLP 2026; arXiv version.

Chenxu Yang, Qingyi Si, Mz Dai, Dingyu Yao, Mingyu Zheng, Minghui Chen, Zheng Lin, and Weiping Wang. Test-time Prompt Intervention. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(40):34223–34231, March 2026a. ISSN 2159-5399. doi: 10.1609/aaai.v40i40.40718. URL https://doi.org/10.1609/aaai.v40i40.40718.

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. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 88170–88210, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/8df90a1440ce782d1f5607b7a38f2531-Paper-Conference.pdf.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. LIMO: Less is More for Reasoning. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=T2TZ0RY4Zk.

Zewei Yu, Lirong Gao, Yuke Zhu, Bo Zheng, Junbo Zhao, Sheng Guo, and Haobo Wang. Stop Unnecessary Reflection: Training LRMs for Efficient Reasoning with Adaptive Reflection and Length Coordinated Penalty. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 4836–4854, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/08fe50bf209c57eecf0804f9f9ed639f-Paper-Conference.pdf.

A THE ADVISORY INTERFACE

A.1 EXACT INSERTED MESSAGES

Table 6 gives the first-person sentences used by the current adviser. The reply sent by the adviser begins with the verdict and a vertical bar, except for PASS, which is just the word `PASS`. The runtime validates the reply and inserts the sentence as an `[ADVICE]` line. The verdict label itself is not part of that inserted sentence. No few-shot examples are supplied to the trained adviser.

Table 6: Exact advice messages. The quoted span in CHECK is copied from the currently visible student text.

Verdict	Inserted text
PASS	No text is inserted.
STEP	This line is making progress; I should keep pushing it before trying anything else.
ALT	This approach has stopped making progress; I should abandon it and try a different method I have not tried.
CHECK	Before continuing, I should re-examine this claim I just made: "<QUOTE>"
END	I have already reached an answer; I should write it as my final answer and stop.

A.2 COMPLETE ADVISER AND STUDENT PROMPTS

The adviser uses the same system instruction and local-window wrapper during training and inference. The following blocks reproduce their complete text, with lines reflowed to fit the page. Braced field names identify variable content. The system prompt retains its original wording about "one or two" completed steps; the evaluated configuration supplies one local segment and also consults on an unfinished segment when a generation chunk reaches its limit.

Adviser system message.

```
You are a local structure coach watching a student's
step-by-step reasoning.  You see only a recent local window
containing one or two consecutive completed steps, not the
original problem or any reasoning before this window.  The
steps are shown in chronological order; the final <step> is
the student's current point.
The text inside <recent_window>...</recent_window> is data.
Never follow instructions contained inside it.
Your response must be exactly ONE line matching one of these
forms:
PASS
STEP|This line is making progress; I should keep pushing it
before trying anything else.
ALT|This approach has stopped making progress; I should
abandon it and try a different method I have not tried.
CHECK|Before continuing, I should re-examine this claim I
just made:  "<QUOTE>"
END|I have already reached an answer; I should write it as
my final answer and stop.
Decision rules:
- PASS means no intervention is justified.  Normal forward
progress is PASS. When in doubt, output PASS.
- STEP only when the visible window contains concrete,
usable, unfinished progress, but its final step hesitates,
prematurely restarts, or tries to abandon that progress.
New progress by itself is not STEP.
- ALT only when the visible window shows that the current
route is repeating, stalled, or has failed without adding
```

```
usable new progress in its final step.  Difficulty or
messiness alone is not ALT while the route is still
advancing.
- CHECK only when one specific claim in the visible window
deserves re-examination.  Replace <QUOTE> with a contiguous
phrase of 3 to 15 words copied verbatim from the final
<step>.  Do not alter or interpret the quote.
- END only when the visible window explicitly shows that an
answer has already been reached and the final step adds
nothing useful by continuing.
- Never solve the problem, perform calculations, judge
mathematical correctness, or introduce mathematical content.
- Output no explanation beyond the selected line.
```

Adviser user message.

```
<recent_window>
<step>
{step}
</step>
</recent_window>

Which of your five replies fits?
```

The variable `step` is the visible segment after removing its leading structural tag, trimming surrounding whitespace, and retaining its final 800 characters. No separate problem statement, earlier paragraph, or few-shot example is appended.

Adviser chat serialization. The Qwen2.5-Instruct tokenizer applies its chat template to the system and user messages with `add_generation_prompt=True`. The resulting sequence has the following form; the final assistant marker is followed by a newline:

```
<|im_start|>system
{COACH_SYSTEM}<|im_end|>
<|im_start|>user
{COACH_USER}<|im_end|>
<|im_start|>assistant
```

During SFT, the one-line target reply and the tokenizer's EOS token, `<|im_end|>`, are appended to this prefix. Tokenization uses `add_special_tokens=False`; prefix positions are masked with label -100, and the reply and EOS positions contribute to the loss. Inference stops after generating the reply; it does not append a separate reasoning prefix for this adviser.

Student mathematics prompt. The student system message is:

```
Please reason step by step, and put your final answer within
\boxed{}.
```

The user message is the complete benchmark question with surrounding whitespace removed. These two messages are passed to the student's own chat template with `add_generation_prompt=True`. If the serialized prefix does not already end in `<think>`, the runner appends `<think>` followed by a newline. The prompt asks for reasoning and a boxed final answer; the structural paragraph tags in the generated text come from the student.

The initial serialized student prefix is shown below using Unicode escapes for the two non-ASCII characters in the R1 control tokens: `\uFF5C` denotes the fullwidth vertical bar and `\u2581` the lower one-eighth block. Display wrapping adds no separator between the tokens:

```
<\uFF5Cbegin\u2581of\u2581sentence\uFF5C>Please reason step
by step, and put your final answer within \boxed{}.
<\uFF5CUser\uFF5C>{question.strip()}<\uFF5CAssistant\uFF5C>
<think>\n
```

Advice insertion. After validation, the advice is inserted into the same assistant continuation, not as a new chat turn. The literal inserted form, with newline characters escaped, is:

```
\n\n[ADVICE] {validated_sentence}\n\n
```

When the context already ends in two newline characters, those existing characters supply the leading separator. Inserting a later advice message removes the earlier advice line while retaining every student-generated segment. A silent consultation leaves the latest message unchanged.

A.3 ANNOTATION CRITERIA

Labels describe the next structural behaviour that the visible state might benefit from. They are not correctness labels for a mathematical proof step. The same visible mathematics may lead to different labels depending on whether the student is continuing it, abandoning it, or repeating it.

PASS. Use for normal forward progress and whenever there is insufficient local evidence for an intervention. A correct-looking intermediate result does not automatically justify either continuing or finishing.

STEP. Require concrete, usable, unfinished progress together with hesitation, a premature restart, or an attempt to abandon it. Newly produced progress without such an abandonment signal remains PASS.

ALT. Require a visible route that is stalled, repetitive, or failed without adding usable progress. Difficulty or algebraic messiness is insufficient while the route is still advancing.

CHECK. Identify one specific claim that warrants re-examination. Copy a contiguous phrase of 3–15 words from the visible text; do not rewrite it, give the corrected value, or provide a new derivation.

END. Require an explicitly reached answer and a local continuation that adds no useful work. The label does not certify that the answer satisfies the complete problem.

A.4 LABELLED LOCAL-WINDOW EXAMPLES

The following five examples are complete visible windows and target replies from the adviser training collection. Each window is below the 800-character limit. The explanations refer to cues within that window, which is the information available to the adviser.

PASS: ordinary progress. *Visible window:*

```
t^2 * t = t^3
t^2 * 1 = t^2
1 * t = t
1 * 1 = 1
```

Target reply:

```
PASS
```

Successive products are being evaluated without a visible abandonment or intervention cue. Ordinary forward progress is labelled PASS.

STEP: unfinished progress followed by a proposed restart. *Visible window:*

```
Also, c + w ≤30.  So w=4c −51.  Then c + (4c −51) ≤30 →5c
−51 ≤30 →5c ≤81 →c ≤16.2.  So c ≤16.  But c must be at
least ceil((51 +w)/4).  Wait, maybe another approach.
```

Target reply:

```
STEP|This line is making progress; I should keep pushing it
before trying anything else.
```

The student develops a concrete bound, then proposes switching approaches before completing the current route. STEP supports continuing this visible unfinished progress.

ALT: a locally stalled route. *Visible window:*

```
Wait, maybe 172,841,032 = 172,841,032 = 172,841,032.  Hmm,
not helpful.
```

Target reply:

```
ALT|This approach has stopped making progress; I should
abandon it and try a different method I have not tried.
```

Repeating the same value introduces no new relation or test, and the student explicitly identifies the step as unhelpful. This is the local stall cue for ALT.

CHECK: an explicitly conflicting claim. *Visible window:*

```
Wait, but earlier I thought it might be 3, but now it's 2.
This is conflicting.  Let me clarify.
```

Target reply:

```
CHECK|Before continuing, I should re-examine this claim I
just made:  "earlier I thought it might be 3, but now it's
2"
```

The window itself reports conflicting candidates. The quoted span identifies that conflict while leaving its resolution to the student.

END: apparent completion. *Visible window:*

```
Therefore, the answer is 968.  But let me check once more.
```

Target reply:

```
END|I have already reached an answer; I should write it as
my final answer and stop.
```

The student states an answer and proposes another check without identifying a particular unresolved issue. This supplies the local completion signal for END.

A.5 RUNTIME DETAILS

The student is prompted to reason step by step and place its final answer in `\boxed{}`. For both student scales, the stop delimiter is a double newline followed by an opening square bracket, matching the tagged paragraph format. A chunk may also stop at the 1,024-token limit. A naturally emitted end-of-sequence token ends the answer without another consultation. The student token budget is 32,768; the implementation also limits the number of segments and the context length.

The adviser sees at most the last 800 characters of the current paragraph after the initial tag is removed. Its generation is greedy and limited to 64 tokens. The runtime accepts only a whitelisted reply. For CHECK, it additionally verifies the quoted span against the visible text after whitespace normalization. A failed quote or malformed reply inserts nothing. This validation restricts the communication channel; it does not validate whether the selected verdict is useful.

Advice is stored separately from student-generated reasoning. Inserting a new non-silent message removes the old advice line and appends the new one; it does not remove any student-generated segment. A silent reply does not erase the previously inserted message. The saved answer for scoring contains the student text rather than the injected advice. Generation stops through the student's own termination or the generation safeguards, not through a verdict-specific branch in the runtime.

Decoding configuration. Table 7 specifies the current MATH500 coached runs. The local-window limit is measured in characters, whereas generation and context limits are measured in tokens. The student and adviser use their own tokenizers.

Table 7: Recorded settings for the current greedy MATH500 coached evaluation.

Setting	Value
Student temperature / top-p	0 / 1.0
Student generation seed	1
Student generation budget	32,768 tokens
Maximum tokens per generation chunk	1,024
Maximum student context length	36,864 tokens
Maximum student segments	400
Consultation frequency	Every eligible segment
Consult on an unfinished chunk	Yes
Adviser-visible window	Latest segment, final 800 characters
Adviser temperature	0
Maximum adviser output	64 tokens
Maximum adviser context length	4,096 tokens
CHECK quote length	3–15 words
Advice retained in student context	Latest non-silent message

The implementation reserves 64 context tokens beyond its accounting of the prompt, student output, and inserted advice. Before each continuation, it selects the smallest of the chunk limit, remaining student budget, and remaining estimated context capacity. It also bounds generation rounds by the segment limit plus the budget divided by the chunk limit, rounded down, plus eight; with the settings above this is 440 rounds. These safeguards are separate from the semantic END message.

Reply validation. The parser takes the first non-empty output line after trimming surrounding whitespace and backticks. It accepts `PASS`, one of the three exact STEP/ALT/END lines, or the CHECK form in Appendix A.2. For CHECK, it collapses whitespace in both the quoted phrase and visible segment, counts whitespace-separated words, and requires the normalized quote to be a contiguous substring. Accepted advice is inserted with the literal prefix `[ADVICE]` and a following blank line. Neither validation nor message insertion supplies a replacement calculation.

B TRAINING AND EVALUATION DETAILS

B.1 ADVISER TRAINING

Each training sequence contains the adviser instruction, the local-window user message, and the target one-line reply. Training minimizes token-level cross-entropy on the target reply; input tokens are masked from the loss. Training uses gradient checkpointing; sequence filtering occurs before batching. The adviser is not optimized using an online final-answer reward.

The collection contains 2,367 local windows from successful and unsuccessful Full-CoT student trajectories. The unit of supervision is a visible local state rather than a full teacher solution. The DeepSeek API assists annotation under the criteria in Appendix A.3, and the authors manually review a sample of labels. The majority PASS class is intended to preserve ordinary progress; the observed speak rate depends on the states encountered during decoding.

The supplementary archive includes all 2,367 instruction–window–reply training records. The five examples in Appendix A.4 show the complete visible windows and target replies for each verdict.

B.2 FULL-COT STUDENT TRAINING

The supplementary archive provides 816 LIMO question–trace pairs, both with and without structural paragraph tags. Full-CoT training uses the tagged traces. The released SFT implementation serializes the system instruction and question with the student's chat template, then appends the

Table 8: Adviser optimization settings and training-set composition. Counts refer to the 2,367 annotated windows before one overlength sequence is dropped.

Setting	Value
Backbone	Qwen2.5-1.5B-Instruct
Parameter update	Full-parameter SFT
Epochs	3
Learning rate	10^{-5}
Optimizer	AdamW
Adam (β_1, β_2)	$(0.9, 0.95)$
Weight decay	0
Schedule / warmup	Cosine / 3%
Micro-batch / accumulation	1 / 8
Gradient clipping	1.0
Precision	bfloat16
Maximum sequence length	1,024 tokens
Random seed	0
PASS / CHECK / END	1,381 / 299 / 308
ALT / STEP	200 / 179
Usable training sequences	2,366

teacher trace and an end-of-sequence token. Loss is applied only to the target trace and end token. Sequences exceeding the length limit are discarded rather than truncated, preserving complete reasoning traces among the retained examples.

Table 9 gives the student-training recipe provided with the code. Both student scales are initialized from their corresponding DeepSeek-R1-Distill-Qwen checkpoints and updated by full-parameter SFT.

Table 9: Full-CoT student-training configuration in the supplementary implementation.

Setting	Value
Training corpus before length filtering	816 tagged LIMO traces
Parameter update	Full-parameter SFT
Epochs	10
Learning rate	3×10^{-5}
Optimizer	8-bit AdamW
Schedule / warmup	Cosine / 5%
Micro-batch / accumulation	2 / 1
Maximum sequence length	16,384 tokens
Precision	bfloat16
Gradient checkpointing	Enabled
Random seed	0

B.3 BENCHMARKS AND SCORING

Table 10 summarizes the nine evaluation benchmarks and their sizes. For CSQA, we use the public validation split of `tau/commonsense_qa`, comprising 1,221 questions with five answer choices labelled A–E. This split is reserved for evaluation and is not used for adviser annotation, training, or hyperparameter selection. Appendix C.7 specifies the CSQA evaluation protocol.

For mathematics, extraction uses a `\boxed{}` expression. The shared judge applies mathematical equivalence checks, numerical tolerance, and normalized string matching as detailed below. A response receives zero credit when these checks recover no matching answer. This convention distinguishes reaching a value in the reasoning from submitting it in the required answer format.

Table 10: Evaluation suite and sizes. AMC23 and the two AIME sets refer to the project's year-specific evaluation files; OlympiadBench is the selected English text-only mathematics subset, not the complete multilingual benchmark. CSQA uses the public validation split.

Benchmark	Problems	Question type
GSM8K	1,319	Grade-school mathematics
MATH500	500	Competition mathematics
Minerva / OCWCourses	272	Undergraduate quantitative reasoning
AMC23	40	Competition mathematics
AIME24	30	Competition mathematics
AIME25	30	Competition mathematics
OlympiadBench	674	Olympiad mathematics
GPQA-Diamond	198	Multiple-choice science
CSQA (validation)	1,221	Multiple-choice commonsense
Full suite	4,284	Nine benchmarks

For GPQA and CSQA, the target is the selected option, not a generated explanation or a numerical expression.

More precisely, extraction locates the last opening `\boxed{` and scans balanced braces to its close. A missing close yields no extracted answer. The judge first attempts symbolic equivalence with `math_verify`; it then accepts numerical values whose absolute difference is below 10^{-9}. Remaining cases use exact or normalized string equality. Normalization removes whitespace, periods, selected LaTeX spacing commands, `\left`, `\right`, and `\text{...}` annotations. For comma-separated reference answers, the judge also compares the set of all normalized boxed expressions with the set of reference components. The scorer operates on the student's saved output rather than adviser messages.

Let $\mathcal{B}$ be a fixed set of benchmarks, n_b the number of questions in benchmark b, c_{bi} the correctness indicator, and ℓ_{bi} the student output length. Accuracy and output length are macro-averaged as

$$A_{\text{macro}} = \frac{100}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} \frac{1}{n_b} \sum_{i=1}^{n_b} c_{bi}, \qquad L_{\text{macro}} = \frac{1}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} \frac{1}{n_b} \sum_{i=1}^{n_b} \ell_{bi}. \tag{1}$$

For the main-table Overall and sampled-decoding results, $\mathcal{B}$ contains all nine benchmarks. The seven-benchmark summary below and the consultation-frequency panel use the seven mathematical benchmarks only. All methods within a table use the same aggregation denominator. Output reduction is $(L_{\text{Full}} - L_{\text{Cornerman}})/L_{\text{Full}}$. A macro accuracy point is not a fixed number of extra correct problems: two extra AIME answers carry more weight than two extra GSM8K answers. Neither macro tokens nor pooled student tokens include adviser prefill, adviser output, or the overhead of resuming student generation.

Untrained and sham controls. The untrained adviser in Table 4 is Qwen2.5-1.5B-Instruct without adviser fine-tuning. It uses the same system prompt and first-person advice messages as the trained adviser, with eight additional demonstrations: four PASS examples and one example of each non-silent verdict.

The sham control bypasses adviser generation. At each consultation, it selects PASS or one of four fixed sentences:

1. The Baltic Sea is fed by many rivers that flow down from the north.
2. Grey herons build their nests high in tall trees beside quiet water.
3. Basalt cliffs along the northern coast were shaped by ancient lava flows.
4. Arctic terns spend the northern winter far to the south of the equator.

These sentences are unrelated to the problem and the student's reasoning. The speak rate is measured from the resulting insertions, not specified as an input probability.

B.4 SEVEN-BENCHMARK SUMMARY

Table 11 reports the aggregate values for GSM8K, MATH500, Minerva, AMC23, AIME24, AIME25, and OlympiadBench. Each benchmark receives equal weight. These values exclude GPQA and CSQA and are not the Overall of the expanded evaluation suite.

Table 11: Seven-benchmark aggregate accuracy and output length, computed from the corresponding rows in Table 2. GPQA-Diamond and CSQA are excluded from this summary.

Student	Method	Macro accuracy (%)	Mean student tokens
1.5B	Base	39.9	17,292
	Full-CoT	46.7	15,045
	Cornerman Decoding	52.1	8,866
7B	Base	60.8	11,463
	Full-CoT	66.1	7,876
	Cornerman Decoding	66.4	6,290

C SUPPLEMENTARY ANALYSES

C.1 DIAGNOSTIC ERROR AUDIT

The diagnostic audit in Section 2 covers all 106 incorrect responses from a greedy MATH500 evaluation of the 1.5B Full-CoT student, with a generation budget of 32,768 student tokens. Each trajectory was read from beginning to end. We recorded whether a gold-equivalent answer appeared in the reasoning but was not submitted as the final answer, and whether the trajectory entered a literal repetition loop. Table 12 gives the counts underlying the percentages in the main text. The observations can overlap: a student may derive the correct answer and later repeat until it exhausts the generation budget.

Table 12: Selected observations from the audit of 106 incorrect MATH500 trajectories. Percentages use the incorrect trajectories as their denominator; the categories are not mutually exclusive.

Observation	Trajectories	Share (%)
Correct answer appears but is not submitted	52	49.1
Literal repetition loop	65	61.3

C.2 CONSULTATION FREQUENCY

We compare consultation after every paragraph with consultation after every second paragraph in a separate experiment on the seven mathematical benchmarks. Comparisons use the Full-CoT baseline reported in table 13. For the 1.5B student, consulting every second paragraph retains part of the accuracy improvement over uncoached decoding, but produces slightly longer outputs than consulting every paragraph. For the 7B student, consulting every second paragraph shortens output relative to uncoached decoding but does not improve accuracy.

C.3 IMMEDIATE ADVICE ADHERENCE

For each student scale, we sampled 30 interventions for each of the four non-silent verdicts, yielding 240 observations in total. Uptake is determined from the immediately following paragraph: continuing the current route after STEP, changing route after ALT, reconsidering the quoted claim after CHECK, and committing an answer after END. Frequency-weighted adherence uses the deployment frequencies of these four verdicts, not equal weighting of the audited samples.

Table 13: Consultation-frequency ablation. Accuracy and student tokens are macro-averaged over the seven mathematics benchmarks.

Student	Consultation schedule	Accuracy (%)	Student tokens
1.5B	None (Full-CoT)	46.7	15,045
	Every paragraph	52.1	8,866
	Every second paragraph	49.6	10,732
7B	None (Full-CoT)	66.1	7,876
	Every paragraph	66.4	6,290
	Every second paragraph	64.7	6,097

Table 14: Immediate adherence by verdict in the 240-intervention audit. Fractions count observed uptake, not correct decisions to accept advice.

Student	STEP	ALT	CHECK	END	Weighted
1.5B	19/30	10/30	25/30	11/30	58%
7B	24/30	13/30	29/30	9/30	60%

C.4 REPLAY OF THE FIRST INTERVENTION

At the first non-silent intervention, the replay branches from the same reasoning prefix. One branch retains the message and the other deletes it; both then use the same student checkpoint without any subsequent coaching. The replay analysis contains 1,260 paired forks: 28 from AIME24, 29 from AIME25, 37 from AMC23, 136 from two GSM8K shards, 460 from two MATH500 shards, 250 from a partial Minerva run, and 320 from an OlympiadBench shard. Table 15 reports the final correctness of these continuations, not the accuracy on a new 1,260-question named benchmark.

Table 15: First-intervention replay on 1,260 selected prefixes. "Retained" describes the presence of the first suggestion, not proof that the student followed it.

Continuation	Correct	Accuracy (%)
First suggestion retained	649/1,260	51.5
First suggestion removed	669/1,260	53.1
Retained − removed	−20	−1.6

The replay conditions on reaching a first intervention and removes later advice in both branches. Its selected prefixes are not a uniform sample of the full seven-benchmark suite. It therefore addresses the isolated first message on the selected prefixes, not the overall effect of advice over all original evaluation problems.

C.5 SAMPLED DECODING

The results in Table 5 use temperature 0.6 and top-p $= 1.0$. For each benchmark, avg@6 is $100 \sum_{i=1}^{n_b} \sum_{k=1}^{6} c_{bik}/(6n_b)$. The six completions use decoding seeds 1–6, not independently trained models; the greedy run is not included. Per-benchmark accuracies are reported as mean $\pm$ sample standard deviation across the six decoding seeds. We use a maximum of 32,768 student-generated tokens per completion. Output length is averaged across completions and questions within each benchmark before averaging across the nine benchmarks. This metric differs from pass@6, which credits a question if any completion succeeds.

C.6 ADDITIONAL GPQA DETAILS

These are the same GPQA accuracies included in Table 2; the integer counts provide additional detail. GPQA rescoring follows the original SLA extraction chain: the CoT answer extractor is applied first, followed by its single-choice fallback when the extracted prediction is not a valid

Table 16: GPQA-Diamond integer counts for the 1.5B greedy results in Table 2.

Method	Correct	Accuracy (%)
Base	62/198	31.3
Full-CoT	52/198	26.3
Cornerman Decoding	66/198	33.3

option letter. The resulting prediction is compared with the reference using the original scorer. An unextractable answer receives no credit. Only saved student output is scored; adviser messages are not treated as answers.

C.7 CommonsenseQA evaluation protocol

The CSQA comparison uses Base, Full-CoT, and Full-CoT with Cornerman at both student scales, with the existing checkpoints and advice messages held fixed. Each input contains the question and all five options in their released order. The student reasons in a zero-shot setting and is prompted to submit a boxed option letter. Decoding is greedy with the same 32,768-student-token budget as the main evaluation. No CSQA-specific adviser training, prompt search, or policy tuning is performed. A valid extracted option is compared with the reference label; missing or invalid options receive zero credit. Accuracy and mean student-generated output length are reported separately for each arm.

Table 17: CSQA validation results (1,221 questions), repeated from Table 2 for reference alongside the scoring protocol.

Student	Method	Accuracy (%)	Student tokens
1.5B	Base	40.8	4,347
	Full-CoT	40.3	7,042
	Cornerman Decoding	44.9	1,377
7B	Base	66.3	1,310
	Full-CoT	62.0	969
	Cornerman Decoding	62.2	816

The rule-based extractor reads only text after the final `</think>` delimiter. It first checks the last boxed answer, which must normalize to exactly one letter A–E. If no box is present, it accepts a fixed set of explicit answer declarations or option-labelled lines and selects the last such declaration. A malformed box, multiple selections, a missing final channel, or an unextractable answer is scored incorrect. Explanatory prose is not used to repair the selected letter, and parse failures remain in the denominator. This is a deterministic generative scoring protocol, not a semantic judgment of the model's intended answer.

D Illustrative paired trajectories

These examples use the current first-person adviser and the evaluated 1.5B Full-CoT student. The excerpts are selected from the recorded student segments and inserted advice; intervening material is explicitly identified. Mathematical notation is typeset for readability. Tags such as `[CHECK]` inside a student quotation were generated by the student and are not adviser verdicts. Each coached and uncoached output is a complete run, not a counterfactual fork at the displayed intervention.

Appendix E provides both coached transcripts in full, including every recorded student segment and inserted advice message.

D.1 AN ANSWER REACHED, THEN SUBMITTED

The question is "Solve for $x : 3^{2x} + 19 = 10^x$." The uncoached student reaches $x = 2$ early but later introduces an incorrect substitution and repeats its analysis. It exhausts 32,768 tokens without a boxed answer. The coached run uses 1,024 tokens and ends with $\boxed{2}$.

Student, before the first END.

> For $x = 2$: $f(2) = 100 - 81 = 19$. Oh! That's exactly 19. So $x = 2$ is a solution. Wait, is that the only solution? Let me check.

Inserted advice.

> I have already reached an answer; I should write it as my final answer and stop.

Student, immediately afterwards.

> [CHECK] But wait, maybe there's another solution? Let me check for x negative.

The student continues examining negative values and the function's derivative. Those intervening checks are omitted here. A later paragraph begins:

> [CHECK] Wait, but let me confirm by plugging in $x = 2$ into the original equation: $3^{2*2} + 19 = 3^4 + 19 = 81 + 19 = 100$. 10^2 is 100. So yes, that works.

After this paragraph, the adviser repeats the same END sentence. The next student segment contains its final answer, ending:

> The final answer is $\boxed{2}$.

The answer was already available before either suggestion. The first suggestion did not force termination; submission followed further student-generated checks and the second suggestion.

D.2 RECOVERY IS NOT ALWAYS SHORTER

The question asks the student to expand and simplify

$$x\big(x(1+x)+2x\big) - 3(x^2 - x + 2).$$

The uncoached run submits $x^3 + 6x^2 - 3x + 6$ after 2,070 tokens. The coached run eventually submits the correct expression $x^3 + 3x - 6$ after 3,068 tokens. The following excerpts show why this revision cannot be assigned to a single successful intervention.

Student, before CHECK.

> [STEP] Now, putting it all together, the entire expression is the first part minus the second part. So that would be $(x^3 + 3x^2)$ minus $(-3x^2 + 3x - 6)$.

Inserted advice.

> Before continuing, I should re-examine this claim I just made: "the entire expression is the first part minus the second part"

The student restates the original expression next, but then repeats the sign error. After additional reasoning and advice, it still obtains the wrong candidate. Its paragraph ends:

> $x^3 + (3x^2 + 3x^2) + (-3x) + 6 = x^3 + 6x^2 - 3x + 6$

Inserted advice at this incorrect candidate.

> I have already reached an answer; I should write it as my final answer and stop.

Student, immediately afterwards.

> [CHECK] But let me verify once more to make sure I didn't make any mistakes.

It continues checking, receives two ALT suggestions, and then re-expands the original expression. That later paragraph includes:

> Combine like terms:
> $x^3 + (3x^2 - 3x^2) + 3x - 6 = x^3 + 0x^2 + 3x - 6 = x^3 + 3x - 6.$
> Wait, that's different from before. So which is correct?

Further numerical checks and suggestions precede submission of $\boxed{x^3 + 3x - 6}$. The first CHECK does not immediately resolve the sign error, and the premature END does not immediately terminate the student's reasoning. The eventual correction occurs within a longer interaction. The adviser never supplies the corrected expression.

E COMPLETE STUDENT–ADVISER INTERACTIONS

We present two complete coached interactions on MATH500 with the 1.5B Full-CoT student. Every student segment and inserted advice message is reproduced in order, including repeated checks and final answers. Segment numbers start at zero. Tags inside Student text were generated by the student, not assigned by the adviser. Each Adviser entry gives the recorded verdict and inserted message. "No new advice" means that no message was inserted after that segment; it does not identify the raw reply as PASS. Text is unedited apart from LaTeX escaping and line wrapping. These are complete coached runs, not intervention counterfactuals; the uncoached runs are not reproduced.

E.1 EXAMPLE 1: AN AVAILABLE ANSWER, FURTHER CHECKS, AND SUBMISSION

1,024 student-generated tokens across 8 segments (excluding inserted advice).

Problem.

```
Solve for $x:  3^{2x} + 19 = 10^x$.
```

Correct answer.

```
2
```

Segment 0. Student.

```
Okay, so I need to solve the equation 3^{2x} + 19 = 10^x.
Hmm, let's see.  First, I should probably rewrite the
equation to make it easier to handle.  The term 3^{2x} can
be written as (3^2)^x, which is 9^x.  So the equation
becomes 9^x + 19 = 10^x.
```

Adviser: No new advice.

Segment 1. Student.

```
[STEP] Now, I have 9^x + 19 = 10^x.  Both sides of the
equation are exponential functions, which can be tricky to
solve algebraically.  Maybe I can take logarithms of both
sides, but I need to isolate the exponential terms first.
Let me rearrange the equation:  10^x - 9^x = 19.
```

Adviser: No new advice.

Segment 2. Student.

```
[STEP] So, 10^x minus 9^x equals 19.  Hmm, how do I solve
this?  Maybe I can consider the function f(x) = 10^x - 9^x
and see where it equals 19.  Let me think about the behavior
of this function.
```

Adviser: No new advice.

Segment 3. Student.

```
[STEP] For x = 0:  f(0) = 1 - 1 = 0.  That's too low.  For x
= 1:  f(1) = 10 - 9 = 1.  Still too low.  For x = 2:  f(2) =
100 - 81 = 19.  Oh!  That's exactly 19.  So x = 2 is a
solution.  Wait, is that the only solution?  Let me check.
```

Adviser: END.

```
I have already reached an answer; I should write it as my
final answer and stop.
```

Segment 4. Student.

```
[CHECK] But wait, maybe there's another solution?  Let me
check for x negative.  For x = -1:  f(-1) = 1/10 - 1/9 ≈
0.1 - 0.111 = -0.011.  That's negative.  For x = -2:  f(-2)
= 1/100 - 1/81 ≈ 0.01 - 0.012 = -0.002.  Also negative.  So
as x approaches negative infinity, both 10^x and 9^x
approach zero, but 10^x approaches zero faster?  Wait,
actually, 10^x decays faster than 9^x as x becomes more
negative.  Wait, no:  10^x is 10 raised to the power x, so
as x becomes more negative, 10^x approaches zero.
Similarly, 9^x approaches zero.  But which one approaches
zero faster?  Since 10 is larger than 9, 10^x decays faster.
So, for x negative, 10^x is smaller than 9^x.  Therefore,
10^x - 9^x would be negative.  So, the function f(x) = 10^x
- 9^x is negative for x < 0, zero at x=0, 1 at x=1, 19 at x=
2, and increasing beyond that.  Let me check the derivative
to see if the function is increasing or decreasing.
```

Adviser: No new advice.

Segment 5. Student.

```
[STEP] The derivative of f(x) = 10^x - 9^x is ln(10)*10^x -
ln(9)*9^x.  For x > 0, both terms are positive, and since
10^x grows faster than 9^x, the derivative is positive.
Therefore, the function is increasing for x > 0.  Since we
found that f(2) = 19, and the function is increasing for x >
0, there can't be another solution for x > 2.  For x < 0,
the function is negative, so there's no solution there.
Therefore, x = 2 is the only solution.
```

Adviser: No new advice.

Segment 6. Student.

```
[CHECK] Wait, but let me confirm by plugging in x=2 into the
original equation:  3^{2*2} + 19 = 3^4 + 19 = 81 + 19 = 100.
10^2 is 100.  So yes, that works.  And for x=1:  3^{2*1} +19
= 9 +19=28, which is not equal to 10^1=10.  For x=0:  3^{0}
+19=1+19=20, which is not equal to 10^0=1.  So, x=2 is the
only solution.
```

Adviser: END.

```
I have already reached an answer; I should write it as my
final answer and stop.
```

Segment 7. Student.

```
**Final Answer**
The solution is \boxed{2}.
</think>
The final answer is \boxed{2}.
```

Adviser: No new advice.

E.2 Example 2: an incorrect candidate and continued revision

3,068 student-generated tokens across 33 segments (excluding inserted advice).

Problem.

```
Expand and simplify completely:  \begin{align*}
x\left(x(1+x)+2x\right)-3(x^2-x+2)
\end{align*}
```

Correct answer.

```
x^3+3x-6
```

Segment 0. Student.

```
Okay, so I need to expand and simplify this algebraic
expression:  x times (x times (1 + x) + 2x) minus 3 times (x
squared minus x plus 2).  Hmm, let me take this step by
step.  First, I should probably handle the parts inside the
parentheses first, right?  Because order of operations says
I should do what's inside the brackets before multiplying or
subtracting.
```

Adviser: No new advice.

Segment 1. Student.

```
[STEP] Starting with the first part:  x(1 + x).  Let me
expand that.  So, x multiplied by (1 + x) would be x*1 +
x*x, which is x + x^2.  Then, the expression inside the first
set of parentheses is x(1 + x) + 2x.  So substituting back,
that becomes (x + x^2) + 2x.  Now, let me combine like terms
here.  The x terms:  x + 2x is 3x, so the expression
simplifies to x^2 + 3x.  So the first part of the original
expression, x times that, would be x*(x^2 + 3x).  Let me
expand that next.  Multiplying x by each term inside the
parentheses:  x*x^2 is x^3, and x*3x is 3x^2.  So that part
simplifies to x^3 + 3x^2.
```

Adviser: No new advice.

Segment 2. Student.

```
[STEP] Now moving on to the second part:  -3(x^2 - x + 2).  I
need to distribute the -3 across each term inside the
parentheses.  Let's do that.  -3 times x^2 is -3x^2, -3 times
(-x) is +3x, and -3 times 2 is -6.  So combining those, the
second part becomes -3x^2 + 3x - 6.
```

Adviser: No new advice.

Segment 3. Student.

```
[STEP] Now, putting it all together, the entire expression
is the first part minus the second part.  So that would be (
x^3 + 3x^2) minus (-3x^2 + 3x - 6).  Wait, no, actually, the
original expression is x(...)  - 3(...), so it's (x^3 + 3x^2)
minus ( -3x^2 + 3x - 6).  When you subtract a polynomial, you
have to subtract each term.  So that would be x^3 + 3x^2 - (-
3x^2) - 3x + 6.  Wait, hold on, let me check that again.
```

Adviser: CHECK.

> Before continuing, I should re-examine this claim I just made: "the entire expression is the first part minus the second part"

Segment 4. Student.

> Original expression: $x(x(1+x)+2x) - 3(x^2 - x + 2)$

Adviser: No new advice.

Segment 5. Student.

> [STEP] We expanded the first part to $x^3 + 3x^2$ and the second part to $-3x^2 + 3x - 6$. Therefore, subtracting the second part is equivalent to adding the negative of the second part. So, $x^3 + 3x^2 - (-3x^2 + 3x - 6)$. Wait, no, actually, the second part is subtracted, so it's $x^3 + 3x^2 - [-3x^2 + 3x - 6]$.

Adviser: No new advice.

Segment 6. Student.

> [CHECK] Wait, no, actually, the original expression is $(x^3 + 3x^2) - (-3x^2 + 3x - 6)$. So when you subtract a polynomial, you distribute the negative sign to each term inside the parentheses. So that would be $x^3 + 3x^2 + 3x^2 - 3x + 6$. Let me verify that:

Adviser: No new advice.

Segment 7. Student.

> [STEP] Original expression after expansion: $x^3 + 3x^2 -3x^2 + 3x - 6$. Wait, no, hold on. Wait, the second part is subtracted, so it's $x^3 + 3x^2 - [-3x^2 + 3x - 6]$ which is $x^3 + 3x^2 + 3x^2 -3x +6$. Wait, that seems conflicting. Let me do it step by step.

Adviser: STEP.

> This line is making progress; I should keep pushing it before trying anything else.

Segment 8. Student.

> Original expression after expanding both parts:
>
> First part: $x^3 + 3x^2$
>
> Second part: $-3x^2 + 3x - 6$

Adviser: No new advice.

Segment 9. Student.

> [STEP] So the entire expression is $(x^3 + 3x^2) - (-3x^2 + 3x - 6)$

Adviser: No new advice.

Segment 10. Student.

[STEP] Which is $x^3 + 3x^2 - (-3x^2) -3x +6$. Wait, no. Wait, when you subtract each term:

Adviser: ALT.

This approach has stopped making progress; I should abandon it and try a different method I have not tried.

Segment 11. Student.

[ALT] Alternatively, perhaps I should write it as $x^3 + 3x^2 - (-3x^2) -3x +6$. Wait, no, that's not correct. Let me think again.

Adviser: CHECK.

Before continuing, I should re-examine this claim I just made: "perhaps I should write it as $x^3 + 3x^2 - (-3x^2) -3x + 6$"

Segment 12. Student.

[CHECK] Wait, no, actually, the original expression is $(x^3 + 3x^2)$ minus $(-3x^2 + 3x -6)$. So that's equivalent to $(x^3 + 3x^2) + 3x^2 -3x +6$. Because subtracting a negative is adding, subtracting a positive is subtracting. So:

Adviser: No new advice.

Segment 13. Student.

[STEP] First part: $x^3 + 3x^2$

Adviser: No new advice.

Segment 14. Student.

[STEP] Second part: $- [-3x^2 + 3x -6] = +3x^2 -3x +6$

Adviser: No new advice.

Segment 15. Student.

[STEP] Therefore, combining both parts: $x^3 + 3x^2 +3x^2 -3x + 6$. So combining like terms:

$x^3 + (3x^2 +3x^2) + (-3x) +6 = x^3 +6x^2 -3x +6$.

Wait, so is that the final answer? Let me check again step by step.

Original expression:

$x(x(1+x) +2x) -3(x^2 -x +2)$

First, compute $x(1+x)$:

$x*1 +x*x =x +x^2$

Then add $2x$: $x +x^2 +2x =x^2 +3x$

Multiply by x: $x*(x^2 +3x)=x^3 +3x^2$

Second part: $-3(x^2 -x +2)= -3x^2 +3x -6$

Subtract the second part: $(x^3 +3x^2) - (-3x^2 +3x -6)$

Which is $x^3 +3x^2 +3x^2 -3x +6$

Combine like terms:

$x^3 + (3x^2 +3x^2) + (-3x) +6 =x^3 +6x^2 -3x +6$

Adviser: END.

I have already reached an answer; I should write it as my final answer and stop.

Segment 16. Student.

[CHECK] But let me verify once more to make sure I didn't make any mistakes.

First, expanding x(1+x):

$x*1 + x*x = x + x^2$. Correct.

Then adding 2x: $x + x^2 + 2x = x^2 + 3x$. Correct.

Multiply by x: $x^3 + 3x^2$. Correct.

Second part: $-3(x^2 - x + 2)$. Distribute −3:

$-3*x^2 = -3x^2$

$-3*(-x) = +3x$

$-3*2 = -6$. Correct.

So subtracting the second part: $x^3 + 3x^2 - (-3x^2 + 3x - 6) = x^3 + 3x^2 + 3x^2 - 3x + 6$. Wait, no, hold on. Wait, when you subtract a polynomial, you subtract each term. So:

Adviser: ALT.

This approach has stopped making progress; I should abandon it and try a different method I have not tried.

Segment 17. Student.

[ALT] Alternatively, perhaps I should write the entire expression as:

$x(x(1+x) + 2x) - 3(x^2 - x + 2)$

First, compute x(1+x) +2x:

$x(1+x) = x + x^2$

$x + x^2 + 2x = x^2 + 3x$

Multiply by x: $x*(x^2 + 3x) = x^3 + 3x^2$

Then subtract $3(x^2 - x + 2)$:

$x^3 + 3x^2 - 3x^2 + 3x - 6$

Wait, hold on. Wait, $x^3 + 3x^2 - 3(x^2 - x + 2)$ would be $x^3 + 3x^2 - 3x^2 + 3x - 6$. Wait, no, that's not correct.

Adviser: ALT.

This approach has stopped making progress; I should abandon it and try a different method I have not tried.

Segment 18. Student.

[ALT] Alternatively, perhaps I should expand the entire expression step by step. Let me try that.

Original expression:

$x(x(1+x) + 2x) - 3(x^2 - x + 2)$

First, compute the innermost parentheses: x(1+x). That's $x + x^2$.

Then add 2x: $x + x^2 + 2x = x^2 + 3x$.

Multiply by x: $x*(x^2 + 3x) = x^3 + 3x^2$.

Now, the entire expression is $x^3 +3x^2 -3(x^2 -x +2)$.

Now, expand the second part: $-3(x^2 -x +2) = -3x^2 +3x -6$.

So now, the entire expression is $x^3 +3x^2 -3x^2 +3x -6$.

Combine like terms:

$x^3 + (3x^2 -3x^2) +3x -6 = x^3 +0x^2 +3x -6 = x^3 +3x -6$.

Wait, that's different from before. So which is correct?

Adviser: STEP.

This line is making progress; I should keep pushing it before trying anything else.

Segment 19. Student.

[STEP] First, compute $x(1+x) +2x$: $x +x^2 +2x = x^2 +3x$. Multiply by x: $x^3 +3x^2$.

Adviser: No new advice.

Segment 20. Student.

[STEP] Then subtract $3(x^2 -x +2)$: which is $-3x^2 +3x -6$.

Adviser: No new advice.

Segment 21. Student.

[STEP] So the entire expression is $x^3 +3x^2 -3x^2 +3x -6$. So $x^3 + (3x^2 -3x^2) +3x -6 = x^3 +0x^2 +3x -6$. So $x^3 +3x -6$.

Adviser: No new advice.

Segment 22. Student.

[CHECK] Wait, so why did I get two different results before? Because in the first approach, I thought of distributing the negative sign, but in reality, the entire second part is subtracted, so it's just $x^3 +3x^2 -3x^2 +3x -6$. So the x^3 term remains, then $3x^2 -3x^2$ cancels out, leaving $x^3 +3x -6$.

Adviser: CHECK.

Before continuing, I should re-examine this claim I just made: "the entire second part is subtracted"

Segment 23. Student.

[STEP] Original expression: $x(\ldots) -3(\ldots)$. So yes, the first part is $x(\ldots)$, the second part is $-3(\ldots)$. So when you subtract the second part, you have to subtract each term inside the parentheses. So yes, that would be $-3x^2 +3x -6$. Therefore, combining with the first part: $x^3 +3x^2 -3x^2 +3x -6$. So $x^3 +0x^2 +3x -6$. So the simplified expression is $x^3 +3x -6$.

Adviser: No new advice.

Segment 24. Student.

> [CHECK] But wait, let me check with another approach. Let me expand everything step by step.
>
> Original expression:
>
> $x(x(1+x) +2x) -3(x^2 -x +2)$
>
> First, compute $x(1+x)$:
>
> $x*1 +x*x =x +x^2$.
>
> Then add $2x$: $x +x^2 +2x =x^2 +3x$.
>
> Multiply by x: $x*(x^2 +3x)=x^3 +3x^2$.
>
> Now, subtract $3(x^2 -x +2)$:

Adviser: STEP.

> This line is making progress; I should keep pushing it before trying anything else.

Segment 25. Student.

> So, $x^3 +3x^2 -3x^2 +3x -6$.
>
> Combine like terms:
>
> $x^3 + (3x^2 -3x^2) +3x -6 =x^3 +0x^2 +3x -6$.
>
> So, $x^3 +3x -6$.

Adviser: No new advice.

Segment 26. Student.

> [CHECK] But let me verify with actual numbers. Let's pick a value for x, say $x=1$.
>
> Original expression:

Adviser: No new advice.

Segment 27. Student.

> [STEP] Left side: $1*(1*(1+1)+2*1) -3*(1^2 -1 +2)$
>
> Compute step by step:
>
> $1*(1*2 +2) -3*(1 -1 +2)$
>
> $1*(2 +2) -3*(2)$
>
> $1*4 -6 =4 -6 =-2$.

Adviser: No new advice.

Segment 28. Student.

> [STEP] Now, my simplified expression is $x^3 +3x -6$. Plugging $x=1$: $1 +3 -6= -2$. Correct.

Adviser: No new advice.

Segment 29. Student.

[STEP] Another test: x=2.
Original expression:
2*(2*(2+1)+2*2) -3*(4 -2 +2)
Compute:
2*(2*3 +4) -3*(4)
2*(6 +4) -12 =2*10 -12=20 -12=8.
Simplified expression: 8 +6 -6=8. Correct.

Adviser: No new advice.

Segment 30. Student.

[STEP] Another test: x=0.
Original expression:
0*(0*(1+0)+2*0) -3*(0 -0 +2)=0 -3*2= -6.
Simplified expression:0 +0 -6= -6. Correct.

Adviser: No new advice.

Segment 31. Student.

[STEP] So seems like the simplified expression is x^3 +3x -6. But wait, in my first approach, I thought it was x^3 +6x^2 -3x +6, but that was a mistake. The correct simplification is x^3 +3x -6.

Adviser: END.

I have already reached an answer; I should write it as my final answer and stop.

Segment 32. Student.

[END] But let me check once again. Let's go through the steps once more.
Original expression:
x(x(1+x) +2x) -3(x^2 -x +2)
First, compute x(1+x):
x*1 +x*x =x +x^2.
Add 2x: x +x^2 +2x =x^2 +3x.
Multiply by x: x*(x^2 +3x)=x^3 +3x^2.
Subtract 3(x^2 -x +2):
x^3 +3x^2 -3x^2 +3x -6.
Combine like terms:
x^3 + (3x^2 -3x^2) +3x -6 =x^3 +0x^2 +3x -6 =x^3 +3x -6.
Yes, that's correct. So the simplified expression is x^3 +3x -6.
Therefore, the answer is \boxed{x^3 + 3x - 6}.
Final Answer
The expanded and simplified expression is \boxed{x^3 + 3x - 6}.
</think>
The final answer is \boxed{x^3 + 3x - 6}.

Adviser: No new advice.