LEARNING WHAT TO DO AND HOW TO DO IT: DISTILLING THE DECISIONS BEHIND CHAIN-OF-THOUGHT MAKES REASONING ROBUST

Anonymous authors
Paper under double-blind review

ABSTRACT

Distilling multi-step reasoning from large language models into smaller students commonly relies on chain-of-thought (CoT) traces. Yet a CoT step shows how an operation was carried out while leaving the choice of operation implicit, requiring the student to infer what to do from a problem-specific execution. We propose action-execution reasoning, which factorizes each step into a latent *action* that specifies what to do and an *execution* that specifies how to do it. Building on this representation, **ACT**ion–**E**xecution **D**istillation (ACTED) combines supervised learning from verified teacher trajectories with teacher supervision on student rollouts, coupling reliable demonstrations of correct reasoning with corrective guidance at states encountered by the student. Across five reasoning benchmarks, ACTED achieves the highest average Pass@1 and Pass@5 among the compared methods in every tested configuration, distilling Qwen3 teachers of 1.7B–8B parameters into 0.6B-1.7B students. With a 1.7B teacher and 0.6B base student, ACTED improves StrategyQA Pass@5 from 80.4% for the strongest baseline to 89.3%. Diagnostic and controlled perturbation experiments further indicate reduced execution uncertainty and improved robustness to noisy reasoning history.

1 INTRODUCTION

Distilling reasoning from large language models into smaller students offers a practical way to transfer problem-solving ability. Chain-of-thought (CoT) prompting exposes intermediate reasoning in text (Wei et al., 2022), allowing students to learn from teacher-generated solutions rather than final answers alone (Ho et al., 2023; Magister et al., 2023; Hsieh et al., 2023). Long traces from recent reasoning models make this transfer more promising (Guo et al., 2025). But richer traces alone do not specify what the student should learn from them. Solving a new problem requires two related abilities: choosing the next operation and carrying it out.

A student trained on CoT traces observes the teacher's choice of operation only through its realization, and must infer *what* was done from the realization of *how* it was done. For example, the step "`The unit price of a pen is` $12/3 = 4$" realizes a choice to compute a unit rate, but presents that choice through problem-specific arithmetic. Across problems, the same operation can have many different realizations. Yet the reusable decision is supervised only indirectly. The student must infer it from varied executions while simultaneously learning to reproduce those executions. During generation, the student must make its next decision from an accumulated history that may contain irrelevant details. An explicit operation can instead provide a compact intermediate decision that identifies what to do next before conditioning its execution. These challenges motivate supervision that identifies the next operation explicitly and helps keep its execution focused.

Prior work has exposed parts of the reasoning structure, but at a different granularity or in a different regime. Plans written before solving provide guidance for the whole problem, but do not separately supervise the decision made after each intermediate result (Wang et al., 2023a; Zhou et al., 2024; Zhang et al., 2025; Yang et al., 2026; Bi et al., 2025; Liu et al., 2026). Per-step signals drawn from a fixed inventory provide local structure, but restrict how the next operation can be expressed (Wang et al., 2023b; Wen et al., 2026). We instead propose to present each step as a joint distribution of an action and its execution. An *action* is a short high-level description of the next operation

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.

(e.g., `"Compute unit rate"`), and the *execution* is the operation's realization. The next action decides *what* to do next, and the corresponding execution decides *how* to do it. Thus, a CoT trace observes only the execution marginal distribution obtained by marginalizing over the action.

Viewed through this factorization, standard CoT supervision collapses the decision of what to do and its realization into a single execution-only target. This creates three difficulties for a small student: the reusable choice of operation is supervised only indirectly through problem-specific executions, the execution must be predicted from a broader marginal rather than conditioned on an explicit action, and each step must recover the intended operation from the accumulated reasoning history, making generation more susceptible to irrelevant or erroneous context.

To address the above issues, **ACT**ion-**E**xecution **D**istillation (ACTED) is proposed to learn the joint distribution of the action and execution. Making actions explicit directly supervises the choice of what to do, while conditioning each execution on its action narrows the prediction target and provides a compact intermediate cue for reasoning over imperfect histories. ACTED further combines complementary sources of supervision: an off-policy objective anchors the student to complete and correct teacher solutions, whereas an on-policy objective provides corrective supervision at student-visited states, improving robustness to errors that arise during reasoning.

Our contributions are summarized as follows.

- We formulate *action-execution reasoning*, which factorizes each CoT step into a latent action and its execution, and under which CoT distillation fits only the execution-only marginal. Empirical evidence shows that making the action explicit renders the teacher's decisions supervisable, reduces the uncertainty of executions, and enhances the robustness to noisy history (Section 4.2).
- We propose ACTED, which distills the joint action-execution distribution on both verified teacher trajectories and the student's own rollouts, providing both reliable solution-level supervision and corrective guidance at student-visited states (Section 4.3).
- We evaluate ACTED across five benchmarks, three teachers, and three students. ACTED consistently outperforms six baselines in most settings, and increases Pass@5 on StrategyQA from 80.4% to 89.3%. Further results show that ACTED has higher tolerance to and better recovery from the noisy history (Section 5).

2 Related Work

On-policy and off-policy distillation. Knowledge distillation trains a student to match a teacher's output distribution (Hinton et al., 2015), and its sequence-level form fine-tunes the student on teacher-generated outputs (Kim and Rush, 2016). Such supervision is off-policy: the student is never trained at the states its own decoding reaches, the exposure-bias problem that motivates casting distillation as imitation learning (Ross et al., 2011; Lin et al., 2020). On-policy methods instead score student rollouts with the teacher, under token-level divergences (Agarwal et al., 2024) or a reverse-KL objective optimized with policy gradients (Gu et al., 2024). Subsequent work tunes the balance between the two data sources through scheduled reuse of student outputs, contrastive losses on teacher- and student-generated data, or interleaved teacher-student sampling (Ko et al., 2024; 2025; Xu et al., 2025), and recent analyses characterize when on-policy distillation of reasoning models succeeds (Li et al., 2026). These methods differ in which sequences they supervise, but they apply one divergence uniformly to every token of a sequence. ACTED adopts the GKD interpolation of the two regimes, while its trajectories expose the teacher's per-step actions and its losses differ between action and execution segments.

Reasoning distillation. Chain-of-thought prompting exposes intermediate reasoning as text (Wei et al., 2022), and a large body of work distills it into small students by fine-tuning on teacher rationales (Ho et al., 2023; Magister et al., 2023; Li et al., 2023; Fu et al., 2023; Hsieh et al., 2023), often filtered for correctness as in rejection-sampling self-training (Zelikman et al., 2022; Yuan et al., 2023). At larger scale, long traces from RL-trained reasoning models yield strong small reasoners (Guo et al., 2025), and a small curated set of such traces can suffice (Muennighoff et al., 2025; Ye et al., 2025). A second line distills the structure of the reasoning rather than a flat rationale: teacher reasoning is abstracted into discrete semantic signals (Wen et al., 2026) or constrained by a bank of reusable reasoning paths (Yang et al., 2026), decomposition and solving are split across two small models (Shridhar et al., 2023; Bi et al., 2025), and the reasoning and acting spans of

agent trajectories receive separately masked losses (Liu et al., 2026). Related evidence shows that the planning part of a solution distills more readily than the solving part (Wu et al., 2024), and that step-level signals outperform outcome-level ones for verification (Lightman et al., 2024; Wang et al., 2024). These recipes train the student off-policy on teacher traces, and where their losses are structure-aware, one divergence is shared across segment types. ACTED additionally distills its structured trajectories on the student's own rollouts, with a loss matched to each segment type.

Factorizing reasoning trajectories. Existing formulations of factorizing reasoning differ mainly in where the decision sits and what it ranges over. Planning methods fix one plan or reasoning structure for the whole problem before solving begins, by prompting (Zhou et al., 2022; Wang et al., 2023a; Zhou et al., 2024) or by fine-tuning into an abstract-then-concrete format (Zhang et al., 2025; Hong et al., 2024). Per-step abstractions place a latent at every step but draw it from a closed inventory, such as a learned codebook of planning tokens (Wang et al., 2023b) or a fixed set of discrete semantic signals (Wen et al., 2026). Skill-based methods instead attach discrete, nameable skills to training data or problems, for data selection (Chen et al., 2023) or skill-matched in-context exemplars (Didolkar et al., 2024). Latent-variable treatments regard the rationale as unobserved and maximize the marginal likelihood of the answer (Phan et al., 2023), while search and agent frameworks interleave reasoning with branching over thoughts or with environment actions (Yao et al., 2023; 2022). Our action is a per-step, free-form commitment written by the teacher; it invokes no environment and is made explicit so that it is distilled rather than marginalized.

3 Preliminaries

3.1 Problem Setup

We study the distillation of reasoning ability from a large teacher language model into a small student. Let π^{T} denote a frozen teacher and π^{S}_θ a trainable student with parameters θ, both autoregressive language models over a shared token vocabulary $\mathcal{V}$. We write $p^{\mathrm{T}}(\cdot|c)$ and $p^{\mathrm{S}}_\theta(\cdot|c)$ for their next-token distributions given a context c, so that a token sequence $w = (w_1, \ldots, w_m)$ has probability $p(w|c) = \prod_{t=1}^{m} p(w_t|c, w_{<t})$ for $p \in \{p^{\mathrm{T}}, p^{\mathrm{S}}_\theta\}$. We write $w \sim \pi^{\mathrm{T}}(\cdot|c)$ or $w \sim \pi^{\mathrm{S}}_\theta(\cdot|c)$ for sampling a sequence w from the respective model, and drop the superscript and θ, writing p, when a statement holds for either model. A training set D consists of problems x paired with gold answers y, drawn from a distribution $\mathcal{P}$. A verifier $v(\hat{y}, y) \in \{0, 1\}$ checks whether a predicted answer $\hat{y}$ matches the gold answer y. The goal is to maximize the accuracy of the student's own generations:

$$\max_\theta \; \mathbb{E}_{(x,y)\sim\mathcal{P}} \, \mathbb{E}_{\tau\sim\pi^{\mathrm{S}}_\theta(\cdot|x)} \big[v\big(\mathrm{ans}(\tau), y\big) \big],$$

where $\mathrm{ans}(\tau)$ extracts the final answer from a generated solution τ, a sequence of tokens.

3.2 Off-Policy and On-Policy Distillation

The standard recipe for transferring reasoning is sequence-level distillation (Kim and Rush, 2016; Ho et al., 2023), which samples a solution τ from the teacher for each training problem and fine-tunes the student with the cross-entropy loss, or supervised fine-tuning (SFT):

$$L_{\mathrm{SFT}}(\theta) = -\,\mathbb{E}_{x\sim D} \, \mathbb{E}_{\tau\sim\pi^{\mathrm{T}}(\cdot|x)} \left[\frac{1}{|\tau|} \sum_{t=1}^{|\tau|} \log p^{\mathrm{S}}_\theta(\tau_t|x, \tau_{<t}) \right]. \tag{1}$$

This supervision is off-policy: every training prefix $\tau_{<t}$ is written by the teacher, so the student is never corrected at the states its own generation visits, and errors compound over the generation horizon (Ross et al., 2011; Lin et al., 2020). Generalized knowledge distillation (GKD) (Agarwal et al., 2024) addresses this mismatch by sampling solutions from the student and matching the teacher's next-token distribution at every visited prefix. The on-policy (OP) loss is:

$$L_{\mathrm{OP}}(\theta) = \mathbb{E}_{x\sim D} \, \mathbb{E}_{\tau\sim\pi^{\mathrm{S}}_\theta(\cdot|x)} \left[\frac{1}{|\tau|} \sum_{t=1}^{|\tau|} D_{\mathrm{KL}}\big(p^{\mathrm{T}}(\cdot|x, \tau_{<t}) \,\big\|\, p^{\mathrm{S}}_\theta(\cdot|x, \tau_{<t})\big) \right], \tag{2}$$

where $D_{\mathrm{KL}}(p \,\|\, q) = \sum_w p(w) \log \frac{p(w)}{q(w)}$ denotes the Kullback-Leibler (KL) divergence. GKD interpolates the two regimes with a mixing rate $\lambda \in [0, 1]$, the fraction of training data drawn from student generations.

3.3 Chain-of-Thought Reasoning

Chain-of-thought (CoT) prompting (Wei et al., 2022) elicits a chain of intermediate reasoning steps before the final answer. Given a problem x, the model generates steps $e_1, \ldots, e_k$ sequentially as $e_i \sim p(\cdot|x, e_{<i})$, with $e_{<i} = (e_1, \ldots, e_{i-1})$, followed by a final-answer line $r \sim p(\cdot|x, e_{\leq k})$. Each step e_i is a contiguous span of tokens that realizes one concrete operation. Distilling CoT traces, off-policy by Eq. (1) or on-policy by Eq. (2), supervises the student on the step sequence $(e_1, \ldots, e_k)$.

4 ACTED: Distilling Action–Execution Reasoning

Chain-of-thought (Wei et al., 2022) distillation transfers what the teacher wrote, not what the teacher decided. We factorize each reasoning step into a latent *action* and its *execution* (Section 4.1) and propose to distill their joint distribution instead of the execution-only marginal that CoT traces realize (Section 4.2). **ACT**ion-**E**xecution **D**istillation (ACTED) in Section 4.3 realizes this objective by optimizing both off-policy and on-policy distillation over explicit action-execution trajectories.

4.1 Action–Execution Reasoning

A CoT step e_i (cf. Section 3.3) is a concrete operation such as `"The unit price of a pen is 12/3=4"`. Since its form is tied to problem-specific content, imitating such a step transfers little about which operation to select once the content changes. We thus posit a latent *action* behind each step and regard the step as its *execution*. For example, the step above executes the action `"Compute unit rate"`. The action, which is a short, high-level guidance to one operation, decides *what* to do next given the history; while the execution, which is a longer, concrete realization of that operation in tokens, decides *how* to do it given the action. We call this **action-execution reasoning**.

Formally, let a_i be the action of the i-th step and e_i its execution. A language model can be viewed as *implicitly* sampling an action first and then generating the execution conditioned on it,

$$a_i \sim p(\cdot|h_i) \quad \text{and} \quad e_i \sim p(\cdot|h_i, a_i),$$

where $h_i = (x, a_1, e_1, \ldots, a_{i-1}, e_{i-1})$, with $h_1 = x$, collects the problem and all preceding action-execution pairs, and p is the sequence-level distribution of the model. Under CoT, the actions remain latent and the observable trace contains the executions alone, so each step is sampled from the execution-only marginal:

$$p(e_i|h_i) = \sum\nolimits_{a_i \in \mathcal{V}^\infty} p(a_i|h_i)\, p(e_i|h_i, a_i),$$

with every earlier action likewise marginalized out, collapsing the observable history to $(x, e_{<i})$.

4.2 Distilling Action-Execution Trajectories

CoT distillation trains on traces sampled from the marginal $p(e_i|h_i)$. We instead make the actions explicit in the trajectory and distill the joint distribution:

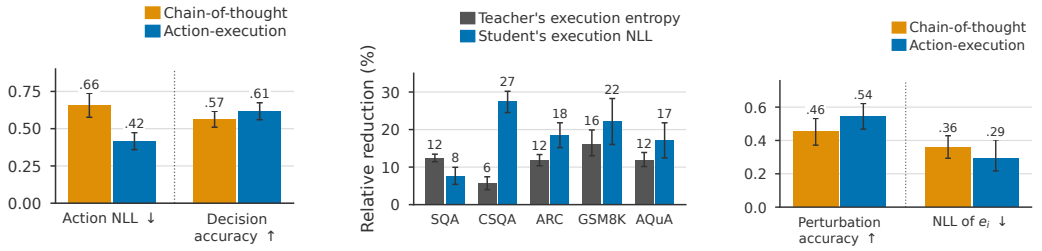
$$p(a_i, e_i|h_i) = p(a_i|h_i)\, p(e_i|h_i, a_i).$$

Rather than imitating the execution-only trace $(e_1, \ldots, e_k)$, the student learns the two factors from separate, explicitly delimited token spans.

Key idea. Distill both *what* to do and *how* to do it: the student learns the teacher's joint action–execution chain $p(a_i, e_i|h_i)$, not the execution-only marginal $p(e_i|h_i)$ that chain-of-thought distillation transfers.

At inference, the student alternates between the two factors, $a_i \sim p(\cdot|h_i)$ and $e_i \sim p(\cdot|h_i, a_i)$. The factorization brings three advantages for distillation as follows.

Decisions become supervisable. A student trained on the marginal $p(e_i|h_i)$ observes the teacher's decision only through its realization, and must infer *what* was done from the surface form of *how* it was done. The explicit trajectory instead emits the decision as a short token span that supervision can target directly, as high-level choices transfer more readily than low-level behavior. In Fig. 1(a), we



(a) Decision supervision (b) Reduced execution uncertainty (c) Enhanced robustness

Figure 1: Empirical evidence supporting three advantages in Section 4.2. **(a)** Compared to a CoT student, an action-execution student has lower negative log-likelihood (NLL) of the teacher's actions (*left*) and higher decision accuracy (*right*). **(b)** Actions reduce up to 16% of the execution entropy of a teacher model (*gray*). Compared to a CoT student, an action-execution student reduces the NLL of the teacher's executions by up to 27% (*blue*). **(c)** Under perturbation on history, an action-execution student has a higher accuracy (*left*) and lower NLL of the clean execution (*right*), showing more robustness. Appendix E shows the experimental setting and more results.

score the teacher's actions under students distilled from chain-of-thought and from action-execution trajectories, and let each student decide every action while the teacher executes it. The action-execution student assigns the teacher's actions a lower NLL, and leads to a higher final-answer accuracy. The student thus learns the teacher's decisions, not only their realizations.

Execution uncertainty is reduced. Conditioning on the action yields an execution target whose entropy is no greater than that of the marginal distribution, i.e., $\mathcal{H}(E_i|h_i, A_i) \leq \mathcal{H}(E_i|h_i)$, where $\mathcal{H}$ denotes conditional entropy and A_i, E_i are the random action and execution of step i. In Fig. 1(b), we estimate the relative reduction of the entropy $[\mathcal{H}(E|H) - \mathcal{H}(E|H, A)]/\mathcal{H}(E|H)$ of a teacher model. Moreover, we investigate the NLL of a student over the teacher's executions. Conditioning on the action removes up to 16% of the execution entropy in the teacher and up to 27% of the NLL in the students. It shows that actions resolve part of the uncertainty before the execution is written.

Robustness is enhanced. As reasoning proceeds, h_i accumulates every preceding action-execution pair, only a small part of which bears on the next operation, and language models use information in long contexts unevenly (Liu et al., 2024). The explicit action writes this selection into the trajectory as a short span that condenses what h_i implies for the coming operation, so that the execution can condition on it rather than on the whole history. In Fig. 1(c), we insert two action-execution pairs from other problems early in each student's own history, let the student decide its next action, and measure the accuracy of its final answer and the NLL of its original execution. Accuracy increases by 8%, and NLL decreases by 0.7. The action thus filters the irrelevant history and makes the student more robust to noise.

4.3 ACTED: Action-Execution Distillation

Two sources of trajectories can carry the joint, with complementary strengths: teacher trajectories exhibit complete, competent solutions but supervise the student only at states the teacher visits (Eq. (1)), while student rollouts place supervision where the student's own decoding goes but need not contain a correct solution (Eq. (2)). ACTED interpolates an off-policy loss on verified teacher trajectories with an on-policy loss on the student's own rollouts, summarized by Algorithm 1.

Distilling teacher trajectories. For a training pair (x, y), the teacher is prompted to solve x as an explicit action-execution trajectory:

$$\tau = [\, a_1, e_1, a_2, e_2, \ldots, a_k, e_k, r \,], \tag{3}$$

where each action a_i is a line naming a free-form operation (e.g., `Action 1: <<set up an equation>>`), each execution e_i is one or more lines applying that operation to the problem, r is the final-answer line, and τ denotes its token sequence. Because distilling an incorrect trajectory trains the student toward reasoning that demonstrably failed, we keep a trajectory only if it is well-formed and its answer is verified correct, $\mathrm{keep}(\tau, y) = \mathrm{valid}(\tau) \wedge v(\mathrm{ans}(\tau), y)$, where $\mathrm{valid}(\tau)$ requires τ to follow an alternating action-execution structure (see Appendices A and H for details and prompts,

Algorithm 1 ACTED: action-execution distillation

Input: dataset D, teacher π^{T}, student π_θ^{S}, mixing rate λ, iterations N

for iteration $= 1, \ldots, N$ **do**
  if $\mathrm{Bernoulli}(1-\lambda) = 1$ **then** ▷ Distilling teacher trajectories
    sample $(x, y) \sim D$; decode $\tau = [a_1, e_1, \ldots, a_k, e_k, r]$ from $\pi^{\mathrm{T}}(\cdot|x)$
    if $\mathrm{keep}(\tau, y) = 1$ **then**
      update θ on L_{off} in Eq. (4) over τ
  else ▷ Distilling on student's rollouts
    sample $x \sim D$; initialize the context $c \leftarrow x$
    repeat
      decode the next line σ greedily from $\pi_\theta^{\mathrm{S}}(\cdot|c)$
      if σ is an action a_i or the final answer r **then** ▷ Action
        update θ on $\ell_{\mathrm{ACT}}(\sigma; \theta)$ in Eq. (5)
      else ▷ Execution
        update θ on $\ell_{\mathrm{EXE}}(\sigma; \theta)$ in Eq. (6)
      $c \leftarrow (c, \sigma)$
    until $\sigma = r$
return θ

respectively). The student is then trained on the kept trajectories with the SFT loss of Eq. (1),

$$L_{\mathrm{off}}(\theta) = -\,\mathbb{E}_{(x,y)\sim D}\,\mathbb{E}_{\tau\sim\pi^{\mathrm{T}}(\cdot|x)}\left[\mathrm{keep}(\tau, y)\,\frac{1}{|\tau|}\sum\nolimits_{t=1}^{|\tau|}\log p_\theta^{\mathrm{S}}(\tau_t|x, \tau_{<t})\right]. \tag{4}$$

Distilling on student's rollouts. A student rollout $\tau \sim \pi_\theta^{\mathrm{S}}(\cdot|x)$ in the format of Eq. (3) decomposes into $K(\tau)$ line segments $\sigma_1, \ldots, \sigma_{K(\tau)}$, each of type $\mathrm{type}(\sigma) = \textsc{act}$ (action and final-answer lines) or $\mathrm{type}(\sigma) = \textsc{exe}$ (all other lines). For a segment $\sigma = (w_1, \ldots, w_m)$ whose context c comprises the problem and the preceding segments, one teacher-forced pass yields $p^{\mathrm{T}}(\cdot|c, w_{<t})$ at every position of σ. Action-type segments are supervised by cross-entropy on the teacher's greedy labels,

$$\ell_{\mathrm{ACT}}(\sigma; \theta) = -\frac{1}{m}\sum\nolimits_{t=1}^{m}\log p_\theta^{\mathrm{S}}\big(w_t^*|c, w_{<t}\big), \qquad w_t^* = \arg\max\nolimits_{u\in\mathcal{V}}\, p^{\mathrm{T}}(u|c, w_{<t}), \tag{5}$$

which encourages the student to learn a sharp mapping from the current context to a compact action decision, rather than matching the teacher's full uncertainty over alternative actions. Given the action, the execution contains dense token-level procedural knowledge that cannot be fully captured by a single sampled trajectory. Therefore, execution-type segments are minimized by the forward KL divergence of Eq. (2) restricted to the segment,

$$\ell_{\mathrm{EXE}}(\sigma; \theta) = \frac{1}{m}\sum\nolimits_{t=1}^{m} D_{\mathrm{KL}}\big(p^{\mathrm{T}}(\cdot|c, w_{<t})\,\big\|\,p_\theta^{\mathrm{S}}(\cdot|c, w_{<t})\big)\,. \tag{6}$$

ACTED combines the two branches with a mixing rate λ,

$$L(\theta) = (1-\lambda)\,L_{\mathrm{off}}(\theta) + \lambda\,\mathbb{E}_{x\sim D}\,\mathbb{E}_{\tau\sim\pi_\theta^{\mathrm{S}}(\cdot|x)}\left[\frac{1}{K(\tau)}\sum\nolimits_{j=1}^{K(\tau)}\ell_{\mathrm{type}(\sigma_j)}\big(\sigma_j; \theta\big)\right], \tag{7}$$

where $\lambda = 0$ recovers verified sequence-level distillation and $\lambda = 1$ a segment-typed analogue of Eq. (2). As in GKD, rollouts only supply the states at which the losses are evaluated, and no gradient flows through their decoding. Algorithm 1 summarizes our algorithm.

Discussion. The off-policy objective in Eq. (4) distills verified teacher trajectories, providing the student with stable demonstrations of complete and correct reasoning traces. However, because these trajectories are fixed, supervision is concentrated on states visited by the teacher and does not directly address deviations that arise under the student's own policy. In contrast, the on-policy objectives in Eqs. (5) and (6) provides teacher's guidance on student's own states, especially when there are wrong actions or imperfect executions. Consequently, the off-policy branch primarily teaches how to follow a correct reasoning trace, whereas the on-policy branch teaches how to continue reasoning appropriately after the student has deviated from one, improving robustness to errors accumulated during autoregressive generation.

Table 1: Pass@1 (P@1) and Pass@5 (P@5) with the Qwen3-1.7B teacher and the Qwen3-0.6B-Base student. Test accuracies (%) are reported as mean ± standard deviation over three seeds, except Scaffold, Think-SFT, and SGFT due to their weak performance. Bold and underline mark the best and second-best core method per column. The last three rows compare ACTED with the best, worst, and average performance across all baselines, respectively.

	StrategyQA		CSQA		ARC-C		GSM8K		AQuA		Average	
Method	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
ACTED (ours)	**55.1** ± 1.3	**89.3** ± 0.9	**64.1** ± 0.6	**81.2** ± 0.3	**68.5** ± 0.5	85.5 ± 0.5	**58.8** ± 0.6	75.3 ± 1.5	**48.6** ± 2.2	75.4 ± 0.5	**59.0**	**81.3**
GKD	54.6 ± 2.0	78.6 ± 2.2	62.5 ± 1.6	80.5 ± 1.4	66.2 ± 1.4	84.3 ± 1.1	57.9 ± 1.5	76.0 ± 0.3	47.9 ± 4.2	**75.5** ± 2.8	57.8	79.0
CoT-SFT	53.5 ± 0.4	80.4 ± 0.8	63.1 ± 0.9	80.5 ± 0.3	68.4 ± 0.9	85.0 ± 1.1	56.3 ± 0.5	76.5 ± 1.2	45.4 ± 1.4	74.3 ± 2.4	57.4	79.4
D-RPC	48.4 ± 6.2	63.9 ± 2.5	59.3 ± 0.9	80.4 ± 0.9	62.1 ± 1.2	**86.3** ± 1.0	52.0 ± 0.4	**77.1** ± 0.7	35.8 ± 3.0	68.4 ± 1.9	51.5	75.2
Scaffold	35.2	73.7	41.1	73.5	42.3	75.7	43.8	69.4	13.4	26.8	35.2	63.8
Think-SFT	34.2	62.0	31.9	63.2	28.7	54.0	43.3	66.7	10.1	18.4	29.6	52.9
SGFT	39.2	72.8	43.4	73.5	40.1	78.2	49.1	74.4	11.7	28.3	36.7	65.4
Δ vs. best	+0.5	+8.9	+1.0	+0.7	+0.1	−0.8	+0.9	−1.8	+0.7	−0.1	+1.2	+1.9
Δ vs. worst	+20.9	+27.3	+32.2	+18.0	+39.8	+31.5	+15.5	+8.6	+38.5	+57.0	+29.4	+28.4
Δ vs. avg.	+10.9	+17.4	+13.9	+5.9	+17.2	+8.3	+8.4	+2.0	+21.2	+26.8	+14.3	+12.0

Table 2: Results with larger teachers, (a) Qwen3-4B and (b) Qwen3-8B, and the Qwen3-0.6B-Base student. Protocol and layout as in Table 1.

(a) Teacher Qwen3-4B

	StrategyQA		CSQA		ARC-C		GSM8K		AQuA		Average	
Method	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
ACTED (ours)	**57.8** ± 0.4	**86.8** ± 0.5	**64.4** ± 1.7	80.9 ± 0.7	**70.7** ± 1.2	**88.9** ± 0.3	**64.8** ± 0.6	79.4 ± 0.8	**47.6** ± 3.1	**73.9** ± 0.8	**61.1**	**82.0**
GKD	54.6 ± 0.6	77.4 ± 2.0	63.1 ± 1.8	82.0 ± 1.3	70.6 ± 0.8	86.6 ± 0.5	60.7 ± 0.2	**81.3** ± 0.7	47.1 ± 1.3	72.4 ± 1.8	59.2	80.0
CoT-SFT	56.6 ± 1.1	80.4 ± 1.7	64.0 ± 0.6	**83.3** ± 1.2	68.1 ± 0.3	87.7 ± 0.8	64.7 ± 1.4	80.7 ± 0.6	**47.6** ± 3.6	73.5 ± 1.9	60.2	81.1
D-RPC	51.1 ± 1.2	68.1 ± 1.0	56.9 ± 1.1	82.1 ± 0.3	66.0 ± 1.0	88.1 ± 0.4	53.6 ± 1.2	77.4 ± 1.2	43.8 ± 3.6	72.0 ± 3.9	54.3	77.5
Scaffold	31.7	65.9	37.5	75.8	40.6	76.6	25.9	58.1	6.7	17.3	28.5	58.7
Think-SFT	22.6	48.9	31.4	65.1	22.4	52.5	24.3	58.3	5.9	13.8	21.3	47.7
SGFT	5.5	20.5	21.5	57.7	22.9	53.6	14.9	42.6	4.7	17.3	13.9	38.4
Δ vs. best	+1.2	+6.4	+0.4	−2.4	+0.1	+0.8	+0.1	−1.9	0.0	+0.4	+0.9	+0.9
Δ vs. worst	+52.3	+66.3	+42.9	+23.2	+48.3	+36.4	+49.9	+36.8	+42.9	+60.1	+47.2	+43.6
Δ vs. avg.	+20.8	+26.6	+18.7	+6.6	+22.3	+14.7	+24.1	+13.0	+21.6	+29.5	+21.5	+18.1

(b) Teacher Qwen3-8B

	StrategyQA		CSQA		ARC-C		GSM8K		AQuA		Average	
Method	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
ACTED (ours)	**59.5** ± 1.7	**85.2** ± 3.0	**62.6** ± 1.0	80.5 ± 0.2	**68.3** ± 1.2	**89.1** ± 1.0	**62.7** ± 0.5	**77.6** ± 0.6	**47.6** ± 1.2	**77.5** ± 0.2	**60.1**	**82.0**
GKD	54.3 ± 1.3	74.1 ± 2.0	62.3 ± 2.4	81.4 ± 0.6	67.0 ± 0.2	87.5 ± 1.0	58.4 ± 1.3	77.2 ± 0.2	47.1 ± 1.2	74.5 ± 3.3	57.8	78.9
CoT-SFT	55.4 ± 0.2	78.1 ± 2.4	60.8 ± 0.7	**82.3** ± 0.2	67.6 ± 0.4	87.9 ± 0.5	55.9 ± 1.3	76.6 ± 0.5	44.2 ± 1.6	73.4 ± 0.8	56.8	79.7
D-RPC	52.3 ± 1.8	73.9 ± 1.3	58.4 ± 0.9	**82.3** ± 0.2	65.5 ± 0.7	89.0 ± 0.5	53.4 ± 1.1	77.0 ± 1.0	37.9 ± 1.6	71.8 ± 1.6	53.5	78.8
Scaffold	25.3	47.7	30.5	69.1	31.4	71.3	21.8	58.6	2.8	15.4	22.4	52.4
Think-SFT	8.3	33.0	26.8	59.0	13.8	37.3	20.2	54.8	2.8	7.5	14.4	38.3
SGFT	0.7	5.7	16.0	45.9	19.5	48.0	21.7	44.5	2.4	9.4	12.0	30.7
Δ vs. best	+4.1	+7.1	+0.3	−1.8	+0.7	+0.1	+4.3	+0.4	+0.5	+3.0	+2.3	+2.3
Δ vs. worst	+58.8	+79.5	+46.6	+34.6	+54.5	+51.8	+42.5	+33.1	+45.2	+70.0	+48.1	+51.3
Δ vs. avg.	+26.8	+33.1	+20.1	+10.5	+24.2	+18.9	+24.1	+12.8	+24.7	+35.5	+24.0	+22.2

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate on five reasoning benchmarks of four types: multi-hop reasoning (StrategyQA; Geva et al., 2021), commonsense question answering (CommonsenseQA; Talmor et al., 2019), science question answering (ARC-Challenge; Clark et al., 2018), and mathematical reasoning (GSM8K, and AQuA; Cobbe et al., 2021; Ling et al., 2017). Appendix C gives the splits and statistics.

Models and training. We use models from the Qwen3 family (Yang et al., 2025). The teachers include instruction-tuned Qwen3-1.7B, Qwen3-4B, and Qwen3-8B, and students include the Qwen3-0.6B base model and instruction-tuned model. All methods train LoRA adapters (Hu et al., 2021) of rank $r = 64$ and scaling $\alpha = 128$. Appendix C shows more details.

Table 3: Results with the Qwen3-1.7B teacher and the instruction-tuned Qwen3-0.6B student. Protocol and layout as in Table 1.

	StrategyQA		CSQA		ARC-C		GSM8K		AQuA		Average	
Method	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
ACTED (ours)	**54.5** ± 0.3	**87.9**	**60.0** ± 1.0	**79.8** ± 1.2	**67.3** ± 0.4	**83.4** ± 1.1	**57.7** ± 0.7	75.2 ± 0.4	**48.6** ± 2.6	74.9 ± 2.9	**57.6**	**80.2**
GKD	53.8 ± 2.2	77.4 ± 0.1	59.2 ± 1.1	78.9 ± 0.8	65.1 ± 0.5	82.1 ± 1.0	57.5 ± 1.0	73.2 ± 1.4	47.1 ± 1.6	**78.0** ± 0.4	56.5	77.9
CoT-SFT	53.8 ± 0.6	79.9 ± 0.8	58.7 ± 1.0	78.8 ± 0.3	64.4 ± 0.3	82.2 ± 0.8	54.7 ± 1.3	74.1 ± 0.3	45.8 ± 1.6	73.9 ± 1.9	55.5	77.8
D-RPC	51.8 ± 0.5	64.5 ± 1.8	56.9 ± 1.3	77.8 ± 0.5	60.9 ± 0.4	81.5 ± 0.9	50.1 ± 1.5	**76.0** ± 0.9	39.6 ± 4.6	70.5 ± 0.7	51.9	74.1
Scaffold	44.3	65.1	42.9	68.1	35.6	63.1	46.3	70.8	13.4	24.0	36.5	58.2
Think-SFT	29.3	55.9	31.0	60.2	25.9	49.7	44.1	68.0	11.0	18.5	28.3	50.5
SGFT	41.2	69.6	41.9	73.9	40.8	73.5	48.1	74.5	9.8	26.0	36.3	63.5
Δ vs. best	+0.7	+8.0	+0.8	+0.9	+2.2	+1.2	+0.2	−0.8	+1.5	−3.1	+1.1	+2.3
Δ vs. worst	+25.2	+32.0	+29.0	+19.6	+41.4	+33.7	+13.6	+7.2	+38.8	+56.4	+29.3	+29.7
Δ vs. avg.	+8.8	+19.2	+11.6	+6.9	+18.5	+11.4	+7.6	+2.4	+20.8	+26.4	+13.4	+13.2

Table 4: Objective ablations with the Qwen3-1.7B teacher and the Qwen3-0.6B-Base student. ACTED-SFT supervises execution segments with the teacher-argmax cross-entropy, and ACTED-offline removes the on-policy branch and fine-tunes offline. Bold and underline mark the best and second-best variant per dataset and metric.

	StrategyQA		CSQA		ARC-C		GSM8K		AQuA	
Method	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
ACTED	**55.1** ± 1.3	**89.3** ± 0.9	**64.1** ± 0.6	**81.2** ± 0.3	**68.5** ± 0.5	**85.5** ± 0.5	58.8 ± 0.6	75.3 ± 1.5	48.6 ± 2.2	75.4 ± 0.5
ACTED-SFT	53.3 ± 0.9	84.1 ± 1.7	61.5 ± 0.7	78.5 ± 1.2	66.8 ± 0.4	83.2 ± 1.4	55.3 ± 0.3	74.2 ± 1.7	48.4 ± 2.0	71.8 ± 1.1
DAER	**55.1** ± 2.0	88.5 ± 1.0	62.1 ± 1.0	78.8 ± 0.8	67.1 ± 0.7	85.2 ± 0.8	**59.5** ± 0.5	**77.0** ± 0.7	**50.7** ± 1.2	**75.7** ± 1.4

Evaluation. Pass@1 (P@1) and Pass@5 (P@5) are used to measure the model's performance. P@1 measures the model's single-sample precision and sampling efficiency, while P@5 measures the model's reasoning coverage, exploration capacity, and capability boundary (Chen et al., 2026; Kruszewski et al., 2026). Tables report the mean and standard deviation across three seeds, except Scaffold, Think-SFT, and SGFT due to their weak performance.

Baselines. All seven baselines share the models and configurations (see Appendix C). *GKD* (Agarwal et al., 2024) mixes on-policy token-level distillation on student rollouts with supervised learning on teacher chain-of-thought. *CoT-SFT* (Ho et al., 2023; Magister et al., 2023) fine-tunes offline on teacher chain-of-thought traces. *D-RPC* (Yang et al., 2026) distills rationales constrained by a bank of reusable reasoning paths. *Scaffold* (Wen et al., 2026) distills the teacher's reasoning as a sequence of discrete semantic signals. *Think-SFT* (Wen et al., 2026) fine-tunes on extended-thinking teacher traces. *SGFT* (Bi et al., 2025) distills teacher-written guidance followed by the solution.

5.2 Main Results

Main results. Table 1 reports Pass@1 (P@1) and Pass@5 (P@5) with the Qwen3-1.7B teacher and the Qwen3-0.6B-Base student. ACTED consistently achieves the highest P@1 across all datasets, suggesting that our method improves the precision of individual samples. The gains are more pronounced for P@5, with ACTED substantially outperforming the baselines on several datasets. In particular, it exceeds the strongest baseline by 8.9% on StrategyQA. These improvements are also reflected in the average performance, where ACTED consistently surpasses the baselines. Notably, ACTED exhibits robust performance across datasets, as it consistently improves over the average baseline performance and, in most cases, outperforms the strongest baseline. In the few settings where ACTED falls short of the best baseline, the performance gap remains marginal.

Scaling the teacher and student. In Table 2, we study the performance of ACTED across different teacher and student scales. Across teacher sizes, ACTED consistently outperforms nearly all baselines on all datasets, demonstrating its robustness to variations in teacher capacity. In Table 3, we further investigate the effect of a student's reasoning ability on distillation performance. On average, as the student's reasoning ability increases, the teacher-student capability gap narrows, leaving less headroom for further improvement through distillation. Consequently, the baselines exhibit a reduction in distillation gains with stronger students. In contrast, ACTED remains consistently competitive

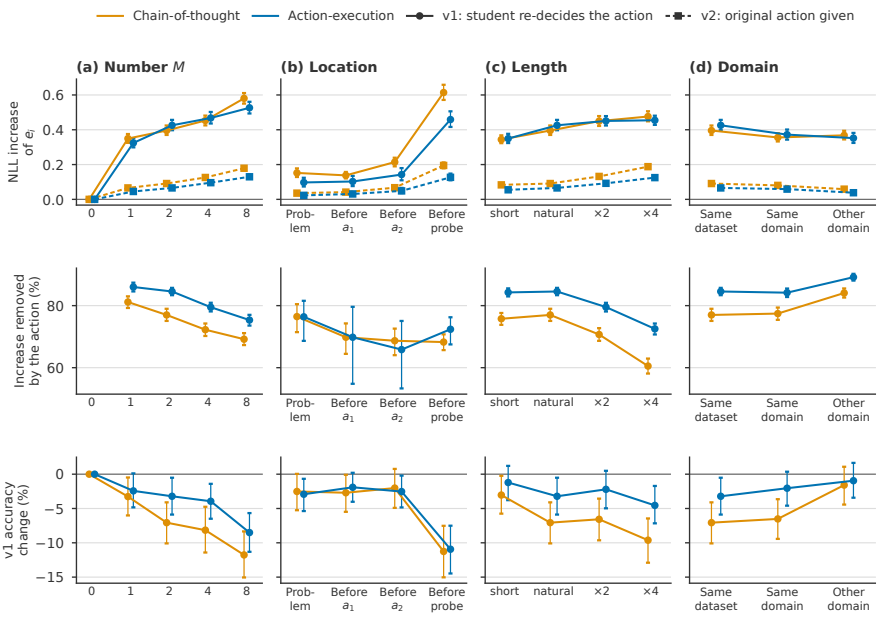


Figure 2: Robustness of ACTED. We inject action-execution pairs from other problems and vary the number (**a**), location (**b**), length (**c**), and domain (**d**) of the inserted pairs. **Top**: increase in the per-token NLL of the student's original execution e_i over the clean history when the student re-decides its action (v1) and when its original action is given (v2). **Middle**: share of the v1 increase that v2 removes. **Bottom**: change in final-answer accuracy under v1.

across student capacities, outperforming the baselines while exhibiting only a modest performance degradation. Further results of distilling to a Qwen3-1.7B student are shown in Appendix D.

Ablation study. Table 4 ablates the key components of ACTED. Specifically, ACTED-SFT supervises the execution segments with Eq. (4), and ACTED-offline removes the on-policy loss and only fine-tunes in an off-policy manner. The performance drop of ACTED-SFT suggests that execution benefits from the dense supervision provided by the teacher distribution. Meanwhile, the relative effectiveness of on- and off-policy distillation varies across datasets, consistent with (Agarwal et al., 2024).

5.3 ACTED IS MORE ROBUST

The robustness of ACTED is shown in Fig. 2 by varying multiple factors of the injected noise in the history, with additional details in Appendix G. Across most settings, the action-execution student exhibits greater robustness to perturbations than the CoT student, as evidenced by a smaller increase in the negative log-likelihood (NLL) of the subsequent execution and a smaller degradation in task accuracy. Moreover, when conditioned on the original action, the action-execution student assigns substantially higher likelihood to the corresponding execution than the CoT student, suggesting that the explicit action provides a more reliable conditioning signal for recovery.

6 CONCLUSION

In this paper, we show that reasoning distillation benefits from supervising not only how a reasoning step is carried out, but also the decision that determines what should be done next. Making this decision explicit improves its learnability, reduces uncertainty in the subsequent execution, and provides a compact interface between the accumulated history and the next step. Proposed ACTED operationalizes this perspective by combining reliable supervision from verified teacher solutions with guidance on states encountered by the student itself. Across teacher scales and student initializations, this design consistently improves reasoning performance and robustness to imperfect histories. More broadly, these findings indicate that explicitly modeling intermediate reasoning decisions can provide a useful inductive bias for transferring reasoning capabilities to smaller language models.

AI DISCLOSURE

In this work, we used generative AI tools to polish writing and create and edit code. We have not used generative AI tools for other tasks with required disclosure, and the rest of the required disclosure tasks are not applicable to this work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Agarwal, R., Vieillard, N., Zhou, Y., Stanczyk, P., Ramos Garea, S., Geist, M., and Bachem, O. (2024). On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pages 21246–21263.

Bi, J., Wu, Y., Xing, W., and Wei, Z. (2025). Enhancing the reasoning capabilities of small language models via solution guidance fine-tuning. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9074–9084.

Chen, M., Roberts, N., Bhatia, K., Wang, J., Zhang, C., Sala, F., and Ré, C. (2023). Skill-it! a data-driven skills framework for understanding and training language models. volume 36, pages 36000–36040.

Chen, Z., Lu, R., Zhao, A., Wang, Z., Yue, Y., Song, S., and Huang, G. (2026). Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? volume 38, pages 57654–57689.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. (2018). Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. (2021). Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Didolkar, A., Goyal, A., Ke, N. R., Guo, S., Valko, M., Lillicrap, T., Rezende, D., Bengio, Y., Mozer, M., and Arora, S. (2024). Metacognitive capabilities of llms: An exploration in mathematical problem solving. volume 37, pages 19783–19812.

Fu, Y., Peng, H., Ou, L., Sabharwal, A., and Khot, T. (2023). Specializing smaller language models towards multi-step reasoning. In *International Conference on Machine Learning*, pages 10421–10430. PMLR.

Geva, M., Khashabi, D., Segal, E., Khot, T., Roth, D., and Berant, J. (2021). Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Gu, Y., Dong, L., Wei, F., and Huang, M. (2024). Minillm: Knowledge distillation of large language models. In *International Conference on Learning Representations*, volume 2024, pages 32694–32717.

Guo, D., Yang, D., Zhang, H., Song, J., Wang, P., Zhu, Q., Xu, R., Zhang, R., Ma, S., Bi, X., et al. (2025). Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Ho, N., Schmid, L., and Yun, S.-Y. (2023). Large language models are reasoning teachers. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 14852–14882.

Hong, R., Zhang, H., Pan, X., Yu, D., and Zhang, C. (2024). Abstraction-of-thought makes language models better reasoners. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1993–2027.

Hsieh, C.-Y., Li, C.-L., Yeh, C.-K., Nakhost, H., Fujii, Y., Ratner, A., Krishna, R., Lee, C.-Y., and Pfister, T. (2023). Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the association for computational linguistics: ACL 2023*, pages 8003–8017.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). Lora: Low-rank adaptation of large language models.

Kim, Y. and Rush, A. M. (2016). Sequence-level knowledge distillation. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 1317–1327.

Ko, J., Chen, T., Kim, S., Ding, T., Liang, L., Zharkov, I., and Yun, S.-Y. (2025). Distillm-2: A contrastive approach boosts the distillation of llms. *arXiv preprint arXiv:2503.07067*.

Ko, J., Kim, S., Chen, T., and Yun, S.-Y. (2024). Distillm: Towards streamlined distillation for large language models. *arXiv preprint arXiv:2402.03898*.

Kruszewski, G., Erbacher, P., Rozen, J., and Dymetman, M. (2026). Whatever remains must be true: Filtering drives reasoning in llms, shaping diversity. In *International Conference on Learning Representations*, volume 2026, pages 85291–85316.

Li, L. H., Hessel, J., Yu, Y., Ren, X., Chang, K.-W., and Choi, Y. (2023). Symbolic chain-of-thought distillation: Small models can also "think" step-by-step. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2665–2679.

Li, Y., Zuo, Y., He, B., Zhang, J., Xiao, C., Qian, C., Yu, T., Gao, H.-a., Yang, W., Liu, Z., et al. (2026). Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. (2024). Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pages 39578–39601.

Lin, A., Wohlwend, J., Chen, H., and Lei, T. (2020). Autoregressive knowledge distillation through imitation learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6121–6133.

Ling, W., Yogatama, D., Dyer, C., and Blunsom, P. (2017). Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 158–167.

Liu, J., Kong, Z., Dong, P., Yang, C., Li, T., Xie, Y., Gong, Y., Shen, X., Zhao, P., Tang, H., et al. (2026). Structured agent distillation for large language model agents. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems*, pages 3676–3685.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., and Liang, P. (2024). Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173.

Magister, L. C., Mallinson, J., Adamek, J., Malmi, E., and Severyn, A. (2023). Teaching small language models to reason. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1773–1781.

Muennighoff, N., Yang, Z., Shi, W., Li, X. L., Fei-Fei, L., Hajishirzi, H., Zettlemoyer, L., Liang, P., Candès, E., and Hashimoto, T. B. (2025). s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20286–20332.

Phan, D., Hoffman, M. D., Dohan, D., Douglas, S., Le, T. A., Parisi, A., Sountsov, P., Sutton, C., Vikram, S., and A Saurous, R. (2023). Training chain-of-thought via latent-variable inference. volume 36, pages 72819–72841.

Ross, S., Gordon, G., and Bagnell, D. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings.

Shridhar, K., Stolfo, A., and Sachan, M. (2023). Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7059–7073.

Talmor, A., Herzig, J., Lourie, N., and Berant, J. (2019). Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *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)*, pages 4149–4158.

Wang, L., Xu, W., Lan, Y., Hu, Z., Lan, Y., Lee, R. K.-W., and Lim, E.-P. (2023a). Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 2609–2634.

Wang, P., Li, L., Shao, Z., Xu, R., Dai, D., Li, Y., Chen, D., Wu, Y., and Sui, Z. (2024). Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439.

Wang, X., Caccia, L., Ostapenko, O., Yuan, X., Wang, W. Y., and Sordoni, A. (2023b). Guiding language model reasoning with planning tokens.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. (2022). Chain-of-thought prompting elicits reasoning in large language models. volume 35, pages 24824–24837.

Wen, X., Huang, J., Li, Z., Li, M., Zhong, J., Xu, Z., Yuan, M., Huang, Y., and Xu, Q. (2026). Reasoning scaffolding: Distilling the flow of thought from llms. In *International Conference on Learning Representations*, volume 2026, pages 80433–80459.

Wu, Z., Bai, R. H., Zhang, A., Gu, J., Vydiswaran, V. V., Jaitly, N., and Zhang, Y. (2024). Divide-or-conquer? which part should you distill your llm? In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2572–2585.

Xu, W., Han, R., Wang, Z., Le, L., Madeka, D., Li, L., Wang, W., Agarwal, R., Lee, C.-Y., and Pfister, T. (2025). Speculative knowledge distillation: Bridging the teacher-student gap through interleaved sampling. In *International Conference on Learning Representations*, volume 2025, pages 64616–64646.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al. (2025). Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yang, J., Wang, J., Wu, J., Leung, H., Zhou, J., and Drew, S. (2026). Structural rationale distillation via reasoning space compression. *arXiv preprint arXiv:2605.07139*.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., and Narasimhan, K. (2023). Tree of thoughts: Deliberate problem solving with large language models. volume 36, pages 11809–11822.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. (2022). React: Synergizing reasoning and acting in language models.

Ye, Y., Huang, Z., Xiao, Y., Chern, E., Xia, S., and Liu, P. (2025). Limo: Less is more for reasoning.

Yuan, Z., Yuan, H., Li, C., Dong, G., Lu, K., Tan, C., Zhou, C., and Zhou, J. (2023). Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*.

Zelikman, E., Wu, Y., Mu, J., and Goodman, N. (2022). Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488.

Zhang, J., Sung, F., Yang, Z., Gao, Y., and Zhang, C. (2025). Learning to plan before answering: Self-teaching llms to learn abstract plans for problem solving. In *International Conference on Learning Representations*, volume 2025, pages 52379–52394.

Zhou, D., Schärli, N., Hou, L., Wei, J., Scales, N., Wang, X., Schuurmans, D., Cui, C., Bousquet, O., Le, Q., et al. (2022). Least-to-most prompting enables complex reasoning in large language models.

Zhou, P., Pujara, J., Ren, X., Chen, X., Cheng, H.-T., Le, Q. V., Zhou, D., Mishra, S., Zheng, H. S., et al. (2024). Self-discover: Large language models self-compose reasoning structures. volume 37, pages 126032–126058.

A TRAJECTORY ELICITATION

This section details how the off-policy branch of Section 4.3 obtains its teacher trajectories. The format of Eq. (3) is elicited purely by prompting: the teacher receives a fixed instruction, a format exemplar, and the question. The instruction asks for abstract actions that never mention the problem's specific numbers or objects, and the instruction and exemplar illustrate a few action names, but the vocabulary is otherwise unconstrained, so the teacher names whatever operation fits the state of its solution. Teacher generation happens online, during training, rather than as a separate corpus-construction stage. Equation (4) supervises only trajectory tokens, with the prompt masked. The structural predicate $\mathrm{valid}(\tau)$ accepts a trajectory only if it contains at least two well-formed action lines, at least $k-1$ execution lines for k action lines, and a final-answer marker; the predicate is deliberately one execution line more permissive than Eq. (3), so a decode that omits a single execution line still passes. When τ lacks a final-answer marker, we set $\mathrm{ans}(\tau) = \varnothing$ with $v(\varnothing, y) = 0$, so such a trajectory is never kept. Because batches are always filled from this pool, the implemented step optimizes the expectation of Eq. (4) conditioned on $\mathrm{keep}(\tau, y) = 1$, which matches Eq. (4) up to a dataset-dependent factor equal to the keep rate.

B IMPLEMENTATION

The student is trained with low-rank adapters (LoRA; Hu et al., 2021) of rank 64, scaling $\alpha = 128$, and no dropout on all attention and MLP projection matrices, with adapters kept in full precision over a bfloat16 base model. The student rolls out, trains, and is evaluated on the same plain-text prompt (instruction, format exemplar, and question) that the teacher generates and scores under; the teacher additionally wraps this prompt in its own chat template with its extended thinking mode disabled. All remaining hyperparameters are listed in Appendix C.

C EXPERIMENTAL DETAILS

This section complements the setup of Section 5.1.

Datasets. StrategyQA (Geva et al., 2021) is boolean multi-hop QA with implicit reasoning strategies (687 test questions). CommonsenseQA (CSQA; Talmor et al., 2019) is five-way commonsense QA (the official 1,221-question development set serves as the test set). ARC-Challenge (ARC-C; Clark et al., 2018) is multiple-choice science QA with three to five options (official test set, 1,172 questions). GSM8K (Cobbe et al., 2021) is grade-school math (1,319 test problems), and AQuA (Ling et al., 2017) is five-way algebraic multiple choice (254 test problems).

Models and training. ACTED runs $N = 600$ iterations with mixing rate $\lambda = 0.5$, using AdamW with learning rate 10^{-4}, cosine decay after 40 warmup steps, and gradient clipping at 1.0. We use a learning rate 5×10^{-5} for AQuA on the main pair. At the 1.7B and 4B teachers, the on-policy branch advances $B = 8$ rollouts in lockstep with segment cap $L_{\mathrm{seg}} = 96$ tokens, at most $K_{\max} = 24$ segments, and a total budget of $L_{\max} = 512$ generated tokens per rollout. Following Ross et al. (2011), the batch-mean segment loss is backpropagated and an optimizer step is taken after every segment round, so each subsequent segment is decoded, and supervised, on states that the already-updated student actually visits. The anchoring branch samples 6 training questions per step with a teacher generation budget of 768 tokens and a gradient micro-batch of 4 trajectories.

Baselines. All baselines share the teacher, student, LoRA budget, and evaluation protocol; generation-side settings follow each source recipe. *GKD* (Agarwal et al., 2024) performs on-policy token-level forward-KL distillation on free-running student rollouts sampled at temperature 1.0, mixed at the same rate λ with cross-entropy on teacher chain-of-thought samples, under a one-shot chain-of-thought prompt. *CoT-SFT* (Ho et al., 2023; Magister et al., 2023) draws one teacher chain-of-thought rollout per question at temperature 0.7 and fine-tunes offline for two epochs. *D-RPC* (Yang et al., 2026) constrains the teacher with a bank of reusable reasoning paths and distills the resulting structured rationales, using the templates from its appendix. *Scaffold* (Wen et al., 2026) distills the teacher's reasoning as a sequence of discrete semantic signals with signal-conditioned generation, following the official pipeline. *Think-SFT* is the long-thinking SFT baseline of Wen et al. (2026): one extended-thinking teacher trace per question, distilled by SFT. *SGFT* (Bi et al., 2025) distills teacher-written solution guidance followed by the guided solution.

Table 5: Results with the Qwen3-1.7B-Base student and (a) Qwen3-4B and (b) Qwen3-8B teachers. Protocol and layout as in Table 1.

(a) Teacher Qwen3-4B

	StrategyQA		CSQA		ARC-C		GSM8K		AQuA		Average	
Method	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
ACTED (ours)	**63.3** ± 1.7	**85.4** ± 2.2	**74.6** ± 0.7	**85.9** ± 0.7	**82.9** ± 0.3	**92.0** ± 0.5	74.2 ± 0.5	87.4 ± 0.6	57.9 ± 2.1	**81.0** ± 0.4	70.6	**86.3**
GKD	60.6 ± 1.2	78.8 ± 0.3	**74.6** ± 1.5	85.5 ± 0.4	82.5 ± 0.4	91.5 ± 0.6	78.5 ± 1.4	**90.3**	**62.9** ± 2.2	**81.0** ± 1.4	71.8	85.4
CoT-SFT	60.5 ± 1.2	81.6 ± 1.4	74.5 ± 0.4	85.0 ± 0.5	**82.9** ± 0.4	91.8 ± 0.4	**78.8** ± 0.3	90.1 ± 0.4	62.6 ± 2.4	80.5 ± 2.0	**71.9**	85.8
D-RPC	54.1 ± 2.2	65.7 ± 2.5	70.0 ± 1.3	85.1 ± 0.3	78.8 ± 0.7	91.2 ± 0.2	70.0 ± 0.6	88.9 ± 0.3	53.3 ± 1.0	78.1 ± 1.5	65.3	81.8
Scaffold	29.5	62.3	48.5	82.9	56.7	85.6	31.2	72.4	9.1	18.1	35.0	64.3
Think-SFT	24.9	50.7	37.6	69.8	25.0	58.2	28.2	66.1	7.1	13.4	24.6	51.6
SGFT	6.1	19.5	29.1	64.5	23.1	53.8	14.5	41.9	6.7	17.3	15.9	39.4
Δ vs. best	+2.7	+3.8	0.0	+0.4	0.0	+0.2	−4.6	−2.9	−5.0	0.0	−1.3	+0.5
Δ vs. worst	+57.2	+65.9	+45.5	+21.4	+59.8	+38.2	+59.7	+45.5	+51.2	+67.6	+54.7	+46.9
Δ vs. avg.	+24.0	+25.6	+18.9	+7.1	+24.7	+13.3	+24.0	+12.5	+24.3	+32.9	+23.2	+18.2

(b) Teacher Qwen3-8B

	StrategyQA		CSQA		ARC-C		GSM8K		AQuA		Average	
Method	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
ACTED (ours)	**64.4** ± 0.8	**83.1** ± 1.7	72.5 ± 1.1	84.4 ± 0.9	**82.8** ± 1.0	92.8 ± 0.9	73.9 ± 0.7	88.4 ± 0.5	59.2 ± 0.6	77.6 ± 0.4	70.6	85.3
GKD	60.2 ± 1.6	76.3 ± 0.8	74.5 ± 1.1	86.7 ± 0.7	82.1 ± 0.8	92.8 ± 0.4	76.7 ± 1.7	**88.5** ± 0.5	61.2 ± 1.5	81.9 ± 1.0	70.9	85.2
CoT-SFT	60.9 ± 0.5	79.5 ± 1.1	**75.2** ± 0.4	88.2 ± 0.4	82.2 ± 0.1	**92.9** ± 0.7	**77.3** ± 0.5	88.0 ± 0.6	**63.4** ± 1.2	**82.4** ± 0.2	**71.8**	**86.2**
D-RPC	55.4 ± 0.7	73.3 ± 0.9	70.7 ± 1.4	**88.4** ± 0.5	80.9 ± 0.7	92.0 ± 0.4	70.7 ± 0.6	88.4 ± 0.7	55.0 ± 0.4	78.9 ± 0.8	66.5	84.2
Scaffold	17.8	47.3	42.3	77.5	39.3	75.0	27.5	70.0	6.3	18.9	26.6	57.7
Think-SFT	12.4	31.1	30.4	63.6	17.9	40.4	26.4	61.2	3.5	8.7	18.1	41.0
SGFT	0.9	4.4	12.9	40.6	2.3	21.9	22.1	45.3	2.4	13.4	8.1	25.1
Δ vs. best	+3.5	+3.6	−2.7	−4.0	+0.6	−0.1	−3.4	−0.1	−4.2	−4.8	−1.2	−0.9
Δ vs. worst	+63.5	+78.7	+59.6	+43.8	+80.5	+70.9	+51.8	+43.1	+56.8	+68.9	+62.5	+60.2
Δ vs. avg.	+29.8	+31.1	+21.5	+10.2	+32.0	+23.6	+23.8	+14.8	+27.2	+30.2	+26.9	+22.1

D Results with the 1.7B student

Table 5 report results for the Qwen3-1.7B-Base student with Qwen3-4B and Qwen3-8B teachers. The protocol, metrics, method blocks, highlighting, and seed conventions match Table 1.

E Motivation experiment details

This section gives the setups of the three experiments in Fig. 1 and their results on the five main datasets.

Common setup. The teacher is Qwen3-1.7B. The students are the Qwen3-0.6B-Base models distilled from it on each dataset by the CoT-SFT and GKD baselines of Section 5.1, which learn from chain-of-thought traces without actions, and by ACTED, all from the main teacher-student pair. Every experiment uses a seeded sample of 200 test problems per dataset, and the teacher's greedy trajectory in the action-execution format on each of them. All students run under one instruction template, the one ACTED is trained with, and under the same format-constrained decoding. At step i, a student first decides whether to answer, by comparing the probability of the line prefixes `Final Answer:` and `Action i:`. If it continues, it writes the action name after the forced prefix `Action i: <<`, and the execution line is then written after `Execution:`; the final-answer line is always written by the student. Constraining the line types keeps the comparison about which decisions a student makes rather than whether it has learned the trajectory syntax. Answers are checked as in the main evaluation, with option-aware matching on AQuA. Confidence intervals are 95% percentile intervals from 2,000 bootstrap resamples of problems, and differences between students are paired over problems.

Supervisable decisions. The teacher-action NLL is computed on the teacher's well-formed, correct trajectories: for every step i, the student scores the name tokens of the teacher's action a_i given the teacher's history h_i and the prefix `Action i: <<`, and the per-token NLL is averaged within each problem and then across problems. Two controls are scored under the same contexts: the NLL of the teacher's execution e_i, and a pairwise test that compares the true action with the teacher's action for the next step (or the previous one, for the last step). In the decision loop, the student decides

Table 6: Decision experiment (200 test problems per dataset). a-NLL and e-NLL: per-token NLL of the teacher's actions and executions on its well-formed, correct trajectories. Pair: how often the student ranks the teacher's true next action above the teacher's action for a neighboring step, averaged per problem. Sampled: accuracy when the student's sampled decisions ($T = 1$, five episodes per problem) are executed by the teacher, averaged over the episodes; Greedy: the same with greedy decisions.

Dataset	Student	a-NLL	e-NLL	Pair	Sampled	Greedy
StrategyQA	CoT-SFT	1.40	0.81	0.77	0.59	0.62
	GKD	1.34	0.79	0.74	0.57	0.59
	ACTED	0.81	0.75	0.89	0.60	0.51
CSQA	CoT-SFT	0.62	0.94	0.90	0.67	0.69
	GKD	0.58	0.94	0.91	0.68	0.71
	ACTED	0.20	0.68	0.97	0.66	0.67
ARC-C	CoT-SFT	0.83	0.62	0.91	0.80	0.79
	GKD	0.73	0.62	0.91	0.79	0.82
	ACTED	0.46	0.51	0.96	0.81	0.80
GSM8K	CoT-SFT	0.73	0.18	0.85	0.63	0.69
	GKD	0.70	0.16	0.85	0.64	0.67
	ACTED	0.39	0.14	0.92	0.69	0.69
AQuA	CoT-SFT	0.66	0.20	0.88	0.57	0.56
	GKD	0.63	0.20	0.91	0.57	0.58
	ACTED	0.42	0.17	0.94	0.61	0.63

Table 7: Uncertainty experiment (Qwen3-1.7B teacher, $K = 16$ sampled actions per step, up to three steps per problem). Steps: number of scored steps; Distinct a: mean number of distinct actions among the K samples. Entropies are in nats per execution (per action for $\mathcal{H}(A|H)$), averaged over steps. Reduction: $[\mathcal{H}(E|H) - \mathcal{H}(E|H, A)]/\mathcal{H}(E|H)$ from the step averages, with a 95% bootstrap interval over problems.

Dataset	Steps	Distinct a	$\mathcal{H}(A\|H)$	$\mathcal{H}(E\|H)$	$\mathcal{H}(E\|H, A)$	Reduction (%) [95% CI]
StrategyQA	292	7.2	2.50	7.82	6.85	12.4 [11.4, 13.4]
CSQA	299	2.3	0.55	3.81	3.60	5.7 [4.0, 7.4]
ARC-C	293	4.5	1.38	3.97	3.50	11.8 [10.4, 13.3]
GSM8K	264	3.2	0.85	1.67	1.40	16.1 [13.0, 19.9]
AQuA	270	4.2	1.30	3.17	2.79	11.9 [10.1, 13.9]

every action and whether to stop, the teacher writes each execution under its own prompt given the trajectory so far, and the result is returned to the student. Decisions are sampled at temperature 1 in five episodes per problem, whose accuracies are averaged, and are also taken greedily once. Table 6 gives the full results.

Reduced execution entropy. For each of the first 100 problems whose greedy teacher trajectory is well-formed and correct, and each of its first three steps, the history h_i is the teacher's own prefix. We sample $K = 16$ actions $a^1, \ldots, a^K \sim \pi^{\mathrm{T}}(\cdot|h_i)$ at temperature 1, and for each action one execution $e^k \sim \pi^{\mathrm{T}}(\cdot|h_i, a^k)$. Every execution is then scored under every sampled action, giving $L_{jk} = \log p^{\mathrm{T}}(e^k|h_i, a^j)$, where the score includes the execution's terminating newline. The estimates are

$$\hat{\mathcal{H}}(E|H, A) = -\frac{1}{K}\sum_k L_{kk}, \qquad \hat{\mathcal{H}}(E|H) = -\frac{1}{K}\sum_k \log \frac{1}{K}\sum_j e^{L_{jk}},$$

where the inner average approximates the marginal, $p^{\mathrm{T}}(e|h_i) \approx \frac{1}{K}\sum_j p^{\mathrm{T}}(e|h_i, a^j)$, and both estimates are averaged over steps. The action entropy in Table 7 is estimated from the same samples as $\hat{\mathcal{H}}(A|H) = -\frac{1}{K}\sum_k \log p^{\mathrm{T}}(a^k|h_i)$. Their difference is an estimate of the conditional mutual information $I(A; E|H)$; because each execution's own action is part of the mixture, it is conservative, bounded above by $\log K$. We report it relative to the execution entropy, $[\hat{\mathcal{H}}(E|H) -$

Table 8: Robustness experiment on the common problem sets. v1 inserts two action-execution pairs from other problems at the start of the solution and lets the student re-decide its action at the probe step $i = 2$; v2 appends the student's original action a_i to the same perturbed history. Accuracy: final-answer accuracy of the greedy continuation. NLL of e_i: per-token NLL of the student's own clean execution at the probe step. Action changed: the fraction of problems where the student's action under v1 differs as a string, after lowercasing and whitespace normalization, from its clean action, so paraphrases count as changes.

		Accuracy			NLL of e_i			Action
Dataset	Student	clean	V1	V2	clean	V1	V2	changed
StrategyQA	CoT-SFT	0.54	0.53	0.60	0.360	1.147	0.436	0.97
	GKD	0.54	0.54	0.57	0.353	0.811	0.387	0.96
	ACTED	0.54	0.52	0.56	0.360	0.863	0.409	0.92
CSQA	CoT-SFT	0.61	0.57	0.58	0.362	0.792	0.557	0.90
	GKD	0.59	0.61	0.62	0.383	0.650	0.493	0.85
	ACTED	0.62	0.57	0.61	0.401	1.319	0.519	0.89
ARC-C	CoT-SFT	0.67	0.59	0.63	0.273	0.575	0.342	0.92
	GKD	0.70	0.66	0.68	0.292	0.495	0.350	0.86
	ACTED	0.69	0.66	0.65	0.289	0.605	0.351	0.88
GSM8K	CoT-SFT	0.59	0.45	0.49	0.054	0.277	0.103	0.49
	GKD	0.56	0.44	0.50	0.074	0.273	0.120	0.49
	ACTED	0.54	0.43	0.47	0.067	0.279	0.112	0.45
AQuA	CoT-SFT	0.55	0.46	0.46	0.098	0.358	0.162	0.65
	GKD	0.53	0.48	0.49	0.133	0.294	0.184	0.68
	ACTED	0.51	0.54	0.54	0.102	0.293	0.155	0.55

$\hat{\mathcal{H}}(E|H, A)]/\hat{\mathcal{H}}(E|H)$, taking the ratio of the two step averages, and its interval resamples problems. Table 7 gives the estimates.

Robustness is enhanced. Each student first rolls out its own clean trajectory under constrained greedy decoding. The probe step is its second action step, $i = 2$, so that exactly one clean step separates the inserted distractors from the probe for every student. The perturbed history inserts, at the start of the solution, two action-execution pairs, each drawn from the teacher's trajectory of a different problem (the same donors for all students), and renumbers all actions. v1 continues from the perturbed history: the student re-decides the action, with the next line forced to be an action line, and then continues until the final answer. v2 appends the student's original action a_i to the perturbed history and continues from there. The NLL of the clean execution e_i is scored under the clean context h_i, a_i, under the perturbed history followed by the re-decided action (v1), and under the perturbed history followed by a_i (v2). Table 8 gives the full results.

F Bottleneck ablation details

This section details the ablations of Appendix G and reports their results per condition and per dataset.

Construction. Every perturbation reuses the clean rollouts, the problems, and the donor draws of the bottleneck experiment in Appendix E. For each problem, the two pairs of that experiment are drawn first, with the same random seed, and eight further donor steps from distinct problems of the same dataset are drawn afterwards, so the default condition ($M = 2$) reproduces the earlier results exactly. The number ablation prepends further pairs from these draws, so that each larger set contains the smaller one and the pairs nearest to the student's history are shared by all levels. The location ablation uses the single pair nearest to the history ($M = 1$) and probes the third step on the problems whose clean rollout has at least three steps. Inside the solution the pair is numbered as an ordinary step, so the probe is Action 4 at every in-solution location; in the problem statement it is written as a sentence, `Note: <action>. <execution>`, before the answer options or at the end of the problem, and the probe stays Action 3. The length ablation keeps the two actions of the default pairs and changes their executions: the short level keeps the first $\min(5, \lfloor n/2 \rfloor)$ words of an n-word

execution, and the extended levels concatenate two or four executions, taken as a contiguous window of the donor trajectory that ends at the drawn step, continued forward over the donor's non-final steps if the window is too short, and then over non-final steps of other same-dataset donors. The domain ablation draws the two pairs uniformly from the pooled donors of the other datasets of the same domain, or of the other domain. The renumbering controls insert nothing and shift the action indices by o, so that the probe carries the same index as with $M = o$ inserted pairs (or with a single in-solution pair for the third-step probe).

Measurement. For each condition, the NLL of the student's clean execution e_i is scored under the clean history with a_i, under the perturbed history with the re-decided action (v1), and under the perturbed history with a_i (v2); v1 and v2 are then continued greedily to the final answer. All numbers are computed on the problems eligible for both students and averaged over the five main datasets; recovery is the ratio of these averages, $1 - \Delta\mathrm{NLL}_{\mathrm{V2}}/\Delta\mathrm{NLL}_{\mathrm{V1}}$. Intervals are 95% percentile intervals from 2,000 bootstrap resamples of problems within each dataset, and conditions that share a problem set share their resamples, so student differences and condition contrasts are paired.

G Analysis: How the Action Bottlenecks the History

Figure 1(c) perturbs the history in one way: two action-execution pairs from other problems of the same dataset, inserted at the start of the solution. This section varies the perturbation along four factors, the number of inserted pairs, their location, their length, and the domain they come from, to test how far the action shields the execution from irrelevant history.

Setup. We use the chain-of-thought (CoT-SFT) and action-execution (ACTED) students of Fig. 1 on the five main datasets, with the same 200 test problems, template, and constrained greedy decoding as in Appendix E, and each student's clean rollout from that experiment. Each perturbation modifies the history before a probe step, the student's second step unless stated otherwise. Under v1 the student re-decides the action at the probe and continues to its final answer; under v2 it is given its original action a_i instead. We vary one factor at a time around the setting of Fig. 1(c): (i) the number of inserted pairs, $M \in \{1, 2, 4, 8\}$, where each larger set keeps the smaller one and adds pairs farther from the probe; (ii) the location of a single pair, placed in the problem statement, before the first step, before the second step, or right before the probe, with the third step as the probe; (iii) the length of the two pairs, whose executions are cut to their first few words or extended to two or four executions, taken as a contiguous window of the donor trajectory and, if it is too short, continued with steps of other donors, so that no answer-bearing final step is added; (iv) the domain of the two pairs, drawn from the same dataset, from the other datasets of the same domain, or from the other domain, where GSM8K and AQuA form the math domain and StrategyQA, CSQA, and ARC-C the question-answering domain. Because inserting a pair into the solution also renumbers the later actions, each number of pairs and the in-solution locations have a control that renumbers the actions in the same way without inserting anything. All quantities are averaged over the five datasets on the problems eligible for both students.

Restoring the action removes most of the damage under every perturbation. Across all insertion conditions, giving back the original action removes 61–84% of the V1 NLL increase for the chain-of-thought student and 66–89% for the action-execution student (Fig. 2, middle row). The renumbering controls change the next action on up to 53% of the problems, yet leave almost no trace once the action is restored (at most 0.007 nats per token), so what remains under V2 comes from the inserted content, not from the shifted indices.

More and longer noise leaks past the action. As the number of pairs grows from one to eight, the V1 increase rises from 0.35 to 0.58 nats per token for the chain-of-thought student, while the part that survives the restored action grows from 0.07 to 0.18, so the action removes a shrinking share, from 81% to 69% (Fig. 2a). Longer executions act the same way at a fixed number of pairs: extending them from their natural length to four executions raises this residual from 0.09 to 0.19 and lowers the removed share from 77% to 61%, whereas cutting them short changes little (Fig. 2c). The bottleneck is therefore partial: the action absorbs most of the irrelevant history, but a direct path from the history to the execution grows with the amount of foreign text.

Noise right before the step leaks most. A single pair in the problem statement or before the first step barely disturbs the third step, leaving a residual of 0.04 nats per token under V2 (Fig. 2b). The

Table 9: Bottleneck ablations, averaged over the five main datasets. Tokens: context tokens added by the perturbation. Changed: share of problems whose re-decided action differs from a_i as a string. Stop: share of problems where the student would have answered instead of writing an action at the probe (the V1 line is forced to be an action). ΔNLL: increase in the per-token NLL of e_i over the clean history under v1 and v2. Recovery: share of the v1 increase removed under v2. ΔAcc: change in final-answer accuracy (points) under v1 and v2. The location block and its control use the third step as the probe.

Perturbation	Student	Tokens	Changed	Stop	ΔNLL V1	ΔNLL V2	Recovery	ΔAcc V1	ΔAcc V2
$M = 1$	CoT-SFT	39	0.74	0.05	0.349	0.066	0.81	−3.3	−2.9
	ACTED	39	0.60	0.00	0.324	0.045	0.86	−2.4	−1.4
$M = 2$ (Fig. 1c)	CoT-SFT	78	0.78	0.29	0.396	0.091	0.77	−7.1	−4.1
	ACTED	78	0.74	0.05	0.426	0.066	0.85	−3.2	−1.4
$M = 4$	CoT-SFT	156	0.83	0.35	0.453	0.126	0.72	−8.2	−3.3
	ACTED	156	0.77	0.08	0.468	0.096	0.79	−3.9	−4.3
$M = 8$	CoT-SFT	314	0.88	0.19	0.581	0.179	0.69	−11.8	−6.7
	ACTED	314	0.79	0.05	0.526	0.130	0.75	−8.5	−5.7
Renumber by 1	CoT-SFT	0	0.53	0.05	0.163	0.006	0.96	0.2	−0.5
	ACTED	0	0.24	0.00	0.065	0.005	0.92	0.3	−0.2
Renumber by 2	CoT-SFT	0	0.42	0.10	0.117	0.005	0.96	−1.5	−1.5
	ACTED	0	0.20	0.00	0.051	0.003	0.94	0.3	−0.3
Renumber by 4	CoT-SFT	0	0.38	0.06	0.101	0.006	0.94	−0.4	−0.1
	ACTED	0	0.21	0.00	0.050	0.004	0.92	−0.1	−0.6
Renumber by 8	CoT-SFT	0	0.40	0.05	0.106	0.007	0.94	−1.0	−0.7
	ACTED	0	0.23	0.00	0.054	0.004	0.92	−0.5	−0.5
In the problem	CoT-SFT	33	0.36	0.02	0.152	0.036	0.76	−2.5	−2.3
	ACTED	33	0.27	0.00	0.097	0.023	0.76	−2.9	−3.4
Before step 1	CoT-SFT	39	0.43	0.34	0.138	0.042	0.70	−2.7	−2.0
	ACTED	39	0.31	0.12	0.103	0.031	0.70	−1.9	−1.6
Before step 2	CoT-SFT	39	0.52	0.37	0.214	0.067	0.69	−2.0	−1.6
	ACTED	39	0.41	0.14	0.143	0.049	0.66	−2.5	−2.6
Before the probe	CoT-SFT	39	0.76	0.53	0.614	0.195	0.68	−11.3	−6.1
	ACTED	39	0.75	0.31	0.459	0.127	0.72	−10.9	−5.5
Renumber by 1	CoT-SFT	0	0.23	0.27	0.038	0.005	0.88	−0.0	−0.2
	ACTED	0	0.13	0.06	0.030	0.003	0.90	−0.4	−0.6
Short executions	CoT-SFT	43	0.78	0.37	0.344	0.083	0.76	−3.0	−1.6
	ACTED	43	0.70	0.03	0.349	0.055	0.84	−1.2	−1.9
×2 executions	CoT-SFT	130	0.81	0.32	0.450	0.132	0.71	−6.6	−4.0
	ACTED	130	0.73	0.06	0.451	0.092	0.80	−2.2	−2.4
×4 executions	CoT-SFT	234	0.81	0.35	0.476	0.188	0.61	−9.6	−5.0
	ACTED	234	0.71	0.05	0.455	0.125	0.73	−4.5	−1.8
Same domain	CoT-SFT	84	0.77	0.26	0.356	0.080	0.77	−6.5	−4.1
	ACTED	84	0.70	0.03	0.372	0.059	0.84	−2.0	−2.8
Other domain	CoT-SFT	84	0.73	0.23	0.368	0.059	0.84	−1.6	−2.3
	ACTED	84	0.62	0.03	0.352	0.038	0.89	−1.0	0.6

same pair placed right before the probe changes the chain-of-thought student's next action on 76% of the problems, raises its v1 NLL increase to 0.61, and costs 11 accuracy points; restoring the action roughly halves this loss, to 6 points, but leaves a residual of 0.20. Recency thus governs both the decision of the next action and the leak around it.

Unrelated domains disturb less. Pairs from the other datasets of the same domain act much like pairs from the same dataset, whereas pairs from the other domain leave a smaller residual (0.06 against 0.09 nats per token for the chain-of-thought student) and cost fewer accuracy points (2 against 7; Fig. 2d). Irrelevant history thus hurts most when it resembles the task at hand.

The action-execution student leaks less. Under every insertion, the residual that survives the restored action is lower for the action-execution student than for the chain-of-thought student, significantly so in all conditions except when the single pair sits before the first or the second step, and the gap widens with the amount of noise, from 0.02 nats per token with one pair to 0.05 with eight pairs

Table 10: Bottleneck ablations per dataset: increase in the per-token NLL of e_i under v1 / share of that increase removed under v2.

Perturbation	Student	StrategyQA	CSQA	ARC-C	GSM8K	AQuA
$M = 1$	CoT-SFT	0.69 / 0.91	0.38 / 0.63	0.30 / 0.84	0.17 / 0.79	0.21 / 0.81
	ACTED	0.45 / 0.91	0.64 / 0.88	0.23 / 0.82	0.16 / 0.80	0.13 / 0.76
$M = 2$ (Fig. 1c)	CoT-SFT	0.77 / 0.90	0.43 / 0.55	0.30 / 0.77	0.22 / 0.77	0.25 / 0.75
	ACTED	0.50 / 0.90	0.92 / 0.87	0.31 / 0.80	0.21 / 0.78	0.19 / 0.71
$M = 4$	CoT-SFT	0.80 / 0.88	0.52 / 0.48	0.35 / 0.66	0.29 / 0.78	0.30 / 0.75
	ACTED	0.51 / 0.87	0.98 / 0.81	0.36 / 0.76	0.23 / 0.76	0.26 / 0.68
$M = 8$	CoT-SFT	0.91 / 0.86	0.68 / 0.44	0.51 / 0.61	0.43 / 0.80	0.38 / 0.74
	ACTED	0.63 / 0.85	1.01 / 0.75	0.39 / 0.71	0.29 / 0.77	0.31 / 0.63
Renumber by 1	CoT-SFT	0.31 / 0.97	0.19 / 0.95	0.17 / 0.95	0.06 / 0.97	0.08 / 0.96
	ACTED	0.16 / 0.96	0.02 / 0.62	0.07 / 0.90	0.04 / 0.96	0.04 / 0.90
Renumber by 2	CoT-SFT	0.19 / 0.97	0.12 / 0.93	0.17 / 0.95	0.04 / 0.98	0.06 / 0.95
	ACTED	0.10 / 0.97	0.02 / 0.69	0.07 / 0.92	0.04 / 0.99	0.03 / 0.93
Renumber by 4	CoT-SFT	0.14 / 0.96	0.09 / 0.90	0.18 / 0.94	0.03 / 0.96	0.07 / 0.95
	ACTED	0.12 / 0.98	0.01 / 0.35	0.06 / 0.89	0.04 / 0.95	0.02 / 0.85
Renumber by 8	CoT-SFT	0.16 / 0.96	0.09 / 0.89	0.19 / 0.94	0.04 / 0.97	0.06 / 0.92
	ACTED	0.13 / 0.97	0.01 / 0.32	0.06 / 0.90	0.05 / 0.95	0.03 / 0.84
In the problem	CoT-SFT	0.35 / 0.90	0.10 / 0.13	0.11 / 0.64	0.07 / 0.92	0.14 / 0.87
	ACTED	0.19 / 0.90	0.06 / 0.25	0.13 / 0.79	0.04 / 0.80	0.06 / 0.79
Before step 1	CoT-SFT	0.30 / 0.83	0.08 / 0.10	0.13 / 0.75	0.08 / 0.73	0.10 / 0.70
	ACTED	0.15 / 0.85	0.04 / 0.22	0.11 / 0.49	0.11 / 0.80	0.11 / 0.80
Before step 2	CoT-SFT	0.45 / 0.80	0.12 / 0.20	0.19 / 0.67	0.17 / 0.76	0.14 / 0.66
	ACTED	0.18 / 0.82	0.07 / 0.38	0.16 / 0.44	0.15 / 0.74	0.15 / 0.76
Before the probe	CoT-SFT	0.80 / 0.76	0.29 / 0.30	0.87 / 0.62	0.46 / 0.83	0.65 / 0.74
	ACTED	0.40 / 0.81	0.29 / 0.70	0.49 / 0.54	0.44 / 0.80	0.67 / 0.77
Renumber by 1	CoT-SFT	0.08 / 0.94	0.02 / 0.47	0.05 / 0.90	0.02 / 0.93	0.03 / 0.86
	ACTED	0.05 / 0.96	0.00 / 0.48	0.01 / 0.55	0.01 / 0.79	0.07 / 0.95
Short executions	CoT-SFT	0.58 / 0.88	0.42 / 0.61	0.34 / 0.74	0.17 / 0.73	0.21 / 0.76
	ACTED	0.40 / 0.89	0.76 / 0.87	0.29 / 0.79	0.14 / 0.77	0.16 / 0.74
$\times 2$ executions	CoT-SFT	0.92 / 0.85	0.46 / 0.44	0.31 / 0.63	0.25 / 0.75	0.30 / 0.73
	ACTED	0.57 / 0.85	0.95 / 0.82	0.34 / 0.78	0.21 / 0.75	0.19 / 0.61
$\times 4$ executions	CoT-SFT	0.88 / 0.75	0.54 / 0.38	0.37 / 0.45	0.26 / 0.70	0.33 / 0.68
	ACTED	0.58 / 0.80	0.90 / 0.73	0.38 / 0.71	0.21 / 0.72	0.21 / 0.54
Same domain	CoT-SFT	0.64 / 0.90	0.36 / 0.60	0.30 / 0.78	0.20 / 0.72	0.27 / 0.74
	ACTED	0.44 / 0.91	0.78 / 0.87	0.27 / 0.83	0.17 / 0.70	0.19 / 0.71
Other domain	CoT-SFT	0.81 / 0.91	0.36 / 0.72	0.35 / 0.78	0.13 / 0.85	0.19 / 0.87
	ACTED	0.53 / 0.91	0.74 / 0.91	0.28 / 0.81	0.08 / 0.85	0.12 / 0.91

and 0.06 with four executions per pair. Its action is also more stable: renumbering alone changes it on 20–24% of the problems, against 38–53% for the chain-of-thought student. When it re-decides the action, it loses fewer accuracy points under larger perturbations, for example 4 against 8 points with four pairs and 5 against 10 with four executions per pair (significant for two and four pairs, both extended lengths, and same-domain pairs), although both students lose about 11 points when the pair sits right before the probe. The action-execution student's execution thus relies more on its stated action and less on the rest of the history, the behavior that distilling the joint action-execution distribution is meant to instill.

H PROMPT TEMPLATES

This section gives the two prompts under which the teacher writes reasoning traces: the action-execution prompt of ACTED and the standard chain-of-thought prompt of the GKD and CoT-SFT baselines. Each prompt consists of an instruction, a one-shot exemplar, and the question, and the two prompts differ only in the instruction and the exemplar (Fig. 3). The teacher receives the instruction as the system message and the exemplar and question as the user message, rendered with its chat template with extended thinking disabled. ACTED and GKD train and evaluate the student on the same text as a single plain-text prompt, the instruction followed by the user message, without a chat template (Appendix B). CoT-SFT uses the chain-of-thought prompt only to generate the teacher's traces; its student is fine-tuned and evaluated on the question alone, as `Problem: {question}` followed by a `Solution:` line. The motivation and bottleneck experiments run every student under the action-execution prompt (Appendices E and F).

Placeholders. In the instruction, `{answer_format}` is the dataset's answer-format line (Table 11). In the user message, `{example}` is the dataset's exemplar (Figs. 4 and 5), and `{question}` is the problem text; on the multiple-choice datasets, the lettered options follow it, one per line, under a `Choices:` header (`Options:` on AQuA). In both prompts the answer-format line follows a generic `Final Answer:` instruction, and every exemplar ends with a line that satisfies both, such as `Final Answer: The answer is B`.

Exemplars. Each dataset has one exemplar problem, solved once in each format (Figs. 4 and 5). The action-execution exemplar names an abstract action on each `Action` line and applies it on the following `Execution` line, and the chain-of-thought exemplar solves the same problem in plain lines with the same final-answer line.

Table 11: Answer-format lines substituted for `{answer_format}` in both instructions.

Dataset	Answer-format line
StrategyQA	`End with a final line of the form: The answer is yes (or: The answer is no)`
CSQA, ARC-C	`End with a final line of the form: The answer is X (X = the letter of the correct choice)`
GSM8K	`State the final line as: Final Answer: N (N = the final integer, no units or commas)`
AQuA	`End with a final line of the form: The answer is X (X = the letter of the correct option)`

(a) Action–execution instruction (ACTED)

```
You are an expert problem solver. Solve the problem as a trajectory of
↪  abstract reasoning actions and their concrete executions.

Format, exactly:
Action 1: <<action name>>
Execution: <one concrete reasoning step applying that action>
Action 2: <<action name>>
Execution: ...
...
Final Answer: <the answer>

Rules:
- An action is a short ABSTRACT description of a reasoning operation
↪  (e.g. <<identify given quantities>>, <<set up an equation>>, <<case
↪  analysis>>) -- never mention the problem's specific numbers or
↪  objects inside <<...>>.
- The Execution after each Action is the CONCRETE application of that
↪  abstract action to this problem: actual quantities, equations, and
↪  calculations.
- Think before acting: choose each next action based on the state of
↪  the solution so far.
- Use as many Action/Execution pairs as needed. Plain text only.
- {answer_format}
```

(b) Chain-of-thought instruction (GKD and CoT-SFT)

```
You are an expert problem solver. Think step by step. Plain text only.
End with a final line of the form: Final Answer: <the answer>
{answer_format}
```

(c) User message (both prompts)

```
{example}

Problem: {question}
Solution:
```

Figure 3: Prompt templates. The teacher receives the instruction, (a) or (b), as the system message and (c) as the user message; the student receives the instruction followed by (c) as one plain-text prompt. Placeholders in braces are filled per dataset and question, and the user message ends with a newline after `Solution:`. ↪ marks a line wrapped for display.

Action–execution *Chain-of-thought*

GSM8K

```
Problem: A bakery sells muffins for $3 each.
↪  Ken buys 4 muffins and pays with a $20
↪  bill. How much change does he get?
Solution:
Action 1: <<compute a total cost>>
Execution: The muffins cost 4 * 3 = 12
↪  dollars.
Action 2: <<compute the difference>>
Execution: The change is 20 - 12 = 8
↪  dollars.
Final Answer: 8
```

```
Problem: A bakery sells muffins for $3 each.
↪  Ken buys 4 muffins and pays with a $20
↪  bill. How much change does he get?
Solution:
The muffins cost 4 * 3 = 12 dollars.
The change is 20 - 12 = 8 dollars.
Final Answer: 8
```

AQuA

```
Problem: A train travels 120 km in 1.5
↪  hours. At the same speed, how long does
↪  it take to travel 200 km?
Options:
A. 2 hours
B. 2.5 hours
C. 3 hours
D. 3.5 hours
E. 4 hours
Solution:
Action 1: <<compute a unit rate>>
Execution: The speed is 120 / 1.5 = 80 km
↪  per hour.
Action 2: <<apply the rate to the new
↪  quantity>>
Execution: The time for 200 km is 200 / 80 =
↪  2.5 hours.
Action 3: <<match the result to an option>>
Execution: 2.5 hours corresponds to option
↪  B.
Final Answer: The answer is B
```

```
Problem: A train travels 120 km in 1.5
↪  hours. At the same speed, how long does
↪  it take to travel 200 km?
Options:
A. 2 hours
B. 2.5 hours
C. 3 hours
D. 3.5 hours
E. 4 hours
Solution:
The speed is 120 / 1.5 = 80 km per hour.
The time for 200 km is 200 / 80 = 2.5 hours,
↪  option B.
Final Answer: The answer is B
```

Figure 4: One-shot exemplars for the mathematical datasets, in the action–execution format (left) and the chain-of-thought format (right). ↪ marks a line wrapped for display.

Action–execution *Chain-of-thought*

StrategyQA

```
Problem: Could a snail cross the English
↪  Channel in a day?
Solution:
Action 1: <<recall a relevant quantitative
↪  fact>>
Execution: A garden snail moves at roughly
↪  0.05 km per hour.
Action 2: <<recall the scale of the target>>
Execution: The English Channel is about 33
↪  km across at its narrowest.
Action 3: <<compare capability against
↪  requirement>>
Execution: In 24 hours a snail covers about
↪  1.2 km, far less than 33 km.
Final Answer: The answer is no
```

```
Problem: Could a snail cross the English
↪  Channel in a day?
Solution:
A garden snail moves at roughly 0.05 km per
↪  hour.
The English Channel is about 33 km across.
In 24 hours a snail covers about 1.2 km, far
↪  less than 33 km.
Final Answer: The answer is no
```

CSQA

```
Problem: Where would you most likely keep
↪  leftover soup to eat tomorrow?
Choices:
A. cupboard
B. refrigerator
C. oven
D. backpack
E. dishwasher
Solution:
Action 1: <<identify the goal of the
↪  action>>
Execution: The soup must stay fresh and safe
↪  to eat until tomorrow.
Action 2: <<recall a relevant everyday
↪  fact>>
Execution: Cooked food stays fresh when it
↪  is kept cold.
Action 3: <<match the requirement to an
↪  option>>
Execution: Of the choices, only a
↪  refrigerator keeps food cold, which is
↪  option B.
Final Answer: The answer is B
```

```
Problem: Where would you most likely keep
↪  leftover soup to eat tomorrow?
Choices:
A. cupboard
B. refrigerator
C. oven
D. backpack
E. dishwasher
Solution:
The soup must stay fresh and safe to eat
↪  until tomorrow.
Cooked food stays fresh when it is kept
↪  cold.
Of the choices, only a refrigerator keeps
↪  food cold, option B.
Final Answer: The answer is B
```

ARC-C

```
Problem: Which process in the water cycle
↪  changes water vapor into liquid water?
Choices:
A. evaporation
B. condensation
C. precipitation
D. runoff
Solution:
Action 1: <<identify what the question asks
↪  for>>
Execution: We need the process that turns
↪  water vapor, a gas, into liquid water.
Action 2: <<recall the relevant science
↪  concept>>
Execution: Condensation is the change of a
↪  gas into a liquid; evaporation is the
↪  reverse change.
Action 3: <<match the concept to an option>>
Execution: Condensation corresponds to
↪  option B.
Final Answer: The answer is B
```

```
Problem: Which process in the water cycle
↪  changes water vapor into liquid water?
Choices:
A. evaporation
B. condensation
C. precipitation
D. runoff
Solution:
The question asks for the process that turns
↪  water vapor, a gas, into liquid water.
Condensation is the change of a gas into a
↪  liquid.
That is option B.
Final Answer: The answer is B
```

Figure 5: One-shot exemplars for the question-answering datasets, in the action-execution format (left) and the chain-of-thought format (right). ↪ marks a line wrapped for display.