AGENT AS COACH: LEARNING TO TEACH FROM STUDENT BEHAVIOR

Anonymous authors
Paper under double-blind review

ABSTRACT

Rubrics have become the basis for both evaluation and training feedback on tasks without a single correct answer, yet they describe what a good answer looks like rather than how a student should produce one. Experience extracted from evaluated student behavior can provide this actionable procedure. Expanding the pool of student behavior yields more useful experience, but a larger pool also creates a long-context challenge that lowers the quality of experience summarized from it in one pass. We introduce **Agent as Coach**, a coach that inspects an external behavior pool, reads selected attempts, and compresses observations before writing experience. Its acquisition and writing actions are jointly trained through the improvement of a frozen student, so the coach learns how to teach from how the student's behavior changes. We then internalize the experience into the student through training, so that the student learns to generate and use experience itself. Across multiple benchmarks with Qwen3-8B-Base and Qwen3.5-9B-Base as students, the complete pipeline consistently outperforms other rubric-based and experience-based methods, improving over the strongest of them by 4.9 and 3.8 points, respectively. Further experiments show that scored student behavior carries information beyond the rubric, that agentic reading avoids the decline of one-pass reading, and that the student learns from the experience itself, indicating that a coach can learn to teach from student behavior.

1 INTRODUCTION

Reinforcement learning from verifiable rewards has driven rapid progress in mathematical reasoning and code generation, where each problem has a single answer that can be checked (Shao et al., 2024; Guo et al., 2025; Lambert et al., 2024; Yu et al., 2025). Many tasks, such as instruction following, open-ended writing, and dialogue, admit no single correct answer (Ouyang et al., 2022; Zhou et al., 2023; Jiang et al., 2024). Their evaluation relies instead on rubrics, prompt-specific criteria that a judge scores (Zheng et al., 2023; Qin et al., 2024; Wen et al., 2024; Liu et al., 2026), and an answer is judged unsatisfactory when it departs from the preferences the rubric expresses for that prompt. Training follows the same logic, with rubrics also serving as the reward (Gunjal et al., 2026; Huang et al., 2025), so rubric-based training amounts to aligning the model with these preferences.

Existing methods can be viewed as different ways of encoding these preferences into supervision. Rubric-based reinforcement learning encodes them as a scalar reward per response, a sparse signal (Gunjal et al., 2026; Huang et al., 2025; Hou et al., 2026); rubric-guided distillation (Rezaei et al., 2026) places the rubric in a teacher's context and conveys it implicitly through token-level supervision (Snell et al., 2022; Agarwal et al., 2024). In both, the model never sees the rubric itself, only a signal derived from it. A third line of work instead places the criteria directly in the model's context, either given or written by the model before it answers (Bai et al., 2022; Sun et al., 2023; Cook et al., 2024; Guan et al., 2024; Yu et al., 2026b). In every case, however, the rubric only describes what a good answer looks like; how to produce one is left to the model's own understanding.

A more direct encoding would tell the model how to meet the rubric, not only what it requires. Experiential learning takes this step, in which a *coach* model reads a response and its feedback and writes short guidance, which we call an *experience*, that turns the rubric's criteria into concrete steps (Shinn et al., 2023; Madaan et al., 2023; Zhao et al., 2024; Ye et al., 2026a). A single response, however, shows only one way of answering, so the coach can offer only a local repair of it, and the

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

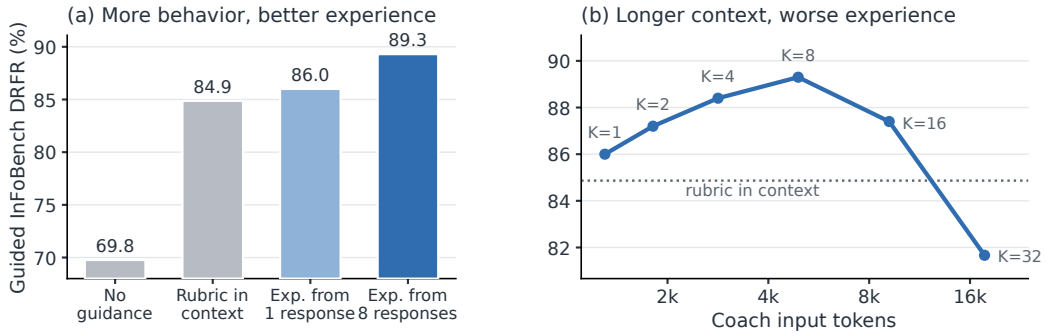


Figure 1: **Experience from more student behavior helps, but reading it in one pass does not scale.** Guided InFoBench DRFR; the student and the coach are both Qwen3-8B-Base without any training. (a) Guidance of increasing behavioral content; experiences are written by a full-context coach from 1 or 8 scored responses. (b) The same coach as its input grows with K responses.

resulting experience adds little over the rubric (86.0 versus 84.9). Given several scored responses, the coach can contrast the ways of answering that meet the rubric with those that do not, and write more useful experience (Cai et al., 2025). With eight scored responses, the experience raises the score to 89.3 (Figure 1a).

However, reading more responses is not free. Existing coaches take their input as given and read it in one pass. This holds for prompted coaches (Zhao et al., 2024; Suzgun et al., 2026; Zhang et al., 2026a; Ye et al., 2026a) and for coaches trained on downstream utility (Yao et al., 2024; Xie et al., 2025; Xi et al., 2026; Zhang et al., 2026b). Beyond eight responses, such a full-context coach degrades and falls below rubric-in-context guidance (Figure 1b). This matches findings on long contexts (Liu et al., 2024; Hsieh et al., 2024; Modarressi et al., 2025). The problem is also structural. The input grows with every response, so a large enough pool will exceed the coach's context window.

We propose **Agent as Coach** to let the coach learn from rich student behavior. The coach keeps the scored responses outside its context and reads them through tools (Yao et al., 2023), inspecting scores, reading selected responses, and compressing what it has read (Yu et al., 2026a; Sun et al., 2025) before writing the experience. After a brief protocol warm-up, reinforcement learning trains this trajectory with the frozen student's improvement as the only reward, in the spirit of learning to teach (Fan et al., 2018; Cetin et al., 2025), so the coach learns how to encode the rubric into experience from how the student's behavior changes under it. The student then internalizes the experience (Snell et al., 2022; Ye et al., 2026b) and learns to write it itself before answering, first from coached answers that beat its own best attempt, then through reinforcement learning from the question alone. At deployment it needs only the question.

On IFEval, IFBench, and InFoBench, Agent as Coach improves over rubric-based RL by 8.1 and 7.7 average points with Qwen3-8B-Base and Qwen3.5-9B-Base, and over the strongest baseline by 4.9 and 3.8 (Table 1). Scored responses add 0.69 judge points of coaching gain over the rubric alone (Table 2). The trained coach holds its gain beyond its training pool size, where full-context coaches decline (Table 3). With answer targets fixed, removing the experience costs 2.0 points, and experience from another task is worse than none (Figure 4b). Our contributions are:

- We recast rubric-based methods as encodings of rubric preferences and show that experience from a student's own scored responses encodes how to meet the rubric better than the rubric or single-response experience.
- We propose Agent as Coach, which makes experience production an agentic process: the coach learns to read behavior and write experience from the student's improvement, and the student internalizes the experience for coach-free deployment.
- We show consistent gains over rubric- and experience-based methods, and controlled analyses trace them to the information in student behavior and to the experience itself.

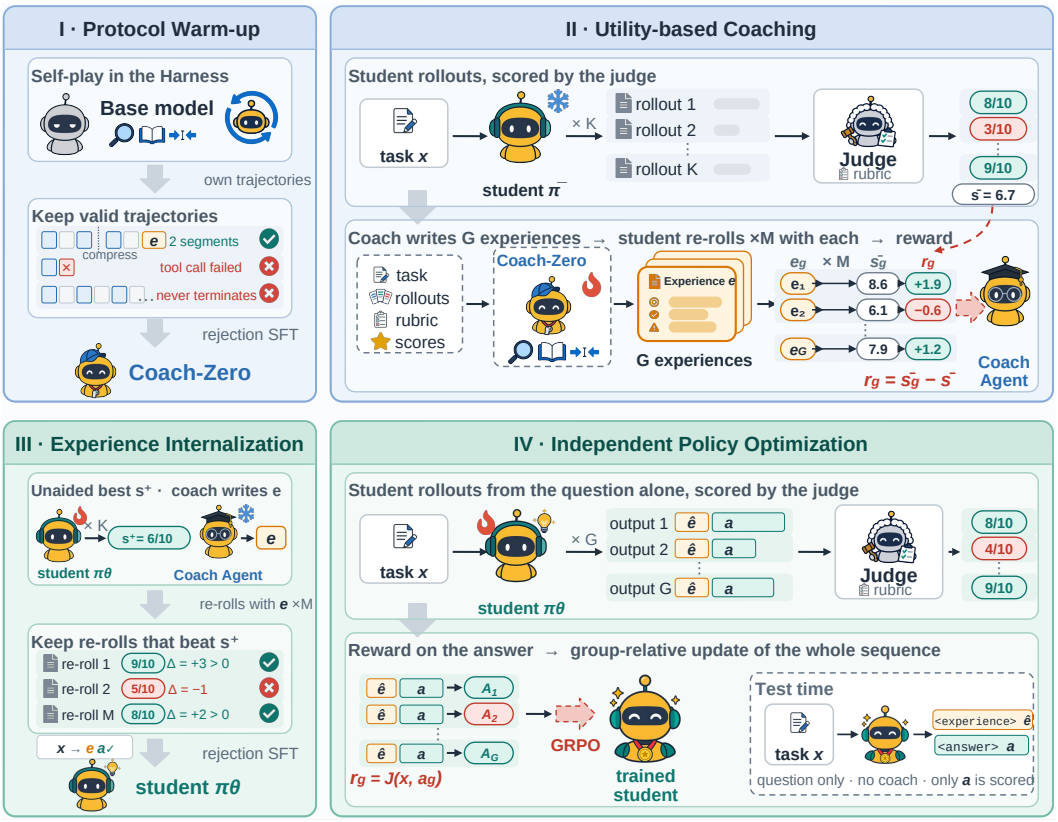


Figure 2: **Learning to teach from student behavior, then learning from the coach.** Stages I–II train the coach: a protocol warm-up, then reinforcement learning rewarded by the improvement of a frozen student. Stages III–IV train the student: supervised learning on selected experience–answer pairs, then reinforcement learning from the question alone.

2 Agent as Coach

2.1 Two learning problems

Agent as Coach first trains a coach to write experience for a frozen student, then trains the student to write its own experience (Figure 2). For a task $x \sim \mathcal{D}$ with rubric $\rho(x)$, a judge scores an answer a as $J(x, a) \in \{1, \ldots, 10\}$. A student π answers unaided, $a \sim \pi(\cdot \mid x)$, or with experience e, $a \sim \pi(\cdot \mid x, e)$. With the rubric fixed, utility and gain are

$$U_\pi(e; x) = \mathbb{E}_{a \sim \pi(\cdot|x,e)} J(x, a), \qquad \nu_\pi(e; x) = U_\pi(e; x) - U_\pi(\varnothing; x), \tag{1}$$

where $\varnothing$ denotes empty guidance.

The student supplies a behavior pool $B_x^\pi = \{(a_k, s_k)\}_{k=1}^K$ with $a_k \sim \pi(\cdot \mid x)$ and $s_k = J(x, a_k)$. A coach c_ϕ interacts with the pool through a trajectory h and emits $e = \mathrm{out}(h)$. For a frozen training student $\bar{\pi}$, the *coaching problem* (Section 2.2) is

$$\max_\phi \; \mathbb{E}_{x, B_x^{\bar{\pi}}, h \sim c_\phi} \left[\nu_{\bar{\pi}}(\mathrm{out}(h); x) \right]. \tag{2}$$

The *student problem* (Section 2.3) removes the coach at deployment. Given only x, the student writes its own experience $\hat{e}$ and then the answer a:

$$\max_\theta \; \mathbb{E}_{x \sim \mathcal{D}} \, \mathbb{E}_{(\hat{e}, a) \sim \pi_\theta(\cdot|x)} \left[J(x, a) \right]. \tag{3}$$

2.2 LEARNING TO COACH

Summarizing a large behavior pool in one pass degrades the experience, and the pool can outgrow the coach's context window (Figure 1b). We therefore place the coach in an agent harness, where the pool stays outside its context and the coach explores it with three tools in a ReAct-style loop (Yao et al., 2023): `inspect` lists the candidate answers with their scores to survey the pool, `read` retrieves a selected answer or the rubric to examine the contrasts it needs, and `compress` replaces what has been read with coach-written memory to keep useful findings. The coach thus decides which answers to read and what to keep before writing the experience. We train it in two stages (Figure 2, top).

Stage I: protocol warm-up. To use the harness correctly, the base model runs in it on training tasks; we keep trajectories with valid tool calls that end with an experience and fine-tune on them. The resulting Coach-Zero uses the tools reliably but has not learned what makes an experience useful.

Stage II: utility-based coaching. Stage II optimizes the coaching problem of Eq. (2). For each task, the coach samples G trajectories on the frozen student's pool, and each reward estimates the gain of Eq. (1): the pool mean $\bar{s}$ estimates the unguided utility, and the mean $\bar{s}_g$ of M fresh answers guided by e_g the guided one,

$$r_g = \begin{cases} \bar{s}_g - \bar{s}, & h_g \text{ valid}, \\ -10, & h_g \text{ invalid}, \end{cases} \qquad A_g = \frac{r_g - \operatorname{mean}_{g'} r_{g'}}{\operatorname{std}_{g'} r_{g'}}. \tag{4}$$

Each `compress` call closes a segment, and the next segment starts from the task prompt p_x and the latest memory w_{n-1} instead of the full history, so h_g splits into segments $\sigma_{g,1}, \ldots, \sigma_{g,N_g}$. Only coach-generated tokens $\mathcal{T}(\sigma_{g,n})$ are optimized, tool observations are masked, and all segments share the advantage A_g:

$$\mathcal{J}_{\mathrm{II}}(\phi) = \mathbb{E}\left[\frac{1}{G}\sum_{g=1}^{G}\frac{A_g}{|\mathcal{T}(h_g)|}\sum_{n=1}^{N_g}\sum_{t\in\mathcal{T}(\sigma_{g,n})}\log c_\phi(a_{g,n,t}\,|\,p_x, w_{g,n-1}, \sigma_{g,n,<t})\right]. \tag{5}$$

Dividing by $|\mathcal{T}(h_g)|$, the number of coach-generated tokens in h_g, keeps a trajectory's weight independent of how many segments it has. One reward thus trains what the coach reads, keeps, and writes from how the student's answers change.

2.3 LEARNING FROM THE COACH

At deployment, the student should answer without a coach, so we internalize the experience in two stages (Figure 2, bottom).

Stage III: experience internalization. With the trained coach, Agent as Coach, frozen, the student answers each task K times alone, reaching a best score $s^+ = \max_k J(x, a_k)$. The coach reads this pool and writes e, and the student answers $M = K$ more times with e. We keep only answers that beat s^+, $\mathcal{D}_{\mathrm{III}} = \{(x, e, a'_m) : J(x, a'_m) > s^+\}$, so each reflects a gain beyond the student's own best attempt, and fine-tune the student to produce the experience and then the answer from x alone:

$$\mathcal{L}_{\mathrm{III}}(\theta) = -\mathbb{E}_{(x,e,a)\sim\mathcal{D}_{\mathrm{III}}}\log\pi_\theta\big(y \mid x\big), \qquad y = (e, a). \tag{6}$$

Stage IV: independent policy optimization. Finally, the student optimizes the student problem of Eq. (3) with reinforcement learning. Given only x, it samples G outputs, each writing its own experience $\hat{e}_g$ and then an answer a_g. The judge scores only the answer, and the normalized score updates the whole output:

$$\mathcal{J}_{\mathrm{IV}}(\theta) = \mathbb{E}_x\left[\frac{1}{G}\sum_{g=1}^{G}\tilde{A}_g\,\log\pi_\theta\big(\hat{e}_g, a_g \mid x\big)\right], \qquad \tilde{A}_g = \frac{J(x, a_g) - \operatorname{mean}_{g'} J(x, a_{g'})}{\operatorname{std}_{g'} J(x, a_{g'})}. \tag{7}$$

Because the reward flows into $\hat{e}_g$ as well as a_g, the student learns to write experience that helps its own answers.

Table 1: **Independent performance from the question alone (%).** IFEval/IFBench use strict prompt accuracy; InFoBench uses DRFR; Avg. is their unweighted mean.

	Qwen3-8B-Base				Qwen3.5-9B-Base			
Method	**IFEval**	**IFBench**	**InFoBench**	**Avg.**	**IFEval**	**IFBench**	**InFoBench**	**Avg.**
Base model	43.4	16.9	69.8	43.4	74.5	21.9	84.1	60.2
SFT	55.3	19.3	73.4	49.3	81.3	21.9	85.1	62.8
Rubric-based methods								
Rubric RL (Gunjal et al., 2026)	63.3	23.1	79.1	55.2	82.4	22.1	85.2	63.2
Rubric distillation (Rezaei et al., 2026)	56.5	18.2	83.7	52.8	82.3	25.7	86.6	64.9
Think-with-Rubrics (Yu et al., 2026b)	61.1	20.9	82.6	54.9	85.1	25.8	85.2	65.4
Experience-based methods								
EL-style distillation (Ye et al., 2026a)	65.7	24.8	83.3	57.9	86.5	26.9	87.9	67.1
Group refinement (Huang et al., 2026)	62.2	21.1	82.2	55.2	84.9	25.9	85.6	65.5
Full-context coach	66.9	24.1	84.3	58.4	87.1	26.4	86.8	66.8
Agent as Coach (Ours)	**73.3**	**30.9**	**85.8**	**63.3**	**89.1**	**32.4**	**91.2**	**70.9**

3 EXPERIMENTS

Section 1 argued that experience tells a student how to meet a rubric, that scored behavior makes this experience more useful, and that reading a growing pool calls for an agent. We test these claims and the final internalization step with four questions.

- **RQ1 (Effectiveness).** Does Agent as Coach improve question-only performance over methods that encode the rubric as a standard and over other ways of producing experience? (Table 1)
- **RQ2 (Student behavior).** Does scored student behavior carry information beyond the rubric, and does the reward from student improvement teach the coach to use it? (Table 2, Figure 3b)
- **RQ3 (Agentic reading).** As the pool grows, does reading as an agent avoid the decline of full-context coaches? (Figure 3a, Table 3)
- **RQ4 (Internalization).** Does the student learn from the experience itself, beyond the answers it leads to? (Figure 4, Table 4)

3.1 SETUP

Data and models. All training tasks come from OpenRubrics (Liu et al., 2026), where each task is paired with a rubric. We use 5,000 tasks for the SFT stages and 27,000 other tasks for the RL stages. Coach training and student training share this split, with Stages I and III on the SFT tasks and Stages II and IV on the RL tasks. The students are Qwen3-8B-Base (Yang et al., 2025) and Qwen3.5-9B-Base (Qwen Team, 2026), and for each student the coach starts from the same base model. In Agent as Coach, no stronger model writes experience or answers at any stage. GPT-4o (Hurst et al., 2024) serves as the judge J and only scores answers.

Evaluation. We evaluate on three instruction-following benchmarks, where a question admits many valid answers and each answer must satisfy a list of requirements. IFEval (Zhou et al., 2023) contains 541 prompts with verifiable constraints on length, format, and keywords, and IFBench (Pyatkin et al., 2025) contains 300 prompts with 58 new constraints unseen in common training data. Both are checked by rules, and we report strict prompt accuracy. InFoBench (Qin et al., 2024) contains 500 instructions decomposed into 2,250 requirements, which GPT-4o judges one by one, and we report DRFR, the fraction of requirements satisfied. IFEval and IFBench test whether hard constraints are met exactly, while InFoBench also covers open requirements on content and style, so together the three benchmarks span both checkable and open-ended requirements.

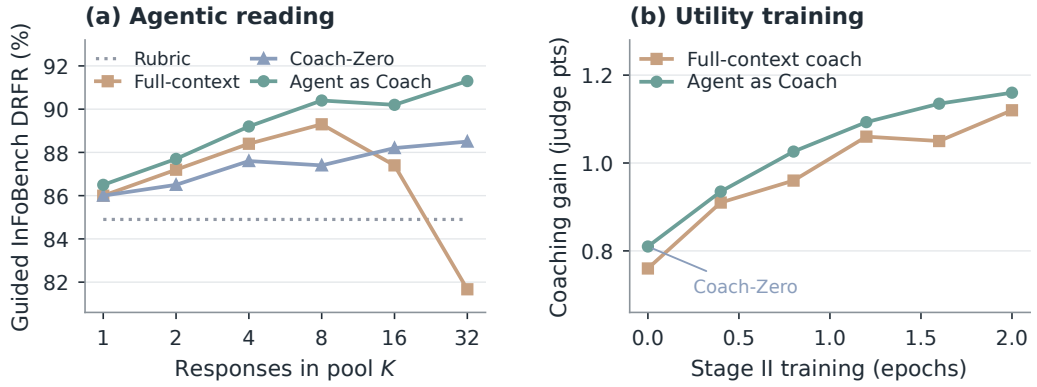


Figure 3: **Agentic reading and utility training.** (a) Guided InFoBench DRFR of a frozen student on the same pools. The full-context coach is untrained, Coach-Zero has only the Stage I warm-up, and Agent as Coach is trained at $K = 8$. (b) Coaching gain during Stage II.

Comparisons. Table 1 compares Agent as Coach with the base models, SFT on the preferred responses of all training tasks, and two groups of methods. Rubric-based methods include rubric-reward RL (Gunjal et al., 2026), rubric distillation (Rezaei et al., 2026), and Think-with-Rubrics (Yu et al., 2026b). Experience-based methods include EL-style context distillation (Ye et al., 2026a;b), GOLF-inspired group refinement (Huang et al., 2026), and a full-context coach, which reads all scored responses in one prompt, is trained on downstream reward as in CTRL (Xie et al., 2025), and otherwise follows the same four stages as Agent as Coach. All methods start from the same base model and use the same training tasks and judge.

3.2 RQ1: Effectiveness

Agent as Coach outperforms all baselines. Table 1 shows that Agent as Coach achieves the best result on every benchmark with both students, with an average of 63.3 on Qwen3-8B-Base and 70.9 on Qwen3.5-9B-Base. It improves over the strongest baselines, the full-context coach and EL-style distillation, by 4.9 and 3.8 points, and over rubric-based RL by 8.1 and 7.7 points. The gain is larger for the weaker student, whose base model follows instructions less well (43.4 versus 60.2 for Qwen3.5-9B-Base).

Experience helps more than the rubric. SFT on preferred responses reaches only 49.3 and 62.8. Rubric-based methods reach 52.8–55.2 on the 8B student and 63.2–65.4 on the 9B student. Experience-based methods reach 55.2–58.4 and 65.5–67.1, and the best of them surpasses every rubric-based method on both students. Telling the student how to meet the rubric thus helps more than conveying what the rubric requires.

The full-context coach goes through the same four stages as Agent as Coach but reaches only 58.4 and 66.8. The remaining gap lies in what the coach reads and how it reads it, which RQ2 and RQ3 examine in turn.

3.3 RQ2: Student behavior

Figure 1a showed that experience from eight scored responses helps more than experience from one. RQ2 asks whether this gain comes from information in the behavior itself, and whether training teaches the coach to use it. We measure a coach by its *coaching gain* on held-out OpenRubrics tasks. It is the mean judge score of the frozen base student's answers with the coach's experience, minus that without it. Table 2 extends the comparison of Figure 1a to trained coaches and adds a rubric-only condition, with the task, the rubric, and the frozen student fixed. With the rubric alone, the trained coach improves the student by 0.47 judge points. One scored answer raises the gain to 0.68, and eight answers raise it to 1.16. Student behavior thus carries information that the rubric does not contain,

Table 2: **Student behavior is worth more to a trained coach.** Coaching gain (judge points) on 500 held-out tasks with the task, rubric, and frozen student fixed. EVE is the gain with eight answers minus the gain with the rubric only.

	Evidence given to the coach			
Coach	**Rubric only**	**+1 answer**	**+8 answers**	**EVE**
Prompted coach	0.28	0.43	0.61	0.33
Coach-Zero (Stage I)	0.36	0.55	0.81	0.45
Agent as Coach (Stage II)	**0.47**	**0.68**	**1.16**	**0.69**

Table 3: **Agentic reading keeps its advantage as the pool grows.** Guided InFoBench DRFR (%). Beyond $K = 32$, the full-context input exceeds the 32k-token window and is truncated.

	Responses in the pool K				
Coach	**8**	**16**	**32**	**64**	**128**
Full-context input (tokens)	5k	9k	18k	35k	68k
Rubric in context	84.9	84.9	84.9	84.9	84.9
Full-context, untrained	89.3	87.4	81.7	80.2	78.9
Full-context, trained on mixed K (8–32)	89.6	88.7	87.2	84.8	82.9
Agent as Coach (trained at K=8)	**90.4**	**90.2**	**91.3**	**90.9**	**91.1**

and each added answer helps the trained coach more than the other two coaches (0.21 and 0.48 for the two steps, against 0.19 and 0.26 for Coach-Zero).

The reward from student improvement teaches the coach to use this information. Relative to the rubric-only condition, eight answers add 0.69 points for the trained coach, compared with 0.45 for Coach-Zero and 0.33 for the prompted coach. Of this gain, the warm-up contributes 0.12 over prompting and Stage II another 0.24, so most of it comes from the reward. Training also improves the rubric-only condition (0.47 versus 0.36 and 0.28), but the largest gap appears with eight answers (1.16 versus 0.81), where the trained coach leads Coach-Zero by 0.35 points. Figure 3b shows this ability forming during Stage II, where the coaching gain rises from 0.81 to 1.16. The full-context coach, trained with the same reward, also rises from 0.76 to 1.12 but stays below Agent as Coach throughout training.

3.4 RQ3: Agentic reading

Section 1 identified two limits of full-context reading. It declines beyond eight responses, and its input eventually exceeds the context window. RQ3 asks whether reading as an agent removes both. Figure 3a evaluates the coaches on the same pools as the pool grows. The full-context coach without training peaks at $K = 8$ (89.3%) and falls to 81.7% at $K = 32$, below the rubric in context (84.9%), even though its input still fits in the context window. The decline already begins at $K = 16$, where the input holds only about 9k tokens. Coach-Zero reads the same pools through the harness with only the Stage I warm-up. It trails the full-context coach at $K = 8$ (87.4% versus 89.3%) but stays stable as the pool grows and leads by 6.8 points at $K = 32$ (88.5%). Reading through tools thus removes the decline even before Stage II. Training then raises the level, and Agent as Coach reaches 90.4% at $K = 8$ and 91.3% at $K = 32$.

Table 3 addresses the structural limit. Training the full-context coach on mixed pool sizes ($K \in \{8, 16, 32\}$) reduces the decline and reaches 87.2% at $K = 32$, but it still stays below Agent as Coach (91.3%). Mixed-K training helps most within its training range, adding 5.5 points over the untrained coach at $K = 32$ but 4.0 points at $K = 128$. Beyond $K = 32$, the full-context input exceeds the 32k-token window and must be truncated, and even the mixed-K coach falls to 82.9% at $K = 128$, below the rubric in context. Agent as Coach reads within a fixed allowance and stays at 91.1% at $K = 128$, so its lead over the mixed-K coach grows from 0.8 points at $K = 8$ to 8.2 points at $K = 128$. Reading as an agent thus removes both limits of full-context reading.

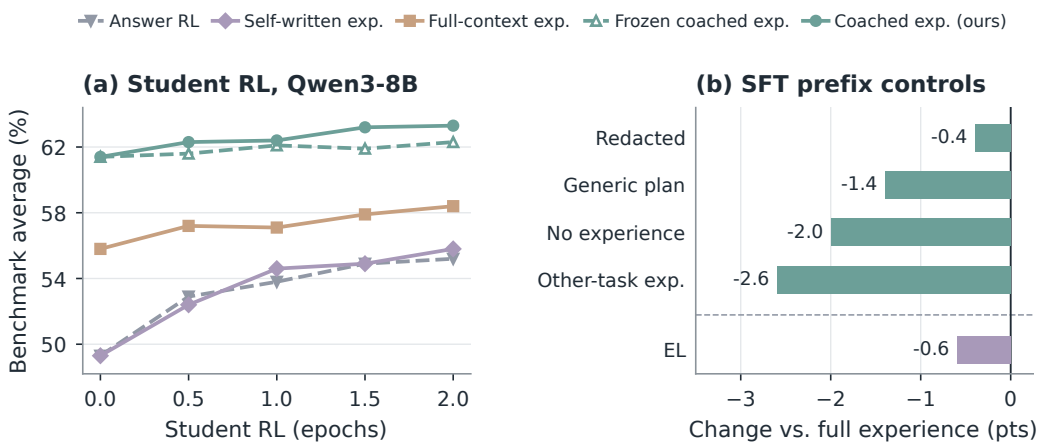


Figure 4: **Internalization on Qwen3-8B-Base.** (a) Question-only average during student RL. Runs start from SFT (49.3), full-context experience SFT (55.8), or Stage III (61.4). (b) Change relative to full experience after SFT. The first four conditions share the same answer targets, and EL uses its own on-policy targets.

3.5 RQ4: Internalization

Table 4: **Internalization on both students.** Question-only average (%) after SFT and after RL. The self-written arm uses no coach.

Student training	Qwen3-8B-Base	Qwen3.5-9B-Base
SFT	49.3	62.8
+ answer-only RL	55.2	63.2
+ self-written exp. RL	55.8	63.6
Full-context coach exp. SFT	55.8	65.2
+ joint RL	58.4	66.8
Stage III exp. SFT	61.4	68.6
+ RL, frozen exp.	62.3	70.2
+ joint RL (Stage IV)	**63.3**	**70.9**

At deployment the student answers without a coach. RQ4 asks whether it learns from the experience itself or only from the better answers that the experience leads to. Figure 4b fixes the answer targets that the 8B student collects on the 5,000 SFT tasks and changes only the prefix before each answer. Removing the experience lowers the score by 2.0 points. A generic plan of the same length recovers only 0.6 points, so the benefit is not the format of planning before answering. Experience from another task is worse than no prefix at all (2.6 versus 2.0 points lost), so the student follows the procedure it is given. Redacting entities and values costs only 0.4 points, which places the benefit in the procedure rather than in answer content the experience reveals. EL, which internalizes the same experience through context distillation, stays 0.6 points below. The student therefore learns from the experience itself, and learns most when it writes the experience explicitly.

The experience also carries into reinforcement learning (Figure 4a, Table 4). Stage III gives the 8B student a starting point of 61.4, well above the 49.3 of SFT on preferred responses, and Stage IV raises it to 63.3. Stage III alone already exceeds every baseline in Table 1 on both students (61.4 versus 58.4 and 68.6 versus 67.1). From SFT, RL that also writes an experience reaches only 55.8, close to the 55.2 of answer-only RL, so the student does not learn useful experience without the coach. When a frozen Stage III copy writes the experience and only the answer is trained, the student reaches 62.3, gaining only 0.9 points during RL against 1.9 for joint training. Updating the experience together with the answer thus adds 1.0 points. Experience from the full-context coach also lifts SFT, from 49.3 to 55.8, but stays 5.6 points below Stage III, so the quality of the coach carries over to the student. On the 9B student, Stage III starts 3.4 points above full-context experience SFT (68.6

versus 65.2), and the final results follow the same order, reaching 70.9 with coached experience and 66.8 with full-context experience. Internalization thus carries the gain of the coach into a student that answers from the question alone.

4 Related work

From rubrics to experience. Judges score rubrics reliably enough to serve as training signals (Zheng et al., 2023), and rubric-based benchmarks now cover complex instructions with many constraints (Qin et al., 2024; Wen et al., 2024; Jiang et al., 2024). Rubrics therefore serve as rewards on tasks without a single correct answer (Gunjal et al., 2026; Liu et al., 2026; Huang et al., 2025) and guide exploration in open-ended reinforcement learning (Hou et al., 2026). They also serve as teacher context for distillation into a question-only student (Rezaei et al., 2026), and as criteria placed in the model's own context before it answers (Bai et al., 2022; Sun et al., 2023; Guan et al., 2024; Yu et al., 2026b). All of these methods encode the rubric as a standard for the final answer. Experiential learning instead turns attempts and feedback into guidance on how to answer. Reflexion, Self-Refine, and ExpeL derive insights from a model's own attempts (Shinn et al., 2023; Madaan et al., 2023; Zhao et al., 2024), later work keeps them in an evolving memory or an experience library updated from group rollouts (Suzgun et al., 2026; Zhang et al., 2026a; Cai et al., 2025), LLM-as-a-Coach writes experience for non-verifiable tasks and internalizes it by on-policy context distillation (Ye et al., 2026a;b). These methods produce experience with a prompted model that reads its input at once, so the experience does not improve with how the student responds to it. GOLF trains the policy to refine its failed attempts from group-level feedback (Huang et al., 2026), but it refines answers rather than writing reusable guidance. Agent as Coach trains the coach and lets it read the behavior of the very student it teaches.

Learning to teach. Another line trains the guidance itself by its downstream effect. Retroformer tunes a reflection model from the actor's returns (Yao et al., 2024), CTRL and Critique-RL reward critiques that help a generator correct its answer (Xie et al., 2025; Xi et al., 2026), Zhang et al. (2026b) train a skill generator by the marginal utility of its skills, and learning to teach rewards a teacher by its effect on the student (Fan et al., 2018; Cetin et al., 2025). Our Stage II follows this principle, and its reward also trains how the coach acquires evidence. This acquisition draws on agents that act on long inputs rather than read them at once, since models degrade as the context grows (Liu et al., 2024; Hsieh et al., 2024). ReAct interleaves reasoning with tool calls (Yao et al., 2023), and MemAgent and context folding compress what the agent has read (Yu et al., 2026a; Sun et al., 2025). The coach in Agent as Coach treats the behavior pool as such an environment. Unlike a critic that reads one answer and repairs it once, it reads many answers and writes guidance meant to transfer to fresh attempts. The student then writes the experience itself instead of distilling it (Snell et al., 2022), which keeps the procedure explicit and lets reinforcement learning refine it.

5 Conclusion

We presented Agent as Coach, a coach that learns to teach from student behavior. The coach reads a pool of scored student attempts as an agent and is trained by how much its experience improves a frozen student. The student then learns to write and use such experience on its own. Across two students and three benchmarks, the complete pipeline outperforms rubric-based and experience-based methods. Student behavior adds value beyond the rubric, agentic reading keeps this value as the pool grows, and the student learns from the experience itself. Because the student ends up writing its own experience, this gain remains after the coach is removed at deployment. More broadly, the results suggest that useful supervision for tasks without a single correct answer need not come from a stronger model. It can come from the student's own scored behavior, read selectively and turned into procedures, with a coach of the same size as the student. Its only training signal is the change in the student's scores, so what it learns reflects how this student fails and succeeds. Richer feedback than scalar scores, such as per-criterion judgments, could let the coach read even more from the same behavior. Extending the coach to domains beyond instruction following, such as long-form writing and agentic tasks, is a natural next step. More generally, the results invite treating the teacher in post-training as a learnable agent rather than a fixed prompt, one that improves by watching the student it teaches.

AI USE STATEMENT

In this work, we used generative AI tools to assist in writing proofs, namely to check and revise the derivation in Appendix C, and to translate our Chinese drafts into English. Additionally, we used generative AI tools to draft and revise parts of the paper, write plotting code, and format tables and figures. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Section 2 specifies the objectives of all four stages, and Section 3.1 describes the data, models, benchmarks, and baselines. The appendix gives the coach harness, the training algorithm, the data splits, the hyperparameters, and the baseline implementations. We will release the code and data.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-Policy Distillation of Language Models: Learning from Self-Generated Mistakes. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2306.13649.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional AI: Harmlessness from AI Feedback, 2022. URL https://arxiv.org/abs/2212.08073.

Yuzheng Cai, Siqi Cai, Yuchen Shi, Zihan Xu, Lichao Chen, Yulei Qin, Xiaoyu Tan, Gang Li, Zongyi Li, Haojia Lin, Yong Mao, Ke Li, and Xing Sun. Training-Free Group Relative Policy Optimization, 2025. URL https://arxiv.org/abs/2510.08191.

Edoardo Cetin, Tianyu Zhao, and Yujin Tang. Reinforcement Learning Teachers of Test Time Scaling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://arxiv.org/abs/2506.08388.

Jonathan Cook, Tim Rocktäschel, Jakob Foerster, Dennis Aumiller, and Alex Wang. TICKing All the Boxes: Generated Checklists Improve LLM Evaluation and Generation, 2024. URL https://arxiv.org/abs/2410.03608.

Yang Fan, Fei Tian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. Learning to Teach. In *International Conference on Learning Representations (ICLR)*, 2018. URL https://arxiv.org/abs/1805.03643.

Melody Y. Guan, Manas Joglekar, Eric Wallace, Saachi Jain, Boaz Barak, et al. Deliberative Alignment: Reasoning Enables Safer Language Models, 2024. URL https://arxiv.org/abs/2412.16339.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as Rewards: Reinforcement Learning Beyond Verifiable Domains. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2507.17746.

Daya Guo, Dejian Yang, Haowei Zhang, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://www.nature.com/articles/s41586-025-09422-z.

Jinkun Hou, Zhuo Liu, Huimin Ren, Hongsheng Xin, Pan Zhou, and Kun Zhan. RISE-RL: Rubric-informed selective exploration for open-ended reinforcement learning, 2026. URL https://arxiv.org/abs/2608.09123.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the Real Context Size of Your Long-Context Language Models? In *Conference on Language Modeling (COLM)*, 2024. URL https://arxiv.org/abs/2404.06654.

Lei Huang, Xiang Cheng, Chenxiao Zhao, Guobin Shen, Junjie Yang, Xiaocheng Feng, Yuxuan Gu, Xing Yu, and Bing Qin. Bootstrapping Exploration with Group-Level Natural Language Feedback in Reinforcement Learning, 2026. URL https://arxiv.org/abs/2603.04597.

Zenan Huang, Yihong Zhuang, Guoshan Lu, Zeyu Qin, Haokai Xu, Tianyu Zhao, Ru Peng, Jiaqi Hu, Zhanming Shen, Xiaomeng Hu, Xijun Gu, Peiyi Tu, Jiaxin Liu, Wenyu Chen, Yuzhuo Fu, Zhiting

Fan, Yanmei Gu, Yuanyuan Wang, Zhengkai Yang, Jianguo Li, and Junbo Zhao. Reinforcement Learning with Rubric Anchors, 2025. URL https://arxiv.org/abs/2508.12790.

Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, et al. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. FollowBench: A multi-level fine-grained constraints following benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4667–4688. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.257. URL https://aclanthology.org/2024.acl-long.257/.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing Frontiers in Open Language Model Post-Training, 2024. URL https://arxiv.org/abs/2411.15124.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Tianci Liu, Ran Xu, Tony Yu, Ilgee Hong, Carl Yang, Tuo Zhao, and Haoyu Wang. OpenRubrics: Towards Scalable Synthetic Rubric Generation for Reward Modeling and LLM Alignment. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17417–17437, 2026. doi: 10.18653/v1/2026.acl-long.791. URL https://aclanthology.org/2026.acl-long.791/.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative Refinement with Self-Feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. URL https://arxiv.org/abs/2303.17651.

Ali Modarressi, Hanieh Deilamsalehy, Franck Dernoncourt, Trung Bui, Ryan A. Rossi, Seunghyun Yoon, and Hinrich Schütze. NoLiMa: Long-Context Evaluation Beyond Literal Matching. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, pp. 44554–44570, 2025. URL https://arxiv.org/abs/2502.05167.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training Language Models to Follow Instructions with Human Feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing Verifiable Instruction Following. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2025. URL https://arxiv.org/abs/2507.02833.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. InFoBench: Evaluating Instruction Following Ability in Large Language Models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13025–13048, 2024. doi: 10.18653/v1/2024.findings-acl.772. URL https://aclanthology.org/2024.findings-acl.772/.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

MohammadHossein Rezaei, Anas Mahmoud, Zihao Wang, Utkarsh Tyagi, Advait Gosai, Razvan-Gabriel Dumitru, Aakash Sabharwal, Bing Liu, and Yunzhong He. Rubric-Guided Self-Distillation: Post-Training Without Rubric Verifiers, 2026. URL https://arxiv.org/abs/2606.12507.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models, 2024. URL https://arxiv.org/abs/2402.03300.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning, 2023. URL https://arxiv.org/abs/2303.11366.

Charlie Snell, Dan Klein, and Ruiqi Zhong. Learning by Distilling Context, 2022. URL https://arxiv.org/abs/2209.15189.

Weiwei Sun, Miao Lu, Zhan Ling, Kang Liu, Xuesong Yao, Yiming Yang, and Jiecao Chen. Scaling Long-Horizon LLM Agent via Context-Folding, 2025. URL https://arxiv.org/abs/2510.11967.

Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David Cox, Yiming Yang, and Chuang Gan. Principle-Driven Self-Alignment of Language Models from Scratch with Minimal Human Supervision. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. URL https://arxiv.org/abs/2305.03047.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic Cheatsheet: Test-Time Learning with Adaptive Memory. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pp. 7080–7106, 2026. doi: 10.18653/v1/2026.eacl-long.333. URL https://aclanthology.org/2026.eacl-long.333/.

Bosi Wen, Pei Ke, Xiaotao Gu, Lindong Wu, Hao Huang, Jinfeng Zhou, Wenchuang Li, Binxin Hu, Wendy Gao, Jiaxin Xu, Yiming Liu, Jie Tang, Hongning Wang, and Minlie Huang. Benchmarking complex instruction-following with multiple constraints composition. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4371. URL https://papers.nips.cc/paper_files/paper/2024/hash/f8c24b08b96a08ec7a7a975feea7777e-Abstract-Datasets_and_Benchmarks_Track.html.

Zhiheng Xi, Jixuan Huang, Xin Guo, Boyang Hong, Dingwen Yang, Xiaoran Fan, Shuo Li, Zehui Chen, Junjie Ye, Siyu Yuan, Zhengyin Du, Xuesong Yao, Yufei Xu, Jiecao Chen, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. Critique-RL: Training Language Models for Critiquing through Two-Stage Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2510.24320.

Zhihui Xie, Jie Chen, Liyu Chen, Weichao Mao, Jingjing Xu, and Lingpeng Kong. Teaching Language Models to Critique via Reinforcement Learning. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, pp. 68559–68577, 2025. URL https://arxiv.org/abs/2502.03492.

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

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://arxiv.org/abs/2210.03629.

Weiran Yao, Shelby Heinecke, Juan Carlos Niebles, Zhiwei Liu, Yihao Feng, Le Xue, Rithesh Murthy, Zeyuan Chen, Jianguo Zhang, Devansh Arpit, Ran Xu, Phil Mui, Huan Wang, Caiming Xiong, and Silvio Savarese. Retroformer: Retrospective Large Language Agents with Policy Gradient Optimization. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2308.02151.

Tianzhu Ye, Li Dong, Guanheng Chen, He Zhu, Xun Wu, Shaohan Huang, and Furu Wei. LLM-as-a-Coach: Experiential Learning for Non-Verifiable Tasks, 2026a. URL https://arxiv.org/abs/2607.18110.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-Policy Context Distillation for Language Models, 2026b. URL https://arxiv.org/abs/2602.12275.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, and Hao Zhou. MemAgent: Reshaping Long-

Context LLM with Multi-Conv RL-based Memory Agent. In *International Conference on Learning Representations (ICLR)*, 2026a. URL https://arxiv.org/abs/2507.02259.

Jiachen Yu, Zhihao Xu, Junjie Wang, and Yujiu Yang. Think-with-Rubrics: From External Evaluator to Internal Reasoning Guidance, 2026b. URL https://arxiv.org/abs/2605.07461.

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

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic Context Engineering: Evolving Contexts for Self-Improving Language Models. In *International Conference on Learning Representations (ICLR)*, 2026a. URL https://arxiv.org/abs/2510.04618.

Zhiwei Zhang, Yudi Lin, Nikki Lijing Kuang, Linlin Wu, Xiaomin Li, Songtao Liu, and Fenglong Ma. Co-Evolving Skill Generation and Policy Optimization, 2026b. URL https://arxiv.org/abs/2606.08755.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM Agents Are Experiential Learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL https://arxiv.org/abs/2308.10144.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023. URL https://arxiv.org/abs/2306.05685.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-Following Evaluation for Large Language Models, 2023. URL https://arxiv.org/abs/2311.07911.

A TRAINING DETAILS

A.1 COACH HARNESS

Figure 5 shows the harness, and Algorithm 1 gives one coaching trajectory with the Stage II update. The scored pool stays outside the coach's context. At each turn the coach either calls one tool or gives its final response, which is the experience. `inspect` returns anonymous identifiers, lengths, and scores. `read` returns one answer or the rubric. `compress` replaces the transcript with memory written by the coach and starts a new segment. A trajectory is invalid if a tool call cannot be executed, a limit is exceeded, or the experience cites candidate identifiers or scores. An invalid trajectory gives empty guidance and a reward of -10.

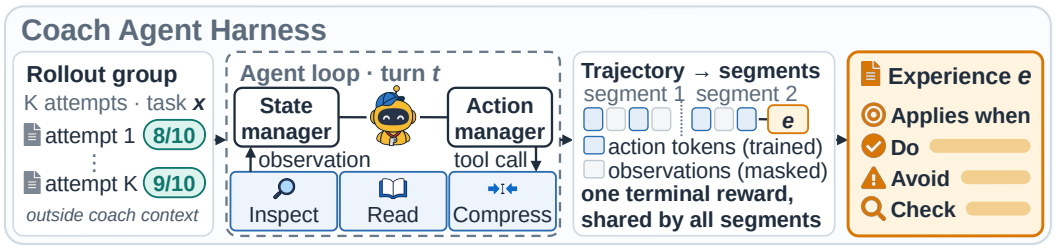


Figure 5: **The coach harness.** The scored pool stays outside the coach's context, and the coach reaches it only through `inspect` and `read`. Each `compress` call closes a training segment, and all segments of a trajectory share one terminal reward.

Experience interface. The coach writes a concise strategy with four fields, namely applicability, actions, pitfalls, and checks. The experience is judged only through the answers of the student who receives it. In the running example of Appendix D, the experience tells the student to choose two rhyme pairs, assign them in ABAB order, and compose the poem around those endings.

Information flow. The coach sees the task, the rubric, the attempted answers, and their scores. The student sees the task and the experience during coaching, and only the task at deployment, where it writes its own experience before answering.

A.2 DATA AND EVALUATION

All training tasks come from OpenRubrics, and each task has a rubric. The SFT stages (I and III) use 5,000 tasks and the RL stages (II and IV) use 27,000 tasks, with no question shared between the two sets. We remove training tasks whose instruction overlaps with an evaluation prompt (word five-gram Jaccard above 0.8). Coaching gain is measured on 500 held-out OpenRubrics tasks outside both sets. For each task, the frozen student samples 32 answers with the experience and 32 without it, and the gain is the difference of their mean judge scores. GPT-4o scores each answer from the question, the rubric, and the answer on the 1–10 scale.

IFEval has 541 prompts, IFBench has 300 prompts, and InFoBench has 500 prompts with 2,250 decomposed requirements. For a benchmark with N prompts, where prompt i has n_i requirements with binary decisions z_{ij}, strict prompt accuracy is $100N^{-1}\sum_i \prod_j z_{ij}$ and InFoBench DRFR is $100\sum_{ij} z_{ij} / \sum_i n_i$. IFEval and IFBench use their deterministic checkers, and InFoBench uses GPT-4o. At evaluation, every student receives only the question and samples one answer at temperature 1.0 with up to 4,096 new tokens. Students trained with experience output one `<experience>` block followed by one `<answer>` block, and only the answer is scored. Table 5 lists the training hyperparameters.

A.3 BASELINES

- **SFT** fine-tunes the base model on the preferred response of each OpenRubrics pair for all 32,000 training tasks, without experience. **Rubric RL** continues from SFT with GRPO on the RL tasks, using the rubric-conditioned judge as reward. It is also the answer-only RL arm of Table 4.

Algorithm 1 One coaching trajectory and the Stage II update

Require: task x with rubric; pool $B_x^{\bar{\pi}}$ with mean score $\bar{s}$; coach c_ϕ; frozen student $\bar{\pi}$; G, M

function ROLLOUT($x, B_x^{\bar{\pi}}$)
  $w \leftarrow \varnothing$; $n \leftarrow 1$; $\sigma_1 \leftarrow ()$; reset the tool, evidence, and token counters
  loop
    $a \sim c_\phi(\cdot \mid p_x, w, \sigma_n)$; append a to σ_n
    charge all generated tokens in a, including memory or final experience
    if action type, arguments, output format, or limits are invalid **then**
      return $(\varnothing, \text{invalid}, \sigma_{1:n})$
    end if
    if a is `inspect` or `read` **then**
      prepare observation o under the remaining evidence allowance
      if the observation cannot be admitted within the limits **then**
        return $(\varnothing, \text{invalid}, \sigma_{1:n})$
      end if
      charge the admitted observation; append o to σ_n
    else if a is `compress` **then**
      $w \leftarrow$ memory in a; close σ_n
      $n \leftarrow n + 1$; $\sigma_n \leftarrow ()$ with initial context (p_x, w)
    else ▷ validated final response, already recorded
      return $(a, \text{valid}, \sigma_{1:n})$
    end if
  end loop
end function

for $g = 1, \ldots, G$ **do** ▷ Stage II, one task
  $(e_g, v_g, \{\sigma_{g,n}\}) \leftarrow$ ROLLOUT($x, B_x^{\bar{\pi}}$)
  $r_g \leftarrow \frac{1}{M} \sum_{m=1}^{M} J(x, a_m) - \bar{s}$ with $a_m \sim \bar{\pi}(\cdot \mid x, e_g)$ if $v_g = \text{valid}$, else $r_g \leftarrow -10$
end for
$A_g \leftarrow (r_g - \text{mean}_{g'}\, r_{g'})/(\text{std}_{g'}\, r_{g'} + \epsilon)$ ▷ Eq. (4)
add each segment $\sigma_{g,n}$ to the batch as its own sequence with advantage A_g; mask tool outputs
update ϕ on generated tokens with Eq. (5) ▷ Stage I: valid h only, max. likelihood

- **Rubric distillation** conditions a frozen teacher on the rubric and distills its token distribution on student-generated prefixes. We follow it with answer-reward RL, which the original method does not use.
- **Think-with-Rubrics** keeps its published reward $R = 0.3R_{\text{gold}} + 0.5R_{\text{self}} + 0.2R_{\text{fmt}}$ and trains with DAPO (Yu et al., 2025) after rubric–answer SFT.
- **EL-style distillation** writes an experience from one current response and its feedback, internalizes it through on-policy context distillation, and is then followed by answer-reward RL, which the original method does not use.
- **Group refinement** is a GOLF-inspired variant (Huang et al., 2026). GOLF refines failed attempts from group-level feedback inside RL; our variant instead drafts one skill from eight scored attempts, tests it on four fresh attempts, and refines it once before experience–answer collection and joint RL.
- **Full-context coach** receives the task, the rubric, and every scored attempt in one context and writes an experience without tool calls. It is warmed up on the same tasks and trained with the same Stage II reward, and its experience is internalized with the same recipe as Agent as Coach.
- **Prompted coach** and **Coach-Zero** use the same harness as Agent as Coach. The prompted coach is not trained, and Coach-Zero stops after Stage I.
- **Self-written experience RL** starts from SFT with the experience–answer format of Stage IV and is trained with answer-reward GRPO, without any coach data.

Table 5: **Training hyperparameters.**

Part	Setting	Value
Harness	Coach initialization	base model of its student
	Context window	32,768 tokens
	Limits	30 tool turns, no cap on reads, 8,192 evidence tokens
	Generated tokens	2,048 per trajectory, experience at most 2,048
Stage I	Data	5,000 SFT tasks, 3 epochs
	Optimizer	AdamW, learning rate 10^{-6}, batch 32, cosine decay
Stage II	Data	27,000 RL tasks, 2 epochs
	Rollouts	pool size $K = 8$, $G = 8$ trajectories, $M = 4$ guided answers
	Optimizer	GRPO, learning rate 10^{-6}, clip 0.2, KL 10^{-3} to Stage I
	Sampling	temperature 1.0, student answers up to 2,048 tokens
Stage III	Data	5,000 SFT tasks, $K = M = 8$, 3 epochs
	Optimizer	AdamW, learning rate 10^{-6}, batch 32
Stage IV	Data	27,000 RL tasks, 2 epochs, 8 outputs per question
	Optimizer	GRPO, learning rate 10^{-6}, KL to Stage III
Judge	Model	GPT-4o, temperature 0, integer score 1–10

B Additional results

B.1 Student behavior (RQ2)

Value of evidence. Let $\bar{\nu}(c; Z)$ be the mean coaching gain of coach c when it sees behavior Z together with the task and the rubric. With B the pool of eight scored answers, the value of evidence and its change under utility training are

$$\mathrm{EVE}(c) = \bar{\nu}(c; B) - \bar{\nu}(c; \varnothing), \qquad \Delta_{\mathrm{learn}} = \mathrm{EVE}(c_{\phi^{\mathrm{II}}}) - \mathrm{EVE}(c_{\phi^{\mathrm{I}}}), \tag{8}$$

where $\varnothing$ removes the answers and their scores, so the coach writes an experience from the rubric alone, and ϕ^{I} and ϕ^{II} are the coach parameters after Stages I and II. In Table 2, EVE is 0.69 for the trained coach and 0.45 for Coach-Zero, so $\Delta_{\mathrm{learn}} = 0.24$.

B.2 Agentic reading (RQ3)

Pool sweep protocol. Figure 1, Figure 3a, and Table 3 use the 500 InFoBench instructions with Qwen3-8B-Base as the frozen student. Each instruction has a pool of up to 128 scored answers from the student in random order, and smaller pools are nested prefixes of larger ones. Each coach writes one experience per pool, and the student then samples 32 guided answers at temperature 1.0. Guided DRFR is the fraction of decomposed requirements these answers satisfy, judged by GPT-4o. The rubric given to the coach is derived from the instruction alone, and the official decomposed questions are never shown to the coach or the student. Rubric in context gives the student this rubric in place of an experience. The full-context coach is untrained, Coach-Zero has only the Stage I warm-up, and Agent as Coach is trained at $K = 8$. All three stay fixed across the sweep. The mixed-K full-context coach in Table 3 is trained with the same Stage II reward on pool sizes sampled from $\{8, 16, 32\}$.

B.3 Internalization (RQ4)

Fixed answer targets. The fixed-target arms of Figure 4b share one bank of (x, e, a) triples that Stage III builds from the 5,000 SFT tasks for Qwen3-8B-Base. All arms keep the same question–answer pairs and change only the prefix before the answer. The generic plan is written from the question alone and matched in length. Other-task experience comes from a different task of similar length. Redaction replaces entities, quantities, quoted answer spans, and final decisions with typed placeholders. EL receives the same experience and trains the student with on-policy reverse KL toward a teacher that sees the experience. Table 6 lists the values.

Table 6: **Experience content with answer targets fixed.** Question-only average (%) of Qwen3-8B-Base after SFT.

Prefix in the training target	Average	Change
Full experience (Stage III)	61.4	—
Redacted experience	61.0	−0.4
Generic plan	60.0	−1.4
No experience	59.4	−2.0
Other-task experience	58.8	−2.6
EL, same experience	60.8	−0.6

C WHY STUDENT BEHAVIOR ADDS INFORMATION

This appendix formalizes why scored student behavior can help a coach beyond the task and the rubric. For the bound, we rescale the judge score to $[0, 1]$.

Consider transfer to a student policy S whose identity is unknown to an ideal coach. Let X contain the task and rubric, let B denote observed student behavior, and let $\mathcal{E}_x$ be a finite nonempty set of experiences available to both informed and uninformed coaches. Set $f_x(S, e) = U_S(e; x) \in [0, 1]$. An oracle knows the joint distribution and chooses after observing either X or (X, B). Define

$$V_0 = \mathbb{E}_X \max_{e \in \mathcal{E}_X} \mathbb{E}[f_X(S, e) \mid X],$$
$$V_B = \mathbb{E}_{X,B} \max_{e \in \mathcal{E}_X} \mathbb{E}[f_X(S, e) \mid X, B], \qquad \mathrm{VOI}(B) = V_B - V_0. \tag{9}$$

Replacing f with gain leaves the difference unchanged because the unaided score is independent of the selected experience.

Proposition 1 (Value and information bound)**.** *Assume regular conditional distributions exist. With natural logarithms in mutual information and* $\mathrm{TV}(P, Q) = \sup_A |P(A) - Q(A)|$*,*

$$0 \le \mathrm{VOI}(B) \le \mathbb{E}_{X,B}\, \mathrm{TV}\left(P_{S|X,B}, P_{S|X}\right) \le \sqrt{\tfrac{1}{2} I(S; B \mid X)}. \tag{10}$$

If $B' = (B, W)$ *supplies additional observations and leaves the feasible experience set unchanged, then* $\mathrm{VOI}(B') \ge \mathrm{VOI}(B)$*.*

Proof. The informed oracle can choose the same maximizing experience as the uninformed oracle, establishing nonnegativity. For each (x, b), abbreviate the posterior by P and the prior conditional on x by Q. Since $f_x(s, e) \in [0, 1]$,

$$\max_e \mathbb{E}_P f_x(S, e) - \max_e \mathbb{E}_Q f_x(S, e) \le \max_e \left(\mathbb{E}_P f_x(S, e) - \mathbb{E}_Q f_x(S, e)\right) \le \mathrm{TV}(P, Q).$$

The first inequality follows by evaluating the second maximum at a maximizer of the first; the second is the bounded-function characterization of total variation. Averaging yields the first upper bound. Pinsker's inequality gives $\mathrm{TV}(P, Q) \le \sqrt{\mathrm{KL}(P\|Q)/2}$; Jensen's inequality and the conditional-mutual-information identity yield the final bound. If the divergence is infinite, that bound is vacuous but valid. Finally, every decision based on (X, B) is feasible after observing (X, B, W), because the oracle can ignore W. Optimizing over this larger decision class proves monotonicity. □

The bound shows that behavior can help only when it reveals something about the student that the task and the rubric do not. A learned coach must still extract this information from the pool, which is what the harness and Stage II training provide.

D A COMPLETE WORKED EXAMPLE OF EXPERIENCE

This illustrative example (Figure 6) follows one task and rubric across three kinds of guidance. The figure shows only the line-ending words, where the failures are immediately visible, and abbreviates the complete experience of Appendix D.3 to its three steps.

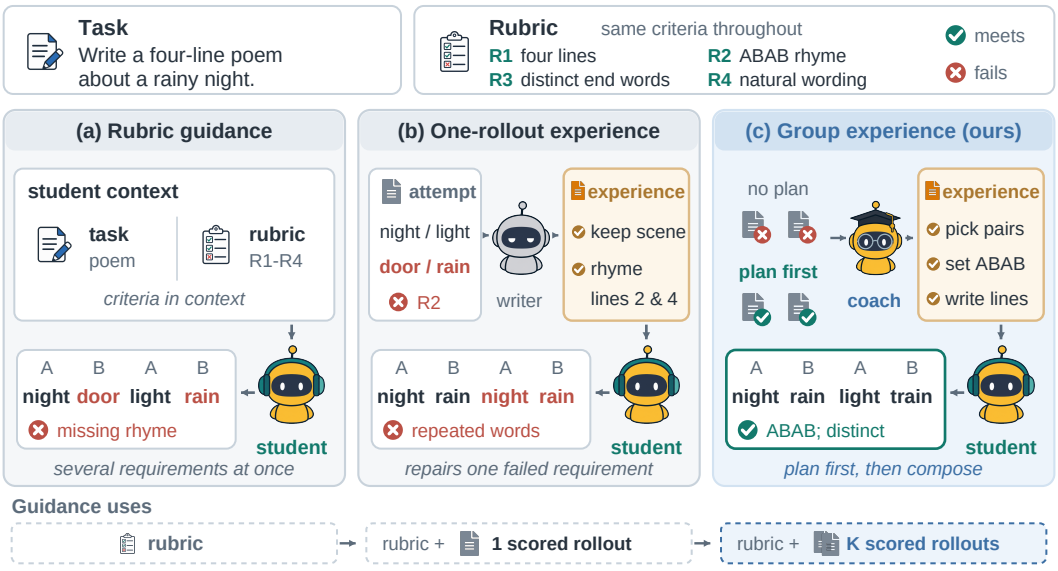


Figure 6: **From rubric requirements to a generation procedure (constructed illustration).** With the rubric fixed, (a) direct writing misses rhyme, (b) local repair repeats end words, and (c) group experience suggests planning rhyme pairs before writing.

D.1 The task, rubric guidance, and a local repair

Task and rubric

Task instruction. Write a four-line English poem about a rainy night. Use an ABAB rhyme scheme with four different line-ending words, and keep the language natural.

The same evaluation rubric for every poem

R1. The poem contains exactly four lines.
R2. The line endings follow ABAB: lines 1 and 3 rhyme, lines 2 and 4 rhyme, and the A and B rhyme sounds differ.
R3. All four line-ending words are different; repeating the same word does not satisfy the rhyme requirement.
R4. The poem evokes a rainy night in natural, coherent language.

(a) Rubric guidance

Context. The student receives the task and R1–R4.

Rain taps the rooftops through the night.
A traveler waits beside the door.
The station windows spill their light.
A whistle trembles through the rain.

Feedback. The poem evokes a rainy night in natural wording, but lines 2 and 4 do not rhyme. R2 is not met.

(b) Experience from one evaluated rollout

Evidence. The writer receives the same task and rubric, plus the poem in (a) and its feedback.
Experience. The previous poem failed because its second and fourth lines did not rhyme. Keep the rainy-night scene and repair those two line endings so that they rhyme with each other.

Rain taps the rooftops through the night.
The empty platform shines with rain.
A whistle echoes in the night.
The station windows blur with rain.

Feedback. The local repair yields repeated end words. Under the unchanged rubric, repeating night and rain fails R2 and R3.

The local instruction addresses a visible rhyme failure, while the next poem violates another requirement by repeating its end words. Both failures are identified by the original rubric. This example motivates coordinating requirements before committing to complete lines.

D.2 The expanded behavior pool

In this illustration, the coach receives four rollouts A1–A4 with feedback under R1–R4. A1 and A2 are written directly without a plan, while A3 and A4 begin with the plans printed below. The rubric applies only to the final poem and gives no separate credit for a plan.

A1: written directly

Rain taps the rooftops through the night.
A traveler waits beside the door.
The station windows spill their light.
A whistle trembles through the rain.

Rubric feedback. R1, R3, and R4 are met. R2 is not met: night/light rhyme, but door/rain do not.

A2: written directly

The city sleeps beneath the rain.
A lamp still glimmers through the night.
The gutters murmur with the rain.
One window holds a fading light.

Rubric feedback. R1 and R4 are met. R2 and R3 are not met: night/light rhyme, but lines 1 and 3 both end with rain, so the A pair repeats a word instead of rhyming.

A3: rhyme plan first

Visible plan. Choose night/light and rain/train. Set the line endings to night, rain, light, train, and then write a rainy-night scene around them.

The station clock shines in the night.
A whistle trembles through the rain.
Wet rails reflect its silver light.
Beyond the bridge rolls one last train.

Rubric feedback. R1–R4 are met: four natural lines evoke a rainy night and use two distinct rhyme pairs with four different end words.

A4: rhyme plan first

Visible plan. Choose street/feet and glow/flow. Set the line endings to street, glow, feet, flow, and then build the scene around them.

Rain threads the dark and empty street.
The windows cast a muted glow.
Cold puddles gather at my feet.
Along the curb, the gutters flow.

Rubric feedback. R1–R4 are met: the four-line rainy-night scene is coherent, street/feet and glow/flow form an ABAB pattern, and no end word repeats.

Table 7: **The same four criteria applied to each rollout.** R1 four lines, R2 ABAB rhyme, R3 distinct end words, R4 theme and natural wording.

Rollout	Visible rhyme plan	R1	R2	R3	R4
A1	Absent	Met	Not met	Met	Met
A2	Absent	Met	Not met	Not met	Met
A3	Present	Met	Met	Met	Met
A4	Present	Met	Met	Met	Met

A1 keeps one rhyme pair but misses the other, and A2 repeats an end word. A3 and A4 use different vocabulary but share one explicit procedure, which fixes two rhyme pairs before composing sentences. This comparison suggests a candidate strategy for the coach to encode.

D.3 Experience from the group and its use

(c) Experience from the group

Evidence. The coach receives the task, R1–R4, and all four rollouts above with their feedback.

Applies when. Writing a poem with a required rhyme scheme and a specified theme.
Do. Before composing sentences, choose two pairs of different rhyming words that fit the theme. Assign the words to the line endings in ABAB order, then write each line around its assigned ending.
Avoid. Do not rely on repeating an end word or patching the last line after committing to all the preceding lines.
Check. Read the end words alone to check the rhyme scheme and word distinctness. Then read the whole poem for natural wording, thematic coherence, and the required line count.

(c) The student applies the experience

Visible plan. Rhyme pairs night/light and rain/train, with endings night, rain, light, train in ABAB order.

Rain taps the rooftops through the night.
I watch the windows blur with rain.
A platform glows with pools of light.
Then silence follows the last train.

Rubric feedback. R1–R4 are met. The poem uses four different end words in ABAB order and maintains a natural rainy-night scene.

What information the experience adds. The rubric states the properties a completed poem should have. The group of answers shows how to coordinate them. Choosing the line endings together resolves their cross-line rhyme dependencies before sentence composition. The experience encodes this ordering and a final check. It contains neither a completed poem nor case-specific rhyme words.