ZONE OF PROXIMAL POLICY OPTIMIZATION: TEACHER IN PROMPTS, NOT GRADIENTS

Anonymous authors
Paper under double-blind review

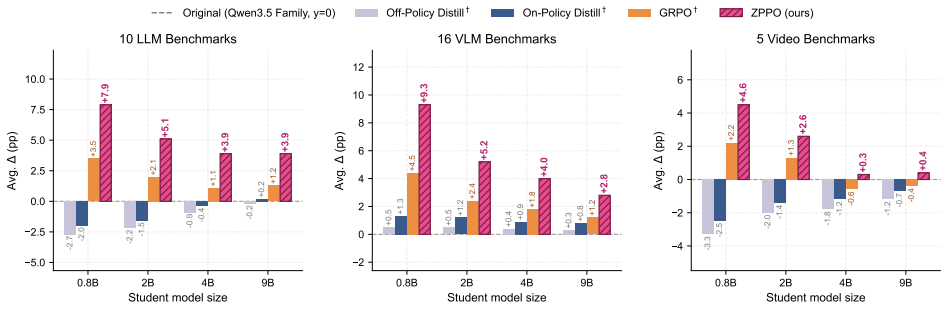


Figure 1: Macro-average gain Δ (pp) on 10 LLM (left), 16 VLM (middle), and 5 Video benchmarks (right), across the four Qwen3.5 student scales. All methods are trained on the same multimodal dataset for vision-language models (VLMs) and evaluated on various benchmarks for generalization. † denotes augmentation with the prompt replay buffer.

ABSTRACT

Knowledge distillation transfers a teacher's competence to a small student but is brittle in the small-student regime: forcing the student to imitate logits from a much larger teacher hurts generalization on benchmark families beyond the training corpus. Reinforcement learning (RL) avoids logit imitation by training on the student's own rollouts. However, on questions where every rollout fails—yielding zero advantage and being silently discarded—injecting a stronger teacher's response into the policy gradient breaks the on-policy assumption and induces drift. We introduce **Zone of Proximal Policy Optimization** (ZPPO), inspired by Vygotsky's *zone of proximal development*. ZPPO keeps the teacher inside the *prompt* rather than the policy gradient. On hard questions, where the student's mean rollout accuracy is below half, ZPPO constructs two reformulated prompts. A *Binary Candidate-included Question* (BCQ) pairs one correct teacher response with one incorrect student response as anonymized candidates the student uses as references. A *Negative Candidate-included Question* (NCQ) aggregates the student's wrong rollouts into a single prompt to surface their shared failure modes. A prompt replay buffer keeps each hard question eligible for re-sampling until it either graduates—the student's mean rollout accuracy on it reaches half or more—or is FIFO-evicted under finite capacity, amplifying BCQ and NCQ inside the student's current zone of proximal development. We post-train Qwen3.5 students at four scales (0.8B–9B) as vision–language models with a 27B teacher and evaluate them on a 31-benchmark suite (16 VLM, 10 LLM, 5 Video); ZPPO outperforms off/on-policy distillation and GRPO, with the largest gains at the smallest scale.

1 INTRODUCTION

Large language models (LLMs) and vision–language models (VLMs) have made rapid progress (Hurst et al., 2024; Team et al., 2023; Anthropic, 2024; Qwen Team, 2026; Guo et al., 2025) on broad benchmarks, with reinforcement learning post-training (Guo et al., 2025; Kimi Team et al., 2025; Liu et al., 2025c; Yu et al., 2025; He et al., 2025; Khatri et al., 2025). The cost of those gains, however, has scaled with the cost of the model: tens of billions of activated parameters. Models of that size do not fit the regime that matters for deployment on mobile phones, AR/VR glasses, or embodied robots (Lee et al., 2025b;a; 2026; Marafioti et al., 2025; Vasu et al., 2025; Qu et al., 2025), where compute and memory budgets are much smaller.

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.

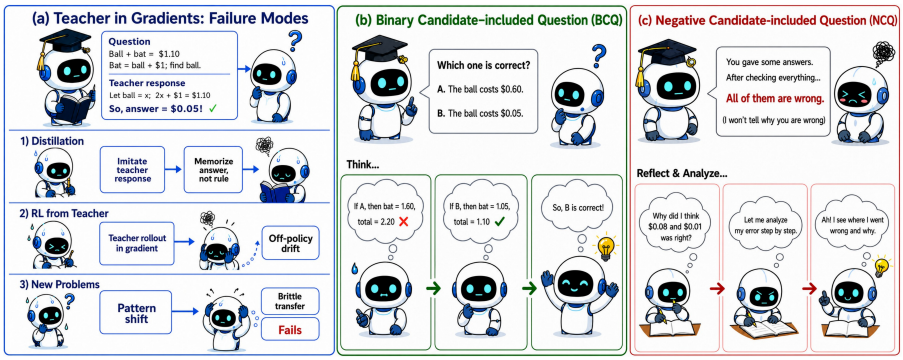


Figure 2: (a) Failure modes that motivate ZPPO. In the small-student regime, fitting the student to a much larger teacher's logits is brittle; on the RL side, injecting a teacher response into the policy gradient as if it were on-policy introduces drift. (b) BCQ recovers a learning signal on hard questions by pairing a correct teacher response with a wrong student response as anonymized in-prompt candidates, asking the student to solve the problem using them as references. (c) NCQ aggregates the student's own wrong rollouts into a single prompt, surfacing shared error patterns.

Closing this deployment gap requires *small* models that generalize broadly, not narrow specialists. The standard knowledge-transfer recipe is *knowledge distillation*: a small student is trained to imitate the logits, hidden states, or sampled outputs of a strong teacher (Hinton et al., 2015; Gou et al., 2021; Sanh et al., 2019; Gu et al., 2024; Ko et al., 2024; Wu et al., 2024). Despite its effectiveness, distillation carries a structural limitation that becomes severe when the student is much smaller than the teacher. Logit imitation is, at its core, supervised fitting on the teacher's distribution: with limited training data, students memorize teacher answers and forget general capabilities (Chu et al., 2025; Chen et al., 2025b; Huan et al., 2025). This limitation grows with the teacher–student capacity gap (Cho & Hariharan, 2019; Mirzadeh et al., 2020; Zhang et al., 2023b;a), especially for students of 3B parameters or fewer (Li et al., 2025c; Ren et al., 2026). The result is a student whose generalization on benchmark families beyond the training corpus is brittle, and whose responses rarely venture beyond the paths the teacher has already taken. Our experiments reproduce this pattern (Sec. 4.1).

A line of work tries to alleviate these issues by using the student's own rollouts to conduct distillation from the teacher to the student (*on-policy* distillation (Agarwal et al., 2024; Lu & Thinking Machines Lab, 2025; Fu et al., 2026; Bousselham et al., 2025; Yang et al., 2026b; Zhang et al., 2026a; Wu et al., 2026a)) or by removing the external teacher and letting the student teach itself (*self*-distillation (Shenfeld et al., 2026; Zhao et al., 2026; Hübotter et al., 2026; Yang et al., 2026a)). These reduce the teacher–student mismatch, but most share the same fundamental constraint: the training signal is still a logit distribution that the student must match (Li et al., 2026b; Song & Zheng, 2026). For 0.8B or 2B students, model capacity is too limited to absorb such a distribution broadly (Zhang et al., 2023b;a; Li et al., 2025c), and the resulting models lose generalization beyond the training corpus (Kim et al., 2026; Li et al., 2023).

A natural alternative is reinforcement learning (RL) (Guo et al., 2025; Shao et al., 2024; Hu et al., 2025a; Yu et al., 2025; Liu et al., 2025c; He et al., 2025; Khatri et al., 2025; Liu et al., 2025a; Hu et al., 2025b; Liu et al., 2025b; 2026a), which avoids logit imitation entirely and trains directly on the student's own rollouts. However, RL has its own pathology in the small-model regime: hard questions with zero successful rollouts produce zero advantage and are silently discarded, so the student never learns from the questions where it most needs help. A simple fix—using a stronger teacher's correct response as if it were the student's—breaks the on-policy assumption and induces policy drift, because the teacher's response lies far outside the student's rollout distribution. Fig. 2a summarizes these two failure modes.

In this paper, we ask a single question: *how can we transfer the teacher's knowledge to the student without ever imitating the teacher's logits or injecting the teacher's response directly into the student's policy gradient?* Our answer, which we call **Zone of Proximal Policy Optimization** (ZPPO), borrows its name from Vygotsky's *zone of proximal development* (Vygotsky & Cole, 1978)—the band of tasks a learner cannot yet solve alone but *can* solve with a small amount of guidance from a more capable peer. Concretely, ZPPO keeps the teacher only inside the *prompt*, never inside the student's policy gradient (Fig. 2b,c and Fig. 3):

- For each hard question (defined as one where the student's mean rollout accuracy is below half), we construct a reformulated prompt—a *Binary Candidate-included Question* (BCQ)—that appends one correct teacher response and one wrong student response as shuffled, anonymized candidates,

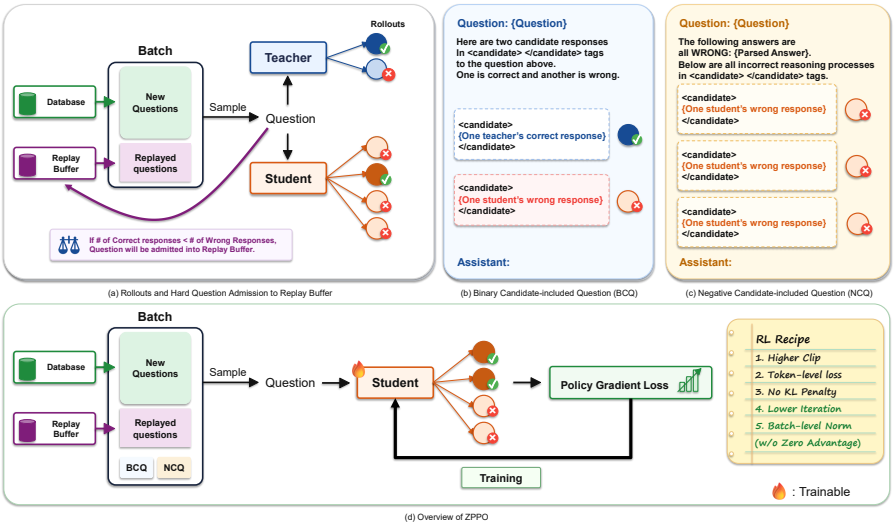


Figure 3: **Overview of ZPPO.** (a) Hard questions (mean rollout accuracy below half) are admitted to the prompt replay buffer. (b) BCQ pairs one correct teacher response with one wrong student response. (c) NCQ aggregates the student's wrong rollouts into a single prompt. (d) Within one step, the plain groups and the BCQ and NCQ groups built from them enter the same policy gradient update, with the RL recipe.

and asks the student to solve the problem afresh using them as references. Since the student generates its own response, its policy gradient remains on-policy at the response level.

- In parallel, we construct a *Negative Candidate-included Question* (NCQ) that appends *all* of the student's wrong responses, explicitly showing that all of them are wrong. In a standard rollout group each wrong rollout contributes its advantage independently, so NCQ is the place where a question's failed attempts become collectively visible, cueing the student to recognize the shared error patterns and avoid them.
- Hard questions are also pushed into a *prompt* replay buffer (no responses are stored). They are re-sampled until they either *graduate* (rollout accuracy reaches half, i.e. $\bar{r}_x \geq 0.5$) or are FIFO-evicted once the buffer reaches capacity (Sec. 3.3, Appendix G.2). This focuses repeated practice on prompts inside the student's zone of proximal development.

These three components (BCQ, NCQ, and prompt replay buffer) are layered on top of GRPO (Shao et al., 2024) augmented with three DAPO ingredients (Yu et al., 2025): clip-higher, a token-level policy gradient loss, and removal of the reference-KL penalty. We also adopt two simple but easily mishandled recipe choices that materially affect a small model's training dynamics: *iterations per step* and *batch-level advantage normalization* of REINFORCE++ (Hu et al., 2025a), with zero-advantage groups excluded from the batch statistics. Both are low-cost, yet ablating them accounts for a sizeable fraction of the performance drop we observe, despite their simplicity (Appendix G.5).

We instantiate ZPPO on the Qwen3.5 family at four student scales—0.8B, 2B, 4B, and 9B—with a 27B teacher (Qwen Team, 2026), post-train each student as a VLM on a multimodal RL dataset of image–question pairs, and evaluate on a 31-benchmark suite spanning 16 VLM, 10 LLM, and 5 Video benchmarks (Appendix C.2). At every scale, ZPPO outperforms (i) off- and on-policy distillation and (ii) GRPO (Fig. 1), and ablating any single component—BCQ, NCQ, or the prompt replay buffer—strictly lowers every family's macro average at every scale (Tab. 3, Tab. 15, Tab. 18). The improvements are largest precisely where the small-model gap is widest: measured against the base student, a 0.8B student gains $+9.3$ pp on the VLM benchmarks, while a 9B student gains $+2.8$ pp. Crucially, ZPPO also improves on the LLM and Video benchmark families that lie beyond the training corpus—the very families where distillation hurts.

DISTILLATION HURTS GENERALIZATION BEYOND THE TRAINING CORPUS; ZPPO DOES NOT

16 VLM benchmarks (training-aligned): Distillation lifts the student only marginally—averaging Off-Distill† and On-Distill† gives $+0.9/+0.9/+0.7/+0.6$ pp at 0.8B/2B/4B/9B (Tab. 1, Tab. 13), while ZPPO gains $+9.3/+5.2/+4.0/+2.8$ pp over the untrained base.
10 LLM and 5 Video benchmarks (beyond the training corpus): Distillation *degrades* the same students by $-2.5/-1.8/-0.9/-0.3$ pp (Off-Distill† and On-Distill† averaged; Tab. 2, Tab. 14), whereas ZPPO *improves* them by $+6.8/+4.3/+2.7/+2.7$ pp.

Table 1: Main results on the 16 VLM benchmarks at 0.8B and 2B; † denotes augmentation with the prompt replay buffer, and Δ reports ZPPO minus the strongest non-ZPPO method in each column on the same model size. Bold marks the column maximum and is shared by every row that attains it; the colored Δ row is a difference, not a maximum. 4B and 9B results: Tab. 13. Benchmark abbreviations: Appendix C.2.

16 VLM Benchmarks	AI2D	BabyV	CharXiv	DynaM	EmbSp	InfoVQA	MVerse	MVision	MVista	MMMU^{Pro}	MM-Vet	OCR^{EN}	OCR^{ZH}	VisP	VBlind	WeMath	Avg
Qwen3.5-0.8B	65.6	6.7	54.3	17.8	67.9	68.6	43.5	16.4	60.7	26.8	53.2	40.0	17.0	20.5	42.8	54.4	41.0
+ Off-Distill	66.3	6.7	55.1	19.4	65.8	66.0	45.3	17.3	62.0	27.6	51.5	38.9	16.2	20.0	41.8	55.9	41.0
+ Off-Distill†	66.7	6.7	56.0	20.1	65.8	66.8	45.8	17.9	62.2	29.0	51.8	39.5	15.9	20.8	42.2	56.4	41.5
+ On-Distill	67.6	7.5	56.0	20.0	66.7	67.1	45.8	19.4	62.7	28.2	52.7	39.6	16.7	21.2	41.5	57.7	41.9
+ On-Distill†	68.6	7.8	56.4	21.1	67.1	67.6	47.4	18.7	63.6	28.8	52.5	39.1	16.7	20.9	42.3	58.6	42.3
+ GRPO	69.5	8.6	57.0	21.3	69.2	71.0	47.7	20.1	65.2	29.9	55.6	40.9	17.0	24.2	43.2	60.6	43.8
+ GRPO†	71.2	9.8	59.9	23.6	69.4	72.4	51.1	20.9	68.3	30.5	57.5	41.3	17.5	27.8	43.6	62.5	45.5
+ ZPPO	**76.5**	**13.9**	**63.9**	**31.1**	**71.5**	**75.3**	**59.3**	**29.2**	**73.2**	**37.6**	**59.9**	**42.5**	**18.7**	**35.0**	**44.7**	**71.7**	**50.3**
Δ	+5.3	+4.1	+4.0	+7.5	+2.1	+2.9	+8.2	+8.3	+4.9	+7.1	+2.4	+1.2	+1.2	+7.2	+1.1	+9.2	+4.8
Qwen3.5-2B	81.9	11.6	71.6	41.1	78.2	81.2	69.7	38.4	78.6	46.2	69.7	44.7	24.0	38.3	55.2	77.9	56.8
+ Off-Distill	82.5	11.9	71.0	43.2	77.2	80.2	70.8	40.4	77.9	47.4	69.2	44.4	23.0	38.4	53.7	79.5	56.9
+ Off-Distill†	82.9	12.5	71.4	43.7	77.4	80.4	71.4	41.8	77.8	47.9	69.2	44.5	23.7	37.8	54.6	79.3	57.3
+ On-Distill	82.6	13.1	71.7	44.5	77.6	81.1	71.9	41.9	78.2	48.8	69.2	45.3	23.9	38.8	54.9	79.3	57.7
+ On-Distill†	83.4	12.8	71.7	44.9	77.5	80.9	72.0	42.7	77.9	49.3	69.4	45.5	24.5	38.8	55.6	80.6	58.0
+ GRPO	83.5	13.7	72.8	45.6	78.3	82.4	72.3	43.1	79.0	49.3	72.3	46.1	24.9	39.2	56.5	80.4	58.7
+ GRPO†	83.8	14.4	73.0	45.9	78.7	83.1	72.8	43.4	79.3	49.6	74.0	46.4	25.2	40.3	57.0	80.7	59.2
+ ZPPO	**85.3**	**18.6**	**73.9**	**52.7**	**79.5**	**84.6**	**76.0**	**50.5**	**80.5**	**53.2**	**77.1**	**48.8**	**26.0**	**42.0**	**60.5**	**82.6**	**62.0**
Δ	+1.5	+4.2	+0.9	+6.8	+0.8	+1.5	+3.2	+7.1	+1.2	+3.6	+3.1	+2.4	+0.8	+1.7	+3.5	+1.9	+2.8

Table 2: Evaluation over the 10 LLM and 5 Video benchmarks (probing generalization beyond the training data). Δ is ZPPO minus the strongest non-ZPPO row per column. Bold marks the column maximum, shared by every row attaining it; the colored Δ row is a difference. 4B and 9B: Tab. 14.

	10 LLM Benchmarks											5 Video Benchmarks					
Method	AIME25	AIME26	CEval	GPQA-D	HLE	IMO-AB	MMLU	MMLU-Pro	MMLU-Rd	MultiCh	Avg	MMVU	MVBench	VMME	VMME^{S}	VMMMU	Avg
Qwen3.5-0.8B	**3.3**	0.0	52.3	12.1	2.2	3.3	57.7	41.7	62.5	16.6	25.2	39.0	51.6	50.3	51.6	48.8	48.3
+ Off-Distill†	0.0	0.0	47.9	10.1	2.1	2.9	52.6	38.3	56.9	14.7	22.5	35.4	48.9	46.8	48.5	45.4	45.0
+ On-Distill†	0.0	0.0	49.2	10.6	2.8	4.0	53.8	38.8	58.2	14.9	23.2	36.8	49.2	47.8	49.6	45.7	45.8
+ GRPO†	**3.3**	0.0	54.5	25.5	3.5	7.0	59.9	47.1	65.1	20.8	28.7	41.3	53.5	52.2	53.0	52.5	50.5
+ ZPPO	**3.3**	0.0	**57.6**	**42.4**	**5.0**	**10.5**	**63.0**	**51.7**	**69.0**	**28.6**	**33.1**	**42.5**	**56.6**	**53.9**	**54.9**	**56.4**	**52.9**
Δ	**0.0**	**0.0**	+3.1	+16.9	+1.5	+3.5	+3.1	+4.6	+3.9	+7.8	+4.4	+1.2	+3.1	+1.7	+1.9	+3.9	+2.4
Qwen3.5-2B	**26.7**	30.0	71.1	53.5	3.5	11.3	74.6	66.2	80.4	35.8	45.3	53.7	60.7	59.9	61.2	67.6	60.6
+ Off-Distill†	**26.7**	26.7	68.2	50.3	2.0	10.9	71.2	63.8	77.5	34.0	43.1	51.5	59.2	57.9	59.0	65.4	58.6
+ On-Distill†	23.3	30.0	69.3	51.9	3.8	11.5	72.2	63.8	77.9	34.0	43.8	52.4	59.9	58.6	59.4	65.5	59.2
+ GRPO†	**26.7**	33.3	72.4	56.5	4.2	19.3	75.4	67.3	81.4	37.3	47.4	56.7	61.7	61.2	62.6	67.3	61.9
+ ZPPO	**26.7**	**40.0**	**74.0**	**60.6**	**5.1**	**29.5**	**76.4**	**68.2**	**82.9**	**40.6**	**50.4**	**59.0**	**62.7**	**62.4**	**63.9**	**68.1**	**63.2**
Δ	**0.0**	+6.7	+1.6	+4.1	+0.9	+10.2	+1.0	+0.9	+1.5	+3.3	+3.0	+2.3	+1.0	+1.2	+1.3	+0.5	+1.3

2 Related Work

Four lines of work bear on ZPPO. *Knowledge distillation* (Gou et al., 2021; Agarwal et al., 2024; Lu & Thinking Machines Lab, 2025; Shenfeld et al., 2026) turns the teacher's output distribution into the signal the student must match. That signal grows brittle as the teacher–student gap widens (Cho & Hariharan, 2019; Mirzadeh et al., 2020; Li et al., 2025c), inducing memorization of teacher answers and forgetting of general capabilities (Chu et al., 2025; Chen et al., 2025b; Huan et al., 2025). Self-training (Zelikman et al., 2022) needs no teacher distribution but still trains on a fixed text target: the student's own graded-correct outputs. *RL post-training* (Shao et al., 2024; Yu et al., 2025; Guo et al., 2025) trains on the student's own rollouts, but a group of all-wrong rollouts has zero advantage and gives no learning signal. Methods that target such groups reshape advantages (Le et al., 2026; Feng et al., 2025b), interleave supervised updates (Ma et al., 2026; Han et al., 2026), or feed the teacher back into the gradient through hybrid RL+distillation objectives (Xu et al., 2025; 2026) or teacher traces added to the group (Yan et al., 2025). All of them change the loss or the target, never the prompt. *Replay, curricula, and prompt-side guidance* do act on the prompt. Replay and curricula (Zhan et al., 2025; Zhang et al., 2025a;c) re-expose a hard prompt without saying *why* the last attempt failed. Hints (Nath et al., 2025; Chen et al., 2026a; Wang et al., 2026; Xia et al., 2026) supply only a correct cue, with no wrong alternative. Partial solutions (Zhang et al., 2025d;b; Li et al., 2025d; Huang et al., 2025b) hand over a reference solution's opening steps, a start the student could not reach alone. *Learning from mistakes* fine-tunes on corrected solutions, or on correct and wrong answers labeled by an input tag (An et al., 2023; Wang et al., 2024b; Wu et al., 2026b), or reweights wrong responses in the loss (Lin et al., 2026a; Guo et al., 2026). SocraticPO (Liu et al., 2026c) lets the student continue a wrong attempt after teacher guidance, under a decayed reward. ZPPO instead reformulates the question and samples a fresh group from the student's own policy under the full binary reward. Its replay buffer then amplifies BCQ and NCQ inside the student's zone of proximal development (Vygotsky & Cole, 1978). Appendix B gives the full version.

3 Zone of Proximal Policy Optimization

ZPPO is built on top of GRPO (Shao et al., 2024). Algorithm 1 in Appendix D gives the full training step.

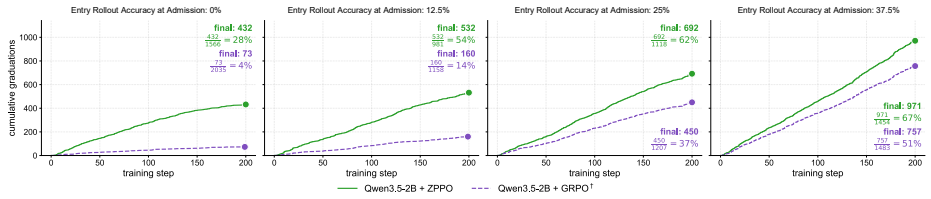


Figure 4: Cumulative graduate counts for ZPPO vs. GRPO† at 2B. The printed ratio is graduated over the entries the buffer re-sampled at least once (Appendix G.2).

3.1 Preliminaries: GRPO's Failure Mode

Let x denote a question and y_{S} a response sampled from a student policy π_θ. For each x we draw a group of G_{S} student rollouts $\{y_{\mathrm{S}}^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot \mid x)$ and assign each an outcome reward $r(x, y_{\mathrm{S}}^{(g)}) \in \{0, 1\}$ that signals whether the final answer is correct. Let $\bar{r}_x$ and std_x denote the within-group mean and standard deviation of $\{r(x, y_{\mathrm{S}}^{(g)})\}_{g=1}^{G_{\mathrm{S}}}$. The standard group-relative advantage (Shao et al., 2024; Yu et al., 2025) is

$$A^{(g)} \;=\; \frac{r(x, y_{\mathrm{S}}^{(g)}) - \bar{r}_x}{\mathrm{std}_x + \epsilon}. \tag{1}$$

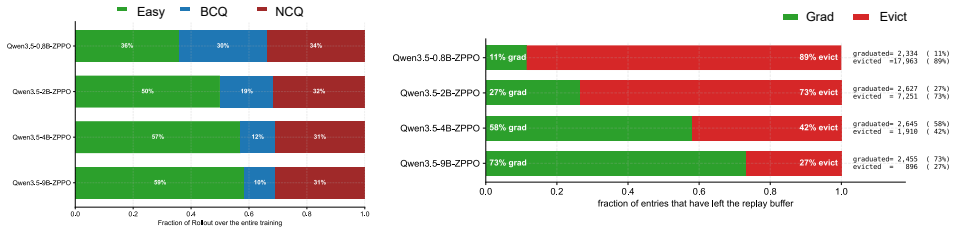
Eq. (1) is the textbook form ZPPO conceptually builds on; our experiments use the two-step REINFORCE++ variant of Hu et al. (2025a) (Step 1 subtracts the per-group mean; Step 2 normalizes across groups, here the non-trivial groups of one mini-batch), restated as Eqs. (2)–(3) in Appendix D. The student update applies the PPO surrogate (Schulman et al., 2017) on top of $A^{(g)}$. Either form leaves a blind spot for small students. Whenever a rollout group is all-wrong ($\bar{r}_x = 0$) or all-correct ($\bar{r}_x = 1$), every advantage in the group is exactly zero, so the question contributes no gradient signal at all. For a small student, the all-wrong case is exactly where teacher guidance could help, and the wrong approach the group agrees on is not even penalized. ZPPO's goal is to recover a learning signal on those hard questions *without ever placing a teacher response in the student's gradient*. We call x a *hard question* when $\bar{r}_x < \tau$ and use $\tau = 0.5$ throughout, the majority-failure boundary, which admits all-wrong and partially solved groups alike (Appendix D; swept in Tab. 10).

3.2 Prompt Reformulation: BCQ and NCQ

Both BCQ and NCQ start from a hard question x on which we have already drawn G_{S} student rollouts $\{y_{\mathrm{S}}^{(g)}\}_{g=1}^{G_{\mathrm{S}}}$. In parallel, we sample G_{T} teacher rollouts on x from a frozen teacher policy π_{T}, score them with the same outcome reward, and keep the correct ones as the pool $\{y_{\mathrm{T}}^{(+)}\}$ from which BCQ draws candidates. We use *on-policy* throughout in the *response-level* sense: every gradient-counted token is sampled by the student, never by the teacher. BCQ and NCQ prompts append teacher-compressed text to the original question, but only as input, never as a gradient target. The same checkpoint also judges free-form answers for every method's binary reward (Appendix E.1). Reformulation is also capped: per rollout step, the combined number of BCQ and NCQ instances stays at or below a fraction ρ_{aug} of the new questions (Sec. 3.3). On every plain rollout, whether the question is new or replayed, training is therefore ordinary on-policy RL on the original prompt. When a question is reformulated, that plain group stays in the same rollout step, and the BCQ or NCQ prompt built from it adds a further group under the same reward and the same parameters. Evaluation always uses the original question (Appendix I.10).

(i) Candidate compression: Before any candidate enters a prompt, the frozen teacher rewrites it under a shared compression prompt and a shared token cap (Fig. 16, Appendix E.1). The rewrite keeps only the essential reasoning steps and the final answer, and drops the exploratory text, self-corrections, and retries of the source rollout; it may not re-derive the problem or add anything the source does not contain. The compressor reads only the source response, never the question or the image.

(ii) Binary Candidate-included Question (BCQ): For each hard question that admits at least one correct teacher response, BCQ uniformly samples one $y_{\mathrm{T}}^{(+)}$ and one wrong student rollout $y_{\mathrm{S}}^{(-)}$. It teacher-compresses both responses, anonymizes them inside identical `<candidate>` tags, and randomly shuffles the order. The result is appended to x together with a single instruction (Fig. 17 in Appendix E.1 reproduces the verbatim code-side template, including the per-candidate `<candidate>` blocks):



(a) BCQ and NCQ Fraction Ratio in A Single Rollout Batch

(b) Graduation and Eviction Ratio in Prompt Replay Buffer

Figure 5: (a) Easy questions ($\bar{r}_x \geq 0.5$) and the BCQ and NCQ reformulations available on hard ones over training, before the caps, per scale, each as a share of their summed count; a hard question can yield both reformulations (Appendix D). (b) Replay-buffer entries that have left the buffer: graduated vs. evicted.

> *Here are two candidate responses in* `<candidate> </candidate>` *tags to the question above. One is correct and another is wrong. Use these as references to help you solve the problem.*

The student then samples a new rollout group $\{y^{(g)}_{\mathrm{BCQ}}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot \mid x_{\mathrm{BCQ}})$ from the reformulated prompt. The intended effect comes from placing a correct teacher response and a wrong student response *side by side*, anonymized, shuffled, and without saying which is correct. The student therefore solves the problem while weighing the two candidates.

(iii) Negative Candidate-included Question (NCQ): For each hard question, NCQ collects every wrong student rollout $y^{(-)}_{\mathrm{S}}$ in the current group whose final answer parses (Appendix D). It lists the *distinct* parsed answers explicitly inside the prompt, so the answer list is generally shorter than the rollout list. Each teacher-compressed reasoning trace is then appended as a `<candidate>` block, with the instruction (full block in Fig. 18, Appendix E.1):

> *The following answers are all WRONG:* ⟨ *parsed answer* ⟩*. Below are the incorrect reasoning processes in* `<candidate> </candidate>` *tags.*

The student then samples a new rollout group $\{y^{(g)}_{\mathrm{NCQ}}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot \mid x_{\mathrm{NCQ}})$, again entirely student-generated. The pedagogical role of NCQ, however, is structurally different. In a standard rollout group each wrong rollout contributes its advantage independently, so NCQ is the first place in our training loop where a question's failed attempts become collectively visible. Confronted with them, the student is cued to recognize the shared error patterns and avoid them.

3.3 Integration with Prompt Replay Buffer

In ZPPO, the prompt replay buffer $\mathcal{B}$ serves *to amplify BCQ and NCQ* on questions the student has not yet mastered. It stores only the question x (image and text), never a response.

(i) Admission and graduation: A question x is *admitted* if its $\bar{r}_x < \tau$. An admitted x leaves $\mathcal{B}$ in one of two ways: it is *graduated* on any later step where $\bar{r}_x \geq \tau$, or it is *FIFO-evicted* once $|\mathcal{B}|$ exceeds $|\mathcal{B}|_{\max}$. We update $\mathcal{B}$ after each training step, and only the original question is stored, never its BCQ or NCQ reformulation. A replayed question that is still hard on its next visit is again eligible for one or both reformulations. The buffer therefore tracks the questions the student cannot yet solve alone, and graduation marks the ones guidance reaches—the student's *current* zone of proximal development.

(ii) Sampling and capacity: Each rollout batch combines new questions from the data loader with replay samples drawn uniformly from $\mathcal{B}$, where the replay count is a fixed fraction ρ_{replay} of the new-question count. From the union of new and replayed questions, BCQ and NCQ are constructed on the hardest ones first, ranked by ascending $\bar{r}_x$. The *combined* BCQ+NCQ count per rollout step is then capped at a fraction ρ_{aug} of the new-question count (Appendix D). The buffer therefore re-exposes hard questions—with freshly sampled BCQ/NCQ candidates on every visit the caps admit—until it lets each one go.

4 Experiments

Students and teacher. We instantiate ZPPO on Qwen3.5 (Qwen Team, 2026) at four student scales (0.8B, 2B, 4B, 9B) with a 27B teacher, post-training each student as a VLM on the ZPPO-77K multimodal RL dataset (Appendix C.1).

Table 3: Component ablation of ZPPO over the 16 VLM benchmarks at 0.8B and 2B. Rows add the prompt replay buffer (†), BCQ, and NCQ to GRPO, alone or combined. Bold marks the column maximum and is shared by every row that attains it. 4B and 9B: Tab. 15.

16 VLM Benchmarks	AI2D	BabyV	CharXiv	DynaM	EmbSp	InfoVQA	MVerse	MVision	MVista	MMMU^{Pro}	MM-Vet	OCR^{EN}	OCR^{ZH}	VisP	VBlind	WeMath	Avg
Qwen3.5-0.8B	65.6	6.7	54.3	17.8	67.9	68.6	43.5	16.4	60.7	26.8	53.2	40.0	17.0	20.5	42.8	54.4	41.0
+ GRPO	69.5	8.6	57.0	21.3	69.2	71.0	47.7	20.1	65.2	29.9	55.6	40.9	17.0	24.2	43.2	60.6	43.8
+ GRPO†	71.2	9.8	59.9	23.6	69.4	72.4	51.1	20.9	68.3	30.5	57.5	41.3	17.5	27.8	43.6	62.5	45.5
+ GRPO + Both	71.7	9.8	58.7	23.9	69.1	72.2	50.7	21.3	67.1	31.7	56.6	40.1	17.1	26.6	43.6	63.3	45.2
+ GRPO† + BCQ	75.2	12.3	61.8	28.2	71.1	74.5	55.9	26.4	71.7	35.2	59.1	42.2	18.4	31.8	44.3	69.6	48.6
+ GRPO† + NCQ	72.0	10.2	60.3	24.3	70.0	73.0	51.5	22.7	68.7	32.1	57.9	41.6	18.1	28.2	44.0	64.6	46.2
+ ZPPO	**76.5**	**13.9**	**63.9**	**31.1**	**71.5**	**75.3**	**59.3**	**29.2**	**73.2**	**37.6**	**59.9**	**42.5**	**18.7**	**35.0**	**44.7**	**71.7**	**50.3**
Qwen3.5-2B	81.9	11.6	71.6	41.1	78.2	81.2	69.7	38.4	78.6	46.2	69.7	44.7	24.0	38.3	55.2	77.9	56.8
+ GRPO	83.5	13.7	72.8	45.6	78.3	82.4	72.3	43.1	79.0	49.3	72.3	46.1	24.9	39.2	56.5	80.4	58.7
+ GRPO†	83.8	14.4	73.0	45.9	78.7	83.1	72.8	43.4	79.3	49.6	74.0	46.4	25.2	40.3	57.0	80.7	59.2
+ GRPO + Both	83.9	14.3	72.7	46.0	78.1	82.8	72.4	42.7	79.4	49.8	73.2	45.4	24.5	39.6	57.7	79.8	58.9
+ GRPO† + BCQ	84.7	16.7	73.7	49.6	79.2	84.0	74.3	47.3	80.1	51.4	75.8	48.1	25.6	41.1	59.1	81.8	60.8
+ GRPO† + NCQ	84.3	15.8	73.4	48.1	79.0	83.6	73.5	45.7	79.7	50.5	74.9	47.6	25.5	40.9	58.4	81.2	60.1
+ ZPPO	**85.3**	**18.6**	**73.9**	**52.7**	**79.5**	**84.6**	**76.0**	**50.5**	**80.5**	**53.2**	**77.1**	**48.8**	**26.0**	**42.0**	**60.5**	**82.6**	**62.0**

Training pipeline. ZPPO follows GRPO (Shao et al., 2024) with three DAPO ingredients (Yu et al., 2025): clip-higher, token-level policy gradient loss, and removal of the KL penalty against a reference policy. We use $I = 4$ iterations per step (vs. DAPO's $I = 16$) and batch-level advantage normalization (Hu et al., 2025a) with zero-advantage groups excluded from the batch statistics. Appendix A gives the population behind every reported number; the full configuration is in Tab. 11, the definition of I and the corresponding budgets in the *Optimizer partitions* paragraph of Appendix E.2, and the per-run wall-clock in Appendix E.3, where ZPPO takes 7–17 more hours than GRPO†.

Baselines. We compare ZPPO against four baselines spanning the first two lines of Sec. 2; the other two are taken up in Sec. 4.5. **(i)** ***Off-policy distillation*** (Algorithm 2): imitate fresh teacher rollouts drawn online and filtered by correctness on every step. **(ii)** ***On-policy distillation*** (Algorithm 3): the student samples its own response, the teacher re-scores it for target logits, and the student imitates them. **(iii)** ***GRPO*** (Algorithm 4): ZPPO's pipeline without BCQ, NCQ, and the replay buffer. **(iv)** ***GRPO***†: ZPPO's pipeline without BCQ and NCQ. Details in Appendix E.2.

4.1 Main results

Tab. 1 (16 VLM benchmarks) and Tab. 2 (10 LLM and 5 Video benchmarks) report the headline comparison; 4B/9B blocks are in Tab. 13 and Tab. 14 (Appendix F). Fig. 1 plots the buffer-augmented rows of both tables as gains over the untrained base, restated point by point in Appendix F.2. Every score is measured on the candidate-free original prompt, the format of at least four in five training groups (Appendix I.10).

(i) Distillation *hurts generalization beyond the training corpus* (Cho & Hariharan, 2019; Li et al., 2025c; Huan et al., 2025). The best distillation variant stays within 1.3 pp of the original on VLM, while Off- and On-Policy distillation *lose* accuracy on Video and *degrade* the pooled LLM–Video average at every scale (Tab. 2, Tab. 14), sharpest at 0.8B.

(ii) Replay alone is insufficient. Adding the buffer to either distillation or GRPO yields modest Avg gains in Tab. 1 ($+0.5$/$+0.4$/$+1.7$ pp at 0.8B for Off-Distill, On-Distill, GRPO): re-exposure without a new learning signal does not recover the signal lost on hard questions.

(iii) ZPPO improves the macro average at every scale and benchmark family. The Avg Δ row is positive across all blocks of Tab. 1 and Tab. 2, with the largest gains where the small-student gap is widest, and Appendix F.1 counts ZPPO against the base and against GRPO† cell by cell over the four scales' 124 (scale, setting) cells. ZPPO and GRPO† both use the replay buffer, so their gap shows what BCQ and NCQ add. That gap stays positive even when the benchmarks in each family are redrawn at random with repeats allowed (Appendix G.7, Tab. 20), which tests the choice of benchmarks. Appendices F.3 and F.4 decompose the LLM- and VLM-family gains setting by setting.

4.2 Inside the Zone: Admission to Graduation

Appendix G.1 tracks how the buffer fills and drains at every scale, and Appendix G.3 the per-step dynamics behind it.

(i) BCQ and NCQ lift plain-rollout accuracy on the hardest training questions (Fig. 4). On entries admitted at 0% rollout accuracy—where ordinary RL has zero gradient signal—ZPPO graduates 28% of its re-sampled entries (432/1566) vs. GRPO†'s 4% (73/2035); on the next-hardest, 54% vs. 14% (Appendix G.2). Graduation is scored on plain rollouts of the same training questions, so it is a training-side diagnostic; the held-out benchmarks of Sec. 4.1 test generalization.

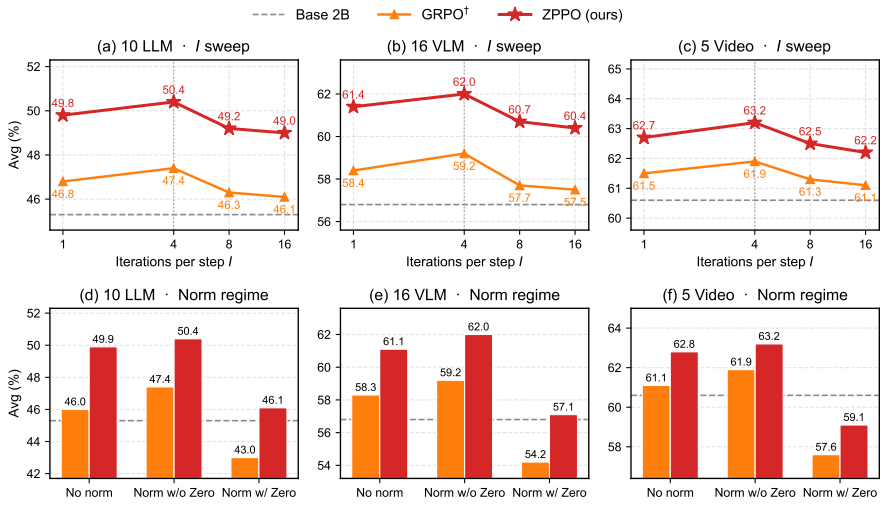


Figure 6: RL recipe ablation at 2B on the 10 LLM (a, d), 16 VLM (b, e), and 5 Video (c, f) benchmarks. **(Top, a–c)** Iterations per step $I \in \{1, 4, 8, 16\}$. **(Bottom, d–f)** Batch-level advantage normalization: *No norm*, *Norm w/o Zero* (ZPPO default), and *Norm w/ Zero*. Dashed gray line marks original 2B.

(ii) Batch and graduation dynamics with scale (Fig. 5). Fig. 5a gives the share of Easy questions and of the BCQ and NCQ reformulations available on hard ones, read before the caps of Algorithm 1 apply. BCQ requires the teacher to succeed where the student fails; as students scale up, the remaining hard questions are increasingly ones the teacher also fails on, so BCQ candidates become scarce (Appendix G.8). In the buffer, most entries are evicted before they graduate at the small scales, and most graduate before they are evicted at the large ones (Fig. 5b).

4.3 Component ablation

Tab. 3 measures each ZPPO component on the 16 VLM benchmarks (4B and 9B in Tab. 15), adding them to GRPO one at a time (*Both* = BCQ + NCQ).

(i) Neither component suffices alone. The buffer on its own (GRPO†) gains a little over GRPO. BCQ+NCQ without the buffer (GRPO + *Both*) does no better than the buffer alone: 45.2 against 45.5 at 0.8B and 58.9 against 59.2 at 2B.

(ii) Replay × reformulation is super-additive. Pairing BCQ *or* NCQ with the buffer beats both the buffer alone (GRPO†) and the reformulations alone (GRPO + *Both*), since each replay visit reformulates the question with a freshly sampled candidate set; Appendix G.4 sizes the interaction and reports the same ablation on the 10 LLM and 5 Video benchmarks.

(iii) The full recipe is strongest at every scale. ZPPO ties or achieves the column max on every benchmark of Tab. 3 and Tab. 15. Within the recipe, BCQ's marginal contribution shrinks with scale (+4.1/+1.9/+1.0/+0.5 pp on VLM) while NCQ's stays inside a 0.7 pp band (+1.7/+1.2/+1.3/+1.0). NCQ therefore carries a growing *share* of a narrowing total, as the BCQ share in Fig. 5a shrinks. BCQ alone leads NCQ alone at 0.8B and 2B, and NCQ alone leads at 4B and 9B (Tab. 3, Tab. 15). Tab. 18 reports the LLM and Video ablation (Appendix G.4), and Tab. 21 bootstraps both marginals.

4.4 Additional RL recipe choices

We isolate two further recipe choices that materially shape small-student RL at 2B across all three benchmark groups in Fig. 6. **(i) Iterations per step balance off-policy drift against rollout cost; $I = 4$ is the sweet spot.** Larger I grows in-step drift: each mini-batch moves the policy further from the rollout policy. DAPO's default $I = 16$ (Yu et al., 2025) runs $4\times$ as many updates as $I = 4$ but lifts GRPO† only marginally over the original 2B base (Fig. 6a–c); $I = 1$ removes drift but undertrains at matched step count, and $I = 4$ peaks on all three families. At matched prompt draws and update count, GRPO† at $I = 4$ is still 0.9/1.7/0.4 pp above $I = 8$ (Appendix G.5). **(ii) Excluding zero-advantage groups from batch normalization (Norm w/o Zero) is critical.** *Norm w/o Zero* (ZPPO default) consistently beats *No norm*, while *Norm w/ Zero* clearly degrades below both (Fig. 6d–f). Zero-advantage groups carry no signal, but replaying hard questions makes them a large, fluctuating share of each mini-batch, so their zeros shrink the batch standard deviation and inflate every nonzero advantage by a factor that varies across mini-batches.

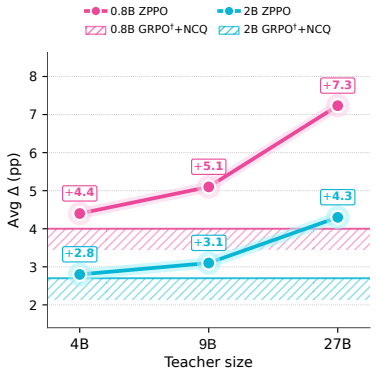


Figure 7: Mean family gain Δ (pp) across teacher size

Table 4: Hint and Prefix methods

Method	Qwen3.5-0.8B LLM	Qwen3.5-0.8B VLM	Qwen3.5-0.8B Video	Qwen3.5-2B LLM	Qwen3.5-2B VLM	Qwen3.5-2B Video
Base	25.2	41.0	48.3	45.3	56.8	60.6
+GRPO†	28.7	45.5	50.5	47.4	59.2	61.9
+GRPO†+Hint	28.8	47.2	50.0	47.6	60.4	61.6
+GRPO†+Prefix	27.1	45.5	49.1	46.3	59.5	61.0
+GRPO†+BCQ	31.5	48.6	52.0	49.1	60.8	62.6
+ZPPO	**33.1**	**50.3**	**52.9**	**50.4**	**62.0**	**63.2**

Table 5: Learning from mistakes methods

Method	Qwen3.5-0.8B LLM	Qwen3.5-0.8B VLM	Qwen3.5-0.8B Video	Qwen3.5-2B LLM	Qwen3.5-2B VLM	Qwen3.5-2B Video
Base	25.2	41.0	48.3	45.3	56.8	60.6
+Correction-SFT†	23.8	41.9	46.7	44.1	57.6	59.3
+Error-reflection-SFT†	24.2	42.2	47.0	44.6	57.9	59.6
+GRPO†	28.7	45.5	50.5	47.4	59.2	61.9
+GRPO†+NCQ	29.4	46.2	51.0	48.5	60.1	62.3
+ZPPO	**33.1**	**50.3**	**52.9**	**50.4**	**62.0**	**63.2**

4.5 Discussion and Analysis

(i) Teacher size matters: A larger teacher widens ZPPO's gain (Fig. 7); a 4B teacher brings it down to the GRPO†+NCQ level (Appendices H.1–H.3). The drop is steepest at 0.8B, where ZPPO's gain is most BCQ-dominated (Sec. 4.3). This is consistent with a weaker teacher solving fewer hard questions and so supplying fewer BCQ candidates (Appendix G.8).

(ii) Hint, Prefix vs. BCQ: We compare BCQ against our implementations of two teacher-guidance families on the shared pipeline, split by what the teacher hands over: *Hint* gives an answer-free cue abstracted from a solution (Nath et al., 2025; Chen et al., 2026a; Wang et al., 2026; Xia et al., 2026; Wu et al., 2025), and *Prefix* gives the opening portion of the solution itself (Zhang et al., 2025d;b; Li et al., 2025d; Huang et al., 2025b;a) (Appendix G.6). Tab. 4 shows that Hint (Algorithm 5; the generator reads the solved response only, Fig. 19) helps VLM but transfers less to the LLM/Video suite. Prefix (Algorithm 6) transfers even less. BCQ is above both, but without its wrong candidate and pair instruction it reaches 0.95 accuracy on its own prompt and still falls 1.1–1.7 pp below GRPO† at 2B (Appendix I.4), so a correct solution alone does not help.

(iii) Learning from mistakes vs. NCQ: Tab. 5 ports the supervised side of the negative-example family—Correction-SFT after LeMa (An et al., 2023) and NAT (Wang et al., 2024b) (Algorithm 7), Error-reflection-SFT after ORION (Wu et al., 2026b) with teacher-side refinement (Algorithm 8); Appendix E.2 gives the executed step of each control. Both lift VLM slightly but land at or below the untrained base on LLM and Video, the same pattern as distillation (Tab. 1, Tab. 2). GRPO† alone clears them; NCQ adds +0.7/+0.7/+0.5 and +1.1/+0.9/+0.4 pp on LLM/VLM/Video at 0.8B/2B and differs on three axes: (a) RL, not supervised fitting, (b) no correct answer shown, and (c) all of the student's wrong answers shown at once.

(iv) Further evidence: Appendix I collects nine controls and two diagnostics; none of the controls changes the ranking. Extra rollouts or replay do not close the gap (Appendices I.1 and I.2). Final-answer-only candidates remove BCQ's gain, landing below GRPO† on LLM and VLM (Appendix I.4). Keeping the reasoning and masking the boxed answer still leaves a gain over GRPO†, +1.1/+0.8 pp on LLM/VLM, so BCQ needs the candidates' reasoning, not just their answers. Rescoring the 23 shared-template settings with seven other judges leaves the ordering intact (Appendix I.5). ZPPO also stays ahead of our zero-variance, self-distillation, hybrid, and multimodal-reward adaptations (Appendices I.3, I.6, and I.7; hybrid RL+distillation at most +1.1 pp above GRPO†), keeps its gain under three other policy objectives (Appendix I.8), and on Ovis2.5-2B (Lu et al., 2025) and InternVL3.5-2B (Wang et al., 2025c) stays above GRPO† (Appendix I.9).

5 Conclusion

ZPPO keeps the teacher only inside the prompt, so the student still benefits from teacher knowledge. BCQ and NCQ recover a learning signal where every rollout fails and the group-relative advantage vanishes, and the replay buffer turns each hard question into repeated practice. Across 31 benchmarks and four student scales it leads distillation, GRPO, and every isolated component, and improves the LLM and Video families that distillation degrades. The flip side is a structural ceiling: questions on which both teacher and student fail admit no BCQ candidate and leave only NCQ, which is limited alone—the *zone* collapses to the NCQ-only regime. Extending the student's *zone* beyond what the current teacher covers is, in our view, an important open problem this work leaves behind.

AI USE STATEMENT

We used generative AI to edit prose, check grammar, cross-check the text against the tables and cited papers, and draw the concept illustration of Fig. 2; we did not use it for method design, experiment design, code, or analysis. The generative models inside the training and evaluation pipeline—the frozen teacher, the reward grader, and the judges—are components of the measured system (Appendix E, Appendix C.2). The authors are responsible for the paper's text, claims, and artifacts.

ETHICS STATEMENT

ZPPO post-trains publicly released open-weight checkpoints on two public datasets, so it can carry their biases. The checkpoints are Qwen3.5 (Qwen Team, 2026), plus Ovis2.5-2B (Lu et al., 2025) and InternVL3.5-2B (Wang et al., 2025c) in Appendix I.9; the datasets are in Appendix C.1. The teacher reaches the student only through the prompt, so its bias enters what the student reads. The reward grades answer correctness, not safety or fairness, so deployment needs a separate safety evaluation.

REPRODUCIBILITY STATEMENT

Every reported score and diagnostic in this paper is measured in our own runs under one evaluation setup, except Correction-SFT†, which keeps its training-time `<correct>` tag at evaluation (Appendix E.2); none is copied from another paper. Appendix A gives the measured population behind each number, and Appendix G.7 the paired bootstrap intervals over the benchmark set. Appendix G also reports the buffer dynamics, the component and recipe results, the Hint and Prefix results, and the candidate-match audit. Appendix F.2 restates Fig. 1 and the boxed takeaway of Sec. 1. Algorithm 1 and Appendix D give the ZPPO training step, Appendix E.2 the executed step of every baseline and control, including what the Hint and Prefix rows take from each cited method (Algorithms 5 and 6), and Appendix I what each other compared row changes: the group-size and corpus-pass sweep (Appendix I.1), the replay control (Appendix I.2), the zero-variance methods (Appendix I.3), the BCQ candidate-content controls (Appendix I.4), the judge sweep (Appendix I.5), the teacher-side alternatives (Appendix I.6), the multimodal rewards (Appendix I.7), the three other policy objectives (Appendix I.8), and the two other student backbones (Appendix I.9). Appendix E lists the training settings, the hardware, the public model identifiers, the grader, the training-side prompt templates, and the measured per-run wall-clock (Tab. 12). Appendix C builds ZPPO-77K from two public datasets, and Appendix C.2 gives the decoding, the instruction in Fig. 8, the scoring path, the judge, and the judge templates of all 31 evaluation settings. Appendix H measures the 27B teacher on the same settings, per benchmark in Appendix H.2. Evaluation never uses a BCQ or NCQ prompt: every reported score is measured on a candidate-free prompt (Appendix I.10), and Appendix I.11 states the conditions the method needs and the limits of the evidence.

REFERENCES

Manoj Acharya, Kushal Kafle, and Christopher Kanan. TallyQA: Answering complex counting questions. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 8076–8084, 2019.

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024.

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. Learning from mistakes makes LLM better reasoner. *arXiv preprint arXiv:2310.20689*, 2023.

Anthropic. The Claude 3 model family: Opus, Sonnet, Haiku. `https://www.anthropic.com`, 2024. URL `https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf`.

Andrei Baroian and Rutger Berger. Prompt replay: Speeding up GRPO with on-policy reuse of high-signal prompts. *arXiv preprint arXiv:2603.21177*, 2026.

Walid Bousselham, Hilde Kuehne, and Cordelia Schmid. VOLD: Reasoning transfer from LLMs to vision-language models via on-policy distillation. *arXiv preprint arXiv:2510.23497*, 2025.

Jie Cao and Jing Xiao. An augmented benchmark dataset for geometric question answering through dual parallel text encoding. In *Proceedings of the 29th international conference on computational linguistics*, pp. 1511–1520, 2022.

Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, et al. MiniMax-M1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*, 2025a.

Hardy Chen, Haoqin Tu, Fali Wang, Hui Liu, Xianfeng Tang, Xinya Du, Yuyin Zhou, and Cihang Xie. SFT or RL? an early investigation into training R1-Like reasoning large vision-language models. *arXiv preprint arXiv:2504.11468*, 2025b.

Justin Chen, Xiangyu Peng, Prafulla Kumar Choubey, Kung-Hsiang Huang, Jiaxin Zhang, Mohit Bansal, and Chien-Sheng Wu. Nudging the boundaries of LLM reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 30315–30335, 2026a.

Liang Chen, Lei Li, Haozhe Zhao, Yifan Song, and Vinci. R1-V: Reinforcing super generalization ability in vision-language models with less than $3. `https://github.com/Deep-Agent/R1-V`, 2025c. Accessed: 2025-02-02.

Liang Chen, Weichu Xie, Yiyan Liang, Hongfeng He, Hans Zhao, Zhibo Yang, Zhiqi Huang, Haoning Wu, Haoyu Lu, Y. Charles, Yiping Bao, Yuantao Fan, Guopeng Li, Haiyang Shen, Xuanzhong Chen, et al. BabyVision: Visual reasoning beyond language. *arXiv preprint arXiv:2601.06521*, 2026b.

Yi Chen, Yuying Ge, Rui Wang, Yixiao Ge, Junhao Cheng, Ying Shan, and Xihui Liu. GRPO-CARE: Consistency-aware reinforcement learning for multimodal reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 4305–4320, 2026c.

Yew Ken Chia, Vernon Toh, Deepanway Ghosal, Lidong Bing, and Soujanya Poria. PuzzleVQA: Diagnosing multimodal reasoning challenges of language models with abstract visual patterns. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 16259–16273, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.962. URL `https://aclanthology.org/2024.findings-acl.962/`.

Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. SFT memorizes, RL generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and PixMo: Open weights and open data for state-of-the-art vision-language models. *arXiv preprint arXiv:2409.17146*, 2024.

Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. OpenVLThinker: Complex vision-language reasoning via iterative SFT-RL cycles. *arXiv preprint arXiv:2503.17352*, 2025.

Mengfei Du, Binhao Wu, Zejun Li, Xuanjing Huang, and Zhongyu Wei. EmbSpatial-Bench: Benchmarking spatial understanding for embodied tasks with large vision-language models. *arXiv preprint arXiv:2406.05756*, 2024.

Yichen Feng, Zhangchen Xu, Fengqing Jiang, Yuetai Li, Bhaskar Ramasubramanian, Luyao Niu, Bill Yuchen Lin, and Radha Poovendran. VisualSphinx: Large-scale synthetic vision logic puzzles for RL. *arXiv preprint arXiv:2505.23977*, 2025a.

Yunzhen Feng, Parag Jain, Anthony Hartshorn, Yaqi Duan, and Julia Kempe. Don't waste mistakes: Leveraging negative RL-groups via confidence reweighting. *arXiv preprint arXiv:2510.08696*, 2025b.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-MME: The first-ever comprehensive evaluation benchmark of multi-modal LLMs in video analysis. *arXiv preprint arXiv:2405.21075*, 2024.

Ling Fu, Zhebin Kuang, Jiajun Song, Mingxin Huang, Biao Yang, Yuzhe Li, Linghao Zhu, Qidi Luo, Xinyu Wang, Hao Lu, et al. OCRBench v2: An improved benchmark for evaluating large multimodal models on visual text localization and reasoning. *arXiv preprint arXiv:2501.00321*, 2025.

Yuqian Fu, Haohuan Huang, Kaiwen Jiang, Jiacai Liu, Zhuo Jiang, Yuanheng Zhu, and Dongbin Zhao. Revisiting on-policy distillation: Empirical failure modes and simple fixes. *arXiv preprint arXiv:2603.25562*, 2026.

Jiahui Gao, Renjie Pi, Jipeng Zhang, Jiacheng Ye, Wanjun Zhong, Yufei Wang, Lanqing Hong, Jianhua Han, Hang Xu, Zhenguo Li, et al. G-LLaVA: Solving geometric problem with multi-modal large language model. *arXiv preprint arXiv:2312.11370*, 2023.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, et al. Are we done with MMLU? *arXiv preprint arXiv:2406.04127*, 2024.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International journal of computer vision*, 129(6):1789–1819, 2021.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Jialiang Guo, Fucheng Xiong, Xu He, Haodong Zhao, Xingyang Li, Ke Zeng, and Xunliang Cai. Turning failures into value: Negative experience replay for RLVR via confidence gating and boundary failure sampling. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 36316–36334, 2026.

Zhuowen Han, Jinwei Xiao, Zhengxi Lu, Renren Jin, Zhiyuan Yao, Yuxin Liu, Hongyan Hao, Yueqing Sun, Yu Yang, Qi Gu, et al. Distill where you fail: Recovering learning signals of negative RL-groups from adaptive teacher guidance. *arXiv preprint arXiv:2608.00782*, 2026.

Bingxiang He, Zekai Qu, Zeyuan Liu, Yinghao Chen, Yuxin Zuo, Cheng Qian, Kaiyan Zhang, Weize Chen, Chaojun Xiao, Ganqu Cui, et al. JustRL: Scaling a 1.5B LLM with a simple RL recipe. *arXiv preprint arXiv:2512.16649*, 2025.

Xuehai He, Yichen Zhang, Luntian Mou, Eric Xing, and Pengtao Xie. PathVQA: 30000+ questions for medical visual question answering. *arXiv preprint arXiv:2003.10286*, 2020.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. REINFORCE++: Stabilizing critic-free policy optimization with global advantage normalization. *arXiv preprint arXiv:2501.03262*, 2025a.

Jian Hu, Mingjie Liu, Ximing Lu, Fang Wu, Zaid Harchaoui, Shizhe Diao, Yejin Choi, Pavlo Molchanov, Jun Yang, Jan Kautz, et al. BroRL: Scaling reinforcement learning via broadened exploration. *arXiv preprint arXiv:2510.01180*, 2025b.

Kairui Hu, Penghao Wu, Fanyi Pu, Wang Xiao, Yuanhan Zhang, Xiang Yue, Bo Li, and Ziwei Liu. Video-MMMU: Evaluating knowledge acquisition from multi-discipline professional videos. *arXiv preprint arXiv:2501.13826*, 2025c.

Maggie Huan, Yuetai Li, Tuney Zheng, Xiaoyu Xu, Seungone Kim, Minxin Du, Radha Poovendran, Graham Neubig, and Xiang Yue. Does math reasoning improve general LLM capabilities? understanding transferability of LLM reasoning. *arXiv preprint arXiv:2507.00432*, 2025.

Muye Huang, Han Lai, Xinyu Zhang, Wenjun Wu, Jie Ma, Lingling Zhang, and Jun Liu. EvoChart: A benchmark and a self-training approach towards real-world chart understanding. *arXiv preprint arXiv:2409.01577*, 2024.

Qihan Huang, Weilong Dai, Jinlong Liu, Wanggui He, Hao Jiang, Mingli Song, Jingyuan Chen, Chang Yao, and Jie Song. Boosting MLLM reasoning with text-debiased Hint-GRPO. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4848–4857. IEEE, 2025a.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, et al. C-Eval: A multi-level multi-discipline Chinese evaluation suite for foundation models. *arXiv preprint arXiv:2305.08322*, 2023.

Zeyu Huang, Tianhao Cheng, Zihan Qiu, Zili Wang, Yinghui Xu, Edoardo M Ponti, and Ivan Titov. Blending supervised and reinforcement fine-tuning with prefix sampling. *arXiv preprint arXiv:2507.01679*, 2025b.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, et al. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

Drew A Hudson and Christopher D Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6700–6709, 2019.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Yiming Jia, Jiachen Li, Xiang Yue, Bo Li, Ping Nie, Kai Zou, and Wenhu Chen. VisualWebInstruct: Scaling up multimodal instruction data through web search. *arXiv preprint arXiv:2503.10582*, 2025.

Bingqing Jiang and Difan Zou. Structured role-aware policy optimization for multimodal reasoning. *arXiv preprint arXiv:2605.07274*, 2026.

Batuhan K Karaman, Aditya Rawal, Suhaila Shakiah, Mohammad Ghavamzadeh, Mingyi Hong, Arijit Biswas, and Ruida Zhou. DISPO: Enhancing training efficiency and stability in reinforcement learning for large language model mathematical reasoning. *arXiv preprint arXiv:2602.00983*, 2026.

Mehran Kazemi, Hamidreza Alvari, Ankit Anand, Jialin Wu, Xi Chen, and Radu Soricut. GeomVerse: A systematic evaluation of large models for geometric reasoning. *arXiv preprint arXiv:2312.12241*, 2023.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pp. 235–251. Springer, 2016.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. Are you smarter than a sixth grader? Textbook question answering for multimodal machine comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern recognition*, pp. 4999–5007, 2017.

Devvrit Khatri, Lovish Madaan, Rishabh Tiwari, Rachit Bansal, Sai Surya Duvvuri, Manzil Zaheer, Inderjit S Dhillon, David Brandfonbrener, and Rishabh Agarwal. The art of scaling reinforcement learning compute for LLMs. *arXiv preprint arXiv:2510.13786*, 2025.

Jeonghye Kim, Xufang Luo, Minbeom Kim, Sangmook Lee, Dohyung Kim, Jiwon Jeon, Dongsheng Li, and Yuqing Yang. Why does self-distillation (sometimes) degrade the reasoning capability of LLMs? *arXiv preprint arXiv:2603.24472*, 2026.

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

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. DistiLLM: Towards streamlined distillation for large language models. *arXiv preprint arXiv:2402.03898*, 2024.

Jason J. Lau, Soumya Gayen, Asma Ben Abacha, and Dina Demner-Fushman. A dataset of clinically generated visual questions and answers about radiology images. *Scientific Data*, 5(1):180251, 2018.

Thanh-Long V Le, Myeongho Jeon, Kim Vu, Viet Lai, and Eunho Yang. No prompt left behind: Exploiting zero-variance prompts in LLM reinforcement learning via entropy-guided advantage shaping. In *International Conference on Learning Representations*, volume 2026, pp. 121956–121982, 2026.

Byung-Kwan Lee, Ryo Hachiuma, Yong Man Ro, Frank Wang, and Yueh-Hua Wu. Unified reinforcement and imitation learning for vision-language models. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 156508–156534. Curran Associates, Inc., 2025a. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/e58497367bc8730f61a87d37800c0a06-Paper-Conference.pdf.

Byung-Kwan Lee, Ryo Hachiuma, Yu-Chiang Frank Wang, Yong Man Ro, and Yueh-Hua Wu. VLsI: Verbalized layers-to-interactions from large to small vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29545–29557, June 2025b.

Byung-Kwan Lee, Yu-Chiang Frank Wang, and Ryo Hachiuma. Masking teacher and reinforcing student for distilling vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10126–10141, June 2026.

Sicong Leng, Jing Wang, Jiaxi Li, Hao Zhang, Zhiqiang Hu, Boqiang Zhang, Yuming Jiang, Hang Zhang, Xin Li, et al. MMR1: Enhancing multimodal reasoning with variance-aware sampling and open resources. *arXiv preprint arXiv:2509.21268*, 2025.

Ang Li, Charles Wang, Deqing Fu, Kaiyu Yue, Zikui Cai, Wang Bill Zhu, Ollie Liu, Peng Guo, Willie Neiswanger, Furong Huang, Tom Goldstein, and Micah Goldblum. Zebra-CoT: A dataset for interleaved vision language reasoning. *arXiv preprint arXiv:2507.16746*, 2025a.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, Limin Wang, and Yu Qiao. MVBench: A comprehensive multi-modal video understanding benchmark. *arXiv preprint arXiv:2311.17005*, 2024a.

Lei Li, Yuqi Wang, Runxin Xu, Peiyi Wang, Xiachong Feng, Lingpeng Kong, and Qi Liu. Multimodal ArXiv: A dataset for improving scientific comprehension of large vision-language models. *arXiv preprint arXiv:2403.00231*, 2024b.

Long Li, Zhijian Zhou, Tianyi Wang, Weidi Xu, Zuming Huang, Wei Chu, Zhe Wang, Shirui Pan, Chao Qu, and Yuan Qi. DyJR: Preserving diversity in reinforcement learning with verifiable rewards via dynamic Jensen-Shannon replay. *arXiv preprint arXiv:2603.16157*, 2026a.

Siheng Li, Zhanhui Zhou, Wai Lam, Chao Yang, and Chaochao Lu. RePO: Replay-enhanced policy optimization. *arXiv preprint arXiv:2506.09340*, 2025b.

Xuanlin Li, Yunhao Fang, Minghua Liu, Zhan Ling, Zhuowen Tu, and Hao Su. Distilling large vision-language model with out-of-distribution generalizability. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2492–2503, 2023.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huan-ang Gao, Wenkai Yang, Zhiyuan Liu, et al. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026b.

Yu Li, Tian Lan, and Zhengling Qi. When right meets wrong: Bilateral context conditioning with reward-confidence correction for grpo. *arXiv preprint arXiv:2603.13134*, 2026c.

Yuetai Li, Xiang Yue, Zhangchen Xu, Fengqing Jiang, Luyao Niu, Bill Yuchen Lin, Bhaskar Ramasubramanian, and Radha Poovendran. Small models struggle to learn from strong reasoners. *arXiv preprint arXiv:2502.12143*, 2025c.

Ziheng Li, Zexu Sun, Jinman Zhao, Erxue Min, Yongcheng Zeng, Hui Wu, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Xu Chen, et al. Staying in the sweet spot: Responsive reasoning evolution via capability-adaptive hint scaffolding. *arXiv preprint arXiv:2509.06923*, 2025d.

Chen Lin, Kedi Chen, and Wei Zhang. ReNIO: Reweighting negative trajectory importance for LLM on-policy distillation. *arXiv preprint arXiv:2606.23104*, 2026a.

Honglin Lin, Zheng Liu, Yun Zhu, Chonghan Qin, Juekai Lin, Xiaoran Shang, Conghui He, Wentao Zhang, and Lijun Wu. MMFineReason: Closing the multimodal reasoning gap via open data-centric methods. *arXiv preprint arXiv:2601.21821*, 2026b.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. ProRL: Prolonged reinforcement learning expands reasoning boundaries in large language models. *arXiv preprint arXiv:2505.24864*, 2025a.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, Yejin Choi, Jan Kautz, and Pavlo Molchanov. DLER: Doing length pEnalty right - incentivizing more intelligence per token via reinforcement learning. *arXiv preprint arXiv:2510.15110*, 2025b.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, et al. GDPO: Group reward-decoupled normalization policy optimization for multi-reward RL optimization. *arXiv preprint arXiv:2601.05242*, 2026a.

Wenpu Liu, Yuqi Xu, Weichu Xie, Yongfu Zhu, Shuai Dong, Ziyue Wang, Wenqi Shao, Xiaoying Zhang, Tong Yang, Nan Duan, et al. Leveraging error diversity in group rollouts for reinforcement learning. *arXiv preprint arXiv:2605.17333*, 2026b.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-Like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025c.

Zirui Liu, Jie Ouyang, Qi Liu, Xianquan Wang, Jiayu Liu, Tingyue Pan, Qingchuan Li, Jing Sha, Zhenya Huang, Shijin Wang, et al. SocraticPO: Policy optimization via interactive guidance. *arXiv preprint arXiv:2606.09887*, 2026c.

Ziyu Liu, Zeyi Sun, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Haodong Duan, Dahua Lin, and Jiaqi Wang. Visual-RFT: Visual reinforcement fine-tuning. *arXiv preprint arXiv:2503.01785*, 2025d.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20251026. https://thinkingmachines.ai/blog/on-policy-distillation.

Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-GPS: Interpretable geometry problem solving with formal language and symbolic reasoning. In *The 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2021a.

Pan Lu, Liang Qiu, Jiaqi Chen, Tony Xia, Yizhou Zhao, Wei Zhang, Zhou Yu, Xiaodan Liang, and Song-Chun Zhu. IconQA: A new benchmark for abstract diagram understanding and visual language reasoning. *arXiv preprint arXiv:2110.13214*, 2021b.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. MathVista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*, 2023.

Shiyin Lu, Yang Li, Yu Xia, Yuwei Hu, Shanshan Zhao, Yanqing Ma, Zhichao Wei, Yinglun Li, Lunhao Duan, Jianshan Zhao, et al. Ovis2.5 technical report. *arXiv preprint arXiv:2508.11737*, 2025.

Thang Luong, Dawsen Hwang, Hoang H. Nguyen, Golnaz Ghiasi, Yuri Chervonyi, Insuk Seo, Junsu Kim, Garrett Bingham, Jonathan Lee, Swaroop Mishra, et al. Towards robust mathematical reasoning. *arXiv preprint arXiv:2511.01846*, 2025.

Lu Ma, Hao Liang, Meiyi Qiang, Lexiang Tang, Xiaochen Ma, Zhen Wong, Junbo Niu, Chengyu Shen, Runming He, Yanhao Li, et al. Learning what reinforcement learning can't: Interleaved online fine-tuning for hardest questions. In *International Conference on Learning Representations*, volume 2026, pp. 80802–80821, 2026.

Ilia Mahrooghi, Aryo Lotfi, and Emmanuel Abbe. Goldilocks RL: Tuning task difficulty to escape sparse rewards for reasoning. *arXiv preprint arXiv:2602.14868*, 2026.

Andrés Marafioti, Orr Zohar, Miquel Farré, Merve Noyan, Elie Bakouch, Pedro Cuenca, Cyril Zakka, Loubna Ben Allal, Anton Lozhkov, Nouamane Tazi, et al. SmolVLM: Redefining small and efficient multimodal models. *arXiv preprint arXiv:2504.05299*, 2025.

Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. *arXiv preprint arXiv:2203.10244*, 2022.

Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. InfographicVQA. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1697–1706, 2022.

Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu, Tiancheng Han, Botian Shi, Wenhai Wang, Junjun He, Kaipeng Zhang, Ping Luo, Yu Qiao, Qiaosheng Zhang, and Wenqi Shao. MM-Eureka: Exploring the frontiers of multimodal reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2503.07365*, 2025.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 5191–5198, 2020.

Vaskar Nath, Elaine Lau, Anisha Gunjal, Manasi Sharma, Nikhil Baharte, and Sean Hendryx. Adaptive guidance accelerates reinforcement learning of reasoning models. *arXiv preprint arXiv:2506.13923*, 2025.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, et al. Humanity's Last Exam. *arXiv preprint arXiv:2501.14249*, 2025.

Runqi Qiao, Qiuna Tan, Guanting Dong, Minhui Wu, Chong Sun, Xiaoshuai Song, Zhuoma GongQue, Shanglin Lei, Zhe Wei, Miaoxuan Zhang, et al. We-Math: Does your large multimodal model achieve human-like mathematical reasoning? *arXiv preprint arXiv:2407.01284*, 2024.

Guanqiao Qu, Qiyuan Chen, Wei Wei, Zheng Lin, Xianhao Chen, and Kaibin Huang. Mobile edge intelligence for large language models: A contemporary survey. *IEEE Communications Surveys & Tutorials*, 2025.

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

Pooyan Rahmanzadehgervi, Logan Bolton, Mohammad Reza Taesiri, and Anh Totti Nguyen. Vision language models are blind: Failing to translate detailed visual features into words. *arXiv preprint arXiv:2407.06581*, 2024.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Qihan Ren, Peng Wang, Ruikun Cai, Shuai Shao, Dadi Guo, Yuejin Xie, Yafu Li, Quanshi Zhang, Xia Hu, Jing Shao, et al. Rethinking generalization in reasoning SFT: A conditional analysis on optimization, data, and model capability. *arXiv preprint arXiv:2604.06628*, 2026.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Gabriel Sarch, Linrong Cai, Qunzhong Wang, Haoyang Wu, Danqi Chen, and Zhuang Liu. Vero: Open reinforcement learning recipes for visual reasoning. In *European Conference on Computer Vision*, pp. 592–612. Springer, 2026.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. A-OKVQA: A benchmark for visual question answering using world knowledge. *arXiv preprint arXiv:2206.01718*, 2022.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, et al. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv:2312.06585*, 2023.

Ved Sirdeshmukh, Kaustubh Deshpande, Johannes Mols, Lifeng Jin, Ed-Yeremai Cardona, Dean Lee, Jeremy Kritz, Willow Primack, Summer Yue, and Chen Xing. MultiChallenge: A realistic multi-turn conversation evaluation benchmark challenging to frontier LLMs. *arXiv preprint arXiv:2501.17399*, 2025.

Mingyang Song and Mao Zheng. A survey of on-policy distillation for large language models. *arXiv preprint arXiv:2604.00626*, 2026.

Yueqi Song, Tianyue Ou, Yibo Kong, Zecheng Li, Graham Neubig, and Xiang Yue. VisualPuzzles: Decoupling multimodal reasoning evaluation from domain knowledge. *arXiv preprint arXiv:2504.10342*, 2025.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Pavan Kumar Anasosalu Vasu, Fartash Faghri, Chun-Liang Li, Cem Koc, Nate True, Albert Antony, Gokula Santhanam, James Gabriel, Peter Grasch, Oncel Tuzel, et al. FastVLM: Efficient vision encoding for vision language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 19769–19780, 2025.

Lev Semenovich Vygotsky and Michael Cole. *Mind in society: Development of higher psychological processes*. Harvard university press, 1978.

Haozhe Wang, Chao Qu, Zuming Huang, Wei Chu, Fangzhen Lin, and Wenhu Chen. VL-Rethinker: Incentivizing self-reflection of vision-language models with reinforcement learning. *arXiv preprint arXiv:2504.08837*, 2025a.

Haozhe Wang, Alex Su, Weiming Ren, Fangzhen Lin, and Wenhu Chen. Pixel Reasoner: Incentivizing pixel-space reasoning with curiosity-driven reinforcement learning. *arXiv preprint arXiv:2505.15966*, 2025b.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with MATH-Vision dataset. *arXiv preprint arXiv:2402.14804*, 2024a.

Renxi Wang, Haonan Li, Xudong Han, Yixuan Zhang, and Timothy Baldwin. Learning from failure: Integrating negative examples when fine-tuning large language models as agents. *arXiv preprint arXiv:2402.11651*, 2024b.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. InternVL3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025c.

Xinyi Wang, Jinyi Han, Zishang Jiang, Tingyun Li, Jiaqing Liang, Sihang Jiang, Zhaoqian Dai, Ma Shuguang, Fei Yu, and Yanghua Xiao. Don't tell the answer, truly guide the reasoning during RL rollouts. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 3437–3455, 2026.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024c.

Zirui Wang, Mengzhou Xia, Luxi He, Howard Chen, Yitao Liu, Richard Zhu, Kaiqu Liang, Xindi Wu, Haotian Liu, Sadhika Malladi, Alexis Chevalier, Sanjeev Arora, and Danqi Chen. CharXiv: Charting gaps in realistic chart understanding in multimodal LLMs. *arXiv preprint arXiv:2406.18521*, 2024d.

Jinyang Wu, Chonghua Liao, Mingkuan Feng, Shuai Zhang, Zhengqi Wen, Pengpeng Shao, Huazhe Xu, and Jianhua Tao. Thought-augmented policy optimization: Bridging external guidance and internal capabilities. *arXiv preprint arXiv:2505.15692v1*, 2025.

Taiqiang Wu, Chaofan Tao, Jiahao Wang, Runming Yang, Zhe Zhao, and Ngai Wong. Rethinking Kullback-Leibler divergence in knowledge distillation for large language models. *arXiv preprint arXiv:2404.02657*, 2024.

Yecheng Wu, Song Han, and Han Cai. Lightning OPD: Efficient post-training for large reasoning models with offline on-policy distillation. *arXiv preprint arXiv:2604.13010*, 2026a.

Zhuoyang Wu, Xinze Li, Zhenghao Liu, Yukun Yan, Zhiyuan Liu, Minghe Yu, Cheng Yang, Yu Gu, Ge Yu, and Maosong Sun. Enhancing long-chain reasoning distillation through error-aware self-reflection. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 1465–1482, 2026b.

Zhiheng Xi, Guanyu Li, Yutao Fan, Honglin Guo, Yufang Liu, Xiaoran Fan, Jiaqi Liu, Jingchao Ding, Wangmeng Zuo, Zhenfei Yin, Lei Bai, Tao Ji, Tao Gui, Qi Zhang, Philip Torr, and Xuanjing Huang. BMMR: A large-scale bilingual multimodal multi-discipline reasoning dataset. *arXiv preprint arXiv:2507.03483*, 2025.

Yu Xia, Canwen Xu, Zhewei Yao, Julian McAuley, and Yuxiong He. Learning to hint for reinforcement learning. *arXiv preprint arXiv:2604.00698*, 2026.

Tong Xiao, Xin Xu, Zhenya Huang, Hongyu Gao, Quan Liu, Qi Liu, and Enhong Chen. Perception-R1: Advancing multimodal reasoning capabilities of MLLMs via visual perception reward. In *International Conference on Learning Representations*, volume 2026, pp. 26868–26898, 2026.

Guowei Xu, Peng Jin, Ziang Wu, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. LLaVA-CoT: Let vision language models reason step-by-step. *arXiv preprint arXiv:2411.10440*, 2024.

Hongling Xu, Qi Zhu, Heyuan Deng, Jinpeng Li, Lu Hou, Yasheng Wang, Lifeng Shang, Ruifeng Xu, and Fei Mi. KDRL: Post-training reasoning LLMs via unified knowledge distillation and reinforcement learning. *arXiv preprint arXiv:2506.02208*, 2025.

Shicheng Xu, Liang Pang, Yunchang Zhu, Jia Gu, Zihao Wei, Jingcheng Deng, Feiyang Pan, Huawei Shen, and Xueqi Cheng. RLKD: Distilling LLMs' reasoning via reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to reason under off-policy guidance. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 117157–117186. Curran Associates, Inc., 2025. doi: 10.52202/085713-3904. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/a9d5c33e38442450f9781bd7bad3c81e-Paper-Conference.pdf.

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled RLVR. *arXiv preprint arXiv:2604.03128*, 2026a.

Wenkai Yang, Weijie Liu, Ruobing Xie, Kai Yang, Saiyong Yang, and Yankai Lin. Learning beyond teacher: Generalized on-policy distillation with reward extrapolation. *arXiv preprint arXiv:2602.12125*, 2026b.

Yi Yang, Xiaoxuan He, Hongkun Pan, Xiyan Jiang, Yan Deng, Xingtao Yang, Haoyu Lu, Dacheng Yin, Fengyun Rao, Minfeng Zhu, et al. R1-Onevision: Advancing generalized multimodal reasoning through cross-modal formalization. *arXiv preprint arXiv:2503.10615*, 2025a.

Yue Yang, Ajay Patel, Matt Deitke, Tanmay Gupta, Luca Weihs, Andrew Head, Mark Yatskar, Chris Callison-Burch, Ranjay Krishna, Aniruddha Kembhavi, and Christopher Clark. Scaling text-rich image understanding via code-guided synthetic multimodal data generation. *arXiv preprint arXiv:2502.14846*, 2025b.

Deheng Ye, Zhao Liu, Mingfei Sun, Bei Shi, Peilin Zhao, Hao Wu, Hongsheng Yu, Shaojie Yang, Xipeng Wu, Qingwei Guo, et al. Mastering complex control in MOBA games with deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 6672–6679, 2020.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. MM-Vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*, 2023.

Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang, Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu, Ge Zhang, Huan Sun, Yu Su, Wenhu Chen, and Graham Neubig. MMMU-Pro: A more robust multi-discipline multimodal understanding benchmark. *arXiv preprint arXiv:2409.02813*, 2024.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. *arXiv preprint arXiv:2203.14465*, 2022.

Runzhe Zhan, Yafu Li, Zhi Wang, Xiaoye Qu, Dongrui Liu, Jing Shao, Derek F. Wong, and Yu Cheng. ExGRPO: Learning to reason from experience. *arXiv preprint arXiv:2510.02245*, 2025.

Chen Zhang, Qiuchi Li, Dawei Song, Zheyu Ye, Yan Gao, and Yan Hu. Towards the law of capacity gap in distilling language models. *arXiv preprint arXiv:2311.07052*, 2023a.

Chen Zhang, Yang Yang, Jiahao Liu, Jingang Wang, Yunsen Xian, Benyou Wang, and Dawei Song. Lifting the curse of capacity gap in distilling language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4535–4553, Toronto, Canada, July 2023b. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.249. URL https://aclanthology.org/2023.acl-long.249/.

Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. RAVEN: A dataset for relational and analogical visual rEasoNing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Dongxu Zhang, Zhichao Yang, Sepehr Janghorbani, Jun Han, Andrew Ressler II, Qian Qian, Gregory D. Lyng, Sanjit Singh Batra, and Robert E. Tillman. Fast and effective on-policy distillation from reasoning prefixes. *arXiv preprint arXiv:2602.15260*, 2026a.

Hongzhi Zhang, Jia Fu, Jingyuan Zhang, Kai Fu, Qi Wang, Fuzheng Zhang, and Guorui Zhou. RLEP: Reinforcement learning with experience replay for LLM reasoning. *arXiv preprint arXiv:2507.07451*, 2025a.

Jiarui Zhang, Ollie Liu, Tianyu Yu, Jinyi Hu, and Willie Neiswanger. Euclid: Supercharging multimodal LLMs with synthetic high-fidelity visual descriptions. *arXiv preprint arXiv:2412.08737*, 2024a.

Kaiyi Zhang, Ang Lv, Jinpeng Li, Yongbo Wang, Feng Wang, Haoyuan Hu, and Rui Yan. StepHint: Multi-level stepwise hints enhance reinforcement learning to reason. *arXiv preprint arXiv:2507.02841*, 2025b.

Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Peng Gao, et al. MathVerse: Does your multi-modal LLM truly see the diagrams in visual math problems? *arXiv preprint arXiv:2403.14624*, 2024b.

Shijie Zhang, Guohao Sun, Kevin Zhang, Xiang Guo, and Rujun Guo. CLPO: Curriculum learning meets policy optimization for LLM reasoning. *arXiv preprint arXiv:2509.25004*, 2025c.

Xuechen Zhang, Zijian Huang, Yingcong Li, Chenshun Ni, Jiasi Chen, and Samet Oymak. BREAD: Branched rollouts from expert anchors bridge SFT & RL for reasoning. *arXiv preprint arXiv:2506.17211*, 2025d.

Zhaoyang Zhang, Shuli Jiang, Yantao Shen, Yuting Zhang, Dhananjay Ram, Shuo Yang, Zhuowen Tu, Wei Xia, and Stefano Soatto. Reinforcement-aware knowledge distillation for LLM reasoning. *arXiv preprint arXiv:2602.22495*, 2026b.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Jpxfof0EaS.

Yilun Zhao, Lujing Xie, Haowei Zhang, Guo Gan, Yitao Long, Zhiyuan Hu, Tongyan Hu, Weiyuan Chen, Chuhan Li, Junyang Song, et al. MMVU: Measuring expert-level multi-discipline video understanding. *arXiv preprint arXiv:2501.12380*, 2025.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization, 2025. URL https://arxiv.org/abs/2507.18071.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.

Hengguang Zhou, Xirui Li, Ruochen Wang, Minhao Cheng, Tianyi Zhou, and Cho-Jui Hsieh. R1-Zero's "aha moment" in visual reasoning on a 2B non-SFT model. *arXiv preprint arXiv:2503.05132*, 2025.

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7W: Grounded question answering in images. *arXiv preprint arXiv:1511.03416*, 2016.

Chengke Zou, Xingang Guo, Rui Yang, Junyu Zhang, Bin Hu, and Huan Zhang. DynaMath: A dynamic visual benchmark for evaluating mathematical reasoning robustness of vision language models. *arXiv preprint arXiv:2411.00836*, 2024.

A MEASUREMENT INDEX

We measure every reported score and diagnostic in our own runs under one evaluation setup. The one exception is Correction-SFT†, which keeps its training-time `<correct>` tag at evaluation (Appendix E.2). For each benchmark, we drop its own instructions on reasoning and answer format and append the instruction in Fig. 8. All checkpoints then share one scoring path and one judge (Appendix C.2). Evaluation never uses a BCQ or NCQ prompt, so every reported benchmark accuracy is measured on a candidate-free prompt (Appendix I.10). Tab. 6 states the population behind each reported number and how it is averaged.

Table 6: The population behind every reported number.

Quantity	Location	Measured population and aggregation
Trained-checkpoint accuracy	Tab. 1–Tab. 5, Tab. 13–Tab. 15, Tab. 18, Tab. 10, Fig. 6 with Tab. 19, the ZPPO rows of Appendix H, and the accuracy tables of Appendix I	Evaluated under the shared decoding and scoring path. The judge sweep of Tab. 33 scores the same responses again with other judges.
Frozen-checkpoint accuracy	Every untrained *Base* row and the Qwen3.5-27B-FP8 teacher rows of Appendix H	The frozen checkpoint, evaluated under the shared decoding and scoring path.
Buffer events and training curves	Fig. 4, Fig. 5, and the buffer and per-step figures of Appendix G	One completed run per (method, scale), logged by rollout step. Each method's buffer counts cover the questions that method admitted (Appendix G.2).
BCQ and NCQ audit rates	Tab. 22, Tab. 23, and the teacher-success rates of Appendix G.8	One logged ZPPO run per scale, read on its last three logged sets of instances. Every instance is built on a hard question. The BCQ audit covers 191 instances over the four scales (1,528 student rollouts). The NCQ audit covers 961 instances over the four scales (7,688 student rollouts).
In-prompt accuracy at the start and end	The *BCQ acc* column of Tab. 32	The first ten and the last ten recorded training steps of one logged 2B run per variant, printed as a start → end pair. The values are rollout accuracy on the reformulated prompt, not on a candidate-free prompt.
Wall-clock time	Tab. 12	One completed run on one 64-GPU allocation, rounded to the nearest hour.
Benchmark-selection intervals	Tab. 20, Tab. 21, Tab. 38	Paired resamples of the printed per-setting cells inside one block, with repeats allowed. The intervals test the choice of benchmarks. Tab. 20 and Tab. 21 draw 10,000 resamples per (scale, family) block. Tab. 38 instead uses the exact resample distribution of each (backbone, family) block. Each cell reports the full-sample Δ, a 95% percentile interval, and the win count.
Plotted gains, win counts, and other derived numbers	Tab. 16, Tab. 17, Tab. 28, the derived numbers of Appendices F.1, G.7, F.3, F.4, and I.11	Arithmetic on per-setting cells or family macro averages, using the printed one-decimal values wherever a table prints them.

Marks used in every table. **(i) †:** A † on a row or a series denotes augmentation with the prompt replay buffer, as in Tab. 1. ZPPO always uses the buffer, so its rows carry no mark. **(ii) *Avg*:** An *Avg* cell is the macro average over that family's settings in its table, with every setting weighted equally. Two gains combine more than one family. The pooled LLM+Video gain of the boxed takeaway in Sec. 1 weights each of its 15 settings equally (Eq. (10)). The mean family gain Δ_{all} of Fig. 7 weights the three families equally (Eq. (11)). **(iii) Δ:** A per-setting Δ is ZPPO minus the strongest non-ZPPO value in that column at the same scale. An *Avg*-column Δ is ZPPO minus the strongest non-ZPPO family average. Every Δ and every derived number is computed from the printed one-decimal cells. A one-problem AIME gap therefore prints as 3.3 or 3.4, depending on rounding

(*e.g.*, $56.7 - 53.3 = 3.4$). The strongest non-ZPPO value can come from the untrained base row, as for AI2D at 4B and EmbSp at 9B in Tab. 13. When a caption names another comparison row, Δ is measured against that row instead. **(iv) Win counts:** A *Win* entry of the bootstrap tables counts the settings on which ZPPO is above its comparator. Appendix F.1 counts ZPPO against the untrained base and against GRPO† cell by cell over the four scales' 124 (scale, setting) cells. **(v) Bold:** Where a table uses bold, it marks the column maximum inside one scale block, or over the whole column when a table has no blocks. Every row that reaches the maximum shares the bold. A column of all zeros has none. The colored Δ entries are differences set in bold color, not column maxima. A caption that states its own bold rule overrides this one.

How the figures read against the tables. The figures plot only the buffer-augmented series, so every plotted baseline uses the prompt replay buffer, as ZPPO does. **(i) Fig. 1** plots ZPPO beside Off-Distill†, On-Distill†, and GRPO† as family macro-average gains over the untrained base, restated point by point in Tab. 16. **(ii) Fig. 6** plots GRPO† and ZPPO, and Tab. 19 restates the plotted family macro averages cell by cell. **(iii) Fig. 7** plots the ZPPO teacher-size sweep against the GRPO†+NCQ reference, and Tab. 28 prints every point as Δ_{all} (Appendix H.3).

B Full Related Work

This appendix expands Sec. 2 and follows its four lines of work. Every compared row takes named elements from its cited methods and runs them on our shared pipeline. Appendix E.2 gives the executed step of every baseline and control, and states what each control takes from its cited methods and what it leaves out. Appendix I holds the other compared rows. Each row there that follows a cited method adds that method's named term to the GRPO† pipeline. The tables therefore compare where and in what form teacher knowledge enters training.

Knowledge distillation. Knowledge distillation trains a student to match the logits, hidden states, or sampled outputs of a teacher (Hinton et al., 2015; Gou et al., 2021; Sanh et al., 2019). Off-policy distillation trains the student on responses the teacher wrote, and on-policy distillation has the teacher re-score the student's own responses for target logits (Ko et al., 2024; Gu et al., 2024; Agarwal et al., 2024; Lu & Thinking Machines Lab, 2025; Fu et al., 2026; Bousselham et al., 2025; Yang et al., 2026b; Zhang et al., 2026a; Wu et al., 2026a). Either way, distillation turns the teacher's output distribution into the signal the student must match (Song & Zheng, 2026; Li et al., 2026b). That signal grows brittle as the teacher–student gap widens (Cho & Hariharan, 2019; Mirzadeh et al., 2020; Zhang et al., 2023b;a), inducing memorization of teacher answers and forgetting of general capabilities (Chu et al., 2025; Chen et al., 2025b; Huan et al., 2025). Students of 3B parameters or fewer benefit inconsistently from distilling a strong reasoner and tend to imitate its surface patterns rather than its transferable procedures (Li et al., 2025c; Ren et al., 2026). Self-training and self-distillation drop the external teacher (Zelikman et al., 2022; Singh et al., 2023; Shenfeld et al., 2026; Zhao et al., 2026; Hübotter et al., 2026; Yang et al., 2026a; Kim et al., 2026). Self-training needs no teacher distribution but still trains on a fixed text target: the student's own graded-correct outputs. Self-distillation often conditions the model on extra context, such as a reference solution or feedback. The student then matches the per-token distribution of that conditioned model. Self-Distilled RLVR (Yang et al., 2026a) is an exception: its distillation sets only the token-level update magnitude, and the verifiable reward sets the direction.

Several distillation methods are built for small VLMs. Layer verbalization turns intermediate teacher layers into text and matches the student's layer-wise progression to them (Lee et al., 2025b). Unified reinforcement and imitation learning pairs the RL objective with adversarial imitation of teacher responses (Lee et al., 2025a). Teacher masking hides the teacher's non-dominant weights and restores them over training, ahead of an offline RL stage (Lee et al., 2026). Representation distillation transfers the teacher's visual space for open-vocabulary generalization (Li et al., 2023). All of these keep the teacher in the loss. Each needs teacher hidden layers, teacher weights, a shared output vocabulary, or a pre-generated teacher corpus. ZPPO uses only text that the frozen teacher writes online, so it needs no shared output vocabulary. Appendix I.9 trains two students from outside the Qwen3.5 family with the same Qwen3.5-27B-FP8 teacher. Each has its own vision encoder, modality connector, and multimodal recipe.

RL post-training and zero-variance groups. PPO, GRPO, DAPO, and related recipes train on the student's own rollouts with rule-based rewards (Schulman et al., 2017; Shao et al., 2024; Yu et al., 2025; Guo et al., 2025; Hu et al., 2025a; Liu et al., 2025c), including for multimodal reasoning (Chen et al., 2025c; Yang et al., 2025a; Deng et al., 2025; Zhou et al., 2025; Liu et al., 2025d). GRPO, DAPO, and other recipes that use the group mean reward as the baseline share one failure mode: an all-wrong group has zero advantage and gives no learning signal. PPO instead subtracts a learned value, so a wrong response can still receive a nonzero advantage. Methods that target all-wrong groups reshape advantages, interleave supervised updates, or feed the teacher back into the gradient. RL-ZVP (Le et al., 2026) and LENS (Feng et al., 2025b) reshape advantages from the student's own rollouts alone. RL-ZVP shapes the advantage of zero-variance groups with token-level entropy. LENS gives each wrong answer a penalty that grows with the student's confidence in it. ReLIFT (Ma et al., 2026) instead interleaves RL with online fine-tuning on teacher solutions to the hardest questions. RSTG (Han et al., 2026) applies on-policy distillation only to all-wrong groups and adds fine-tuning on teacher-generated correct trajectories. Hybrid RL+distillation objectives feed the teacher back into the gradient. KDRL (Xu et al., 2025) adds a KL term toward the teacher, and RLKD (Xu et al., 2026) adds a reward derived from the teacher's reasoning structure. Mixed-policy RL puts the teacher's traces in the rollout group itself. LUFFY (Yan et al., 2025) adds off-policy teacher traces to the on-policy rollouts of each GRPO group and reshapes their importance ratio with regularized importance sampling. All of them change the loss or the target, never the prompt. Appendix I.3 reports our adaptations of the four zero-variance methods, and Appendix I.6 reports the hybrid RL+distillation rows. Appendix E.2 states what our Prefix control takes from LUFFY.

Replay, curricula, and prompt-side guidance. Methods in this line do act on the prompt. Replay methods feed the output of earlier steps back into the update. They reuse past trajectories as extra off-policy groups (Zhan et al., 2025; Zhang et al., 2025a; Li et al., 2025b), take recent trajectories as a divergence reference (Li et al., 2026a), or replay the prompts that earlier rollouts marked as high-signal (Baroian & Berger, 2026). Curriculum methods adapt the training distribution to what the student can currently do (Zhang et al., 2025c; Mahrooghi et al., 2026). Both re-expose a hard prompt without saying *why* the last attempt failed. Hint methods put a cue in the prompt (Nath et al., 2025; Chen et al., 2026a; Xia et al., 2026; Wu et al., 2025). In HINT (Wang et al., 2026), that cue is an abstract Meta-Hint rather than the answer or a partial solution. Hints still supply only a correct cue, with no wrong alternative, so the student does not weigh a correct line of reasoning against a wrong one. Prefix methods hand over a reference solution's opening steps and let the student continue from them (Zhang et al., 2025d;b; Li et al., 2025d; Huang et al., 2025b;a). The optimized trajectories therefore start from a point the student could not reach alone. Prefix-RFT (Huang et al., 2025b) also keeps prefix tokens in the loss and trains the highest-entropy ones with the advantage of the prefixed trajectory. Our Hint control gives an answer-free cue abstracted from a solution, and our Prefix control gives the opening portion of the solution itself. Appendix E.2 states what the two controls take from each cited method, and Tab. 4 compares BCQ with both.

Learning from mistakes. On the supervised side, LeMa (An et al., 2023) fine-tunes on corrected solutions, and NAT (Wang et al., 2024b) fine-tunes on correct and wrong trajectories labeled by an input tag. ORION (Wu et al., 2026b) fine-tunes on teacher chains of thought refined with the student's own errors. Methods that train on the student's own rollouts act on the loss or the advantage instead. NexGRPO (Guo et al., 2026) replays failed trajectories chosen by confidence gating and boundary failure sampling. EDAS (Liu et al., 2026b) shapes the advantage of wrong rollouts by how diverse the group's errors are. ReNIO (Lin et al., 2026a) reweights the student's wrong responses by their importance inside on-policy distillation. A corrected solution is a single target to imitate, with no wrong alternative to reject. Reweighting leaves the question unchanged. BiCC (Li et al., 2026c) scores each correct rollout with the group's wrong rollouts in its context, and each wrong rollout with the correct ones. It recomputes the ratio of rollouts already sampled and draws no new ones. Tab. 5 compares NCQ with Correction-SFT and Error-reflection-SFT, two controls that route the same error information into supervised targets. ZPPO keeps the teacher only inside the prompt, and every gradient-counted token is sampled by the student. Interactive guidance also keeps the teacher in the prompt. In SocraticPO (Liu et al., 2026c), the teacher reads a wrong attempt and writes corrective guidance in natural language. The student then writes a new full answer, with the failed attempt and the guidance in its context, and the exchange can repeat for several rounds. A correct answer reached after guidance receives a decayed reward. Our Hint control takes its teacher-written guidance in the student's context (Appendix E.2).

What ZPPO shares with prompt-side guidance, and what it adds. SocraticPO and the hint methods above also keep teacher text in the context and apply the reward only to student-written tokens. ZPPO differs from them in three choices. **(i) What the context holds.** BCQ pairs one correct teacher response with one wrong student response, so the correct candidate always sits beside a wrong one, and the prompt does not say which is correct. SocraticPO's context holds the student's failed attempt and a diagnosis of it, and a hint context holds a correct cue only. Unlike BiCC, BCQ takes the correct response from the teacher, so it also applies when every student rollout on the question is wrong. NCQ lists every parsed wrong rollout of the student's current group at once, with their distinct answers marked wrong. They are changed only by the shared candidate compression, and NCQ inserts no teacher solution or diagnosis. **(ii) How the context is trained.** Each BCQ or NCQ prompt yields its own group of fresh student rollouts under the full binary reward, beside the plain group of the same question. SocraticPO instead continues one interaction trajectory and decays the reward of assisted answers. **(iii) When the context is used.** The prompt replay buffer revisits each hard question, with freshly drawn candidates on every visit the caps admit, until it graduates or is evicted. This amplifies BCQ and NCQ inside the student's zone of proximal development.

C Training dataset and benchmarks

C.1 ZPPO-77K training corpus

We post-train each student as a VLM on ZPPO-77K, a multimodal RL dataset of image–question pairs (Sec. 4). Each item is a triple $(x_{\text{image}}, x_{\text{text}}, a^\star)$: the image, the question, and the reference answer that the grader scores against. We build ZPPO-77K from two public datasets:

- `zlab-princeton/Vero-600k` (Sarch et al., 2026)[1]: a 600K-sample corpus drawn from 59 source datasets over six task categories. We use the 29 Vero sub-datasets listed below, which cover four of those categories. We drop the remaining categories, captioning/instruction-following and spatial-action, because their answers are not short verifiable strings.
- `OpenDataArena/MMFineReason-SFT-586K` (Lin et al., 2026b)[2]: a 586K-sample corpus whose released reasoning traces come from Qwen3-VL-235B-A22B-Thinking. We keep the question, the image, and the reference answer. We drop the trace when we build the corpus, so it never enters any method's loss, target, or prompt. Each example carries a `pass_rate` column: the released difficulty annotation, measured with Qwen3-VL-4B-Thinking over four attempts. The 586K release is already the `pass_rate` $\neq 1$ slice of MMFineReason-1.8M.

We split the sub-datasets into two tiers, each with its own cap per sub-dataset:

- **Tier 1** (cap 2,800 samples per sub-dataset) contains the STEM and Chart/OCR sub-datasets: `ai2d` (Kembhavi et al., 2016), `geo170k` (Gao et al., 2023), `geomverse` (Kazemi et al., 2023), `geoqa_plus` (Cao & Xiao, 2022), `mmk12` (Meng et al., 2025), `cosyn_math/chart/diagram/table` (Yang et al., 2025b), `wemath` (Qiao et al., 2024), `tqa` (Kembhavi et al., 2017), `visualwebinstruct` (Jia et al., 2025), `chartqa` (Masry et al., 2022), `arxivqa` (Li et al., 2024b); on the MMFineReason side, `MMR1` (Leng et al., 2025), `Euclid` (Zhang et al., 2024a), `Geo170K`, `Geo3K`/`Geometry3K` (Lu et al., 2021a), `mm-openr1`, `WeMath`, `VisualWebInstruct`, `BMMR` (Xi et al., 2025), `TQA`, `AI2D`, `ScienceQA` (Lu et al., 2022), `ViRL39K` (Wang et al., 2025a).
- **Tier 2** (cap 1,400 samples per sub-dataset) contains the auxiliary VQA, knowledge, and counting/grounding sub-datasets: `pathvqa` (He et al., 2020), `vqarad` (Lau et al., 2018), `raven` (Zhang et al., 2019), `evochart` (Huang et al., 2024), `infographic_vqa` (Mathew et al., 2022), `aokvqa` (Schwenk et al., 2022), `gqa` (Hudson & Manning, 2019), `iconqa` (Lu et al., 2021b), `vqav2` (Goyal et al., 2017), `visual7w` (Zhu et al., 2016), `tallyqa` (Acharya et al., 2019), `pixelreasoner` (Wang et al., 2025b), `multihop`, `pixmo` (Deitke et al., 2024), `visual_probe`; on the MMFineReason side, `Raven`, `VisualSphinx` (Feng et al., 2025a), `PuzzleQA` (Chia et al., 2024), `LLaVA-CoT` (Xu et al., 2024), `Zebra-CoT` (Li et al., 2025a). MMFineReason (Lin et al., 2026b) lists PuzzleVQA as `PuzzleQA`.

[1] HF: `zlab-princeton/Vero-600k`.

[2] HF: `OpenDataArena/MMFineReason-SFT-586K-Qwen3-VL-235B-Thinking`.

A name map merges duplicates that appear in both sources (e.g., Vero's `stem-ai2d_merged` and MMFineReason's `AI2D`), and Vero takes priority. Inside MMFineReason, we keep only examples with `pass_rate` below 0.5. Vero has no `pass_rate` column, so we cap it without a difficulty filter. This `pass_rate` filter runs once, when we build the corpus. The threshold τ of Sec. 3.1 is separate: it applies to the student's own mean rollout accuracy $\bar{r}_x$ on every visit. Three per-sample filters then apply: answer length ≤ 512 characters, image side $\min(h, w) \geq 100$ pixels, and a prompt-length filter that drops any example longer than the 4,096-token prompt budget of plain rollouts (Tab. 11). That budget covers the question text, the image tokens, and the instruction in Fig. 8. The BCQ and NCQ blocks appended during training do not pass through this filter. The tier caps apply before the three filters, so a cap bounds a sub-dataset's count before filtering, not its share of the final 77K. Every method trains on the same filtered dataset.

We draw every sub-dataset from its own training split. Three settings in Tab. 7 share a source dataset with the corpus: AI2D (`ai2d` and MMFineReason's `AI2D`), WeMath (`wemath` and `WeMath`), and InfoVQA (`infographic_vqa`). We evaluate these three on their held-out splits. The other similar names are different datasets: `tqa`/`TQA` and `raven`/`Raven` match no setting, and `chartqa`, `arxivqa`, and `cosyn_chart` are chart and figure corpora distinct from CharXiv.

C.2 Evaluation benchmarks

The 31-benchmark suite of Sec. 4 has 16 VLM, 10 LLM, and 5 Video settings, drawn from 29 suites. Two suites contribute two settings each. The LLM settings are text-only except HLE, which keeps its small image-bearing subset. OCR^{EN} and OCR^{ZH} are the English and Chinese subsets of OCRBench v2. VMME and VMME^{S} are the same Video-MME questions without and with subtitles. Macro averages (Avg) weight each setting equally. The bootstrap of Appendix G.7 treats each setting as one unit. Tab. 7, Tab. 8, and Tab. 9 give each setting's abbreviation, full name, scoring path, and data source.

The `\boxed{}` grader of Appendix E.1 scores short verifiable answers. The judge scores free-form answers when the boxed span does not parse, or when it parses but does not match. EXACT+JUDGE and RULE+JUDGE run a deterministic scorer first and then the judge. EXACT is that grader, and RULE is the benchmark's own matching rule. DET. uses deterministic scoring only. BENCHJUDGE runs the benchmark's own judge prompt on the same judge checkpoint (*Benchmark-specific judge prompts* paragraph below).

Five suites report an aggregate rather than item accuracy. DynaMath is scored worst-case: each of its 501 seeds counts as correct only when all 10 generated variants are correct. VBlind averages the 7 BlindTest tasks. CharXiv uses the validation split, whose answers are public. Its score is the equal-weight mean of the Reasoning and Descriptive accuracies, each computed as in the official evaluation. MM-Vet uses its own graded $[0, 1]$ correctness score (Fig. 10). OCR^{EN} and OCR^{ZH} are the official OCRBench v2 English and Chinese aggregates. The official evaluator scores each category with that category's own rule and averages the category scores within each language. On these two rows, DET. means this evaluator with no judge, not a single exact-match score over all items.

Decoding configuration. Training-time student and teacher rollouts sample at `temperature=1.0` and `top-p=1.0`, with no top-k and no penalty terms. Evaluation uses one configuration for every setting, method, and checkpoint of the 31-benchmark suite: `temperature=0.6`, `top-p=0.95`, `top-k=20`, `min-p=0`, `presence-penalty=1.5`, `repetition-penalty=1.0`, `max-new-tokens=12288`, `max-model-len=262144`, `min-pixels=256×32×32`, `max-pixels=1280×32×32`. Video inputs use a per-suite frame budget: 16 frames for MVBench, 32 for Video-MMMU, and up to 64 at 1 fps for MMVU. Video-MME takes 16 to 32 frames, set by its clip-length split (short, medium, long). Every method and checkpoint uses the same frame schedule.

Prompt and judge templates. For every setting, we drop the benchmark's own instructions on reasoning and answer format (e.g., "Think step by step" or "put your final answer within `\boxed{}`"). We then append the instruction the student is trained under (Fig. 8): reason inside `<think>` tags and put the final answer in `\boxed{}`. We evaluate every method under that instruction. Every reported score is measured on a candidate-free prompt (Appendix I.10).

Table 7: The 16 VLM evaluation settings (Tab. 1 and Tab. 3).

Abbreviation	Full name	Scoring	Data source (HF)
AI2D	AI2 Diagrams (Kembhavi et al., 2016)	EXACT+JUDGE	`lmms-lab/ai2d`
BabyV	BabyVision (Chen et al., 2026b)	EXACT+JUDGE	`UnipatAI/BabyVision`
CharXiv	CharXiv (`validation`; mean of Reasoning and Descriptive) (Wang et al., 2024d)	BENCHJUDGE	`princeton-nlp/CharXiv`
DynaM	DynaMath (501 seeds $\times$ 10 variants; worst-case accuracy) (Zou et al., 2024)	EXACT+JUDGE	`DynaMath/DynaMath_Sample`
EmbSp	EmbSpatial-Bench (Du et al., 2024)	EXACT+JUDGE	`FlagEval/EmbSpatial-Bench`
InfoVQA	InfographicVQA (Mathew et al., 2022)	DET.	`lmms-lab/DocVQA` (`InfographicVQA`)
MVerse	MathVerse (`testmini`) (Zhang et al., 2024b)	EXACT+JUDGE	`CaraJ/MathVerse-lmmseval`
MVision	MathVision (Wang et al., 2024a)	EXACT+JUDGE	`MathLLMs/MathVision`
MVista	MathVista (`testmini`) (Lu et al., 2023)	RULE+JUDGE	`AI4Math/MathVista`
MMMU^{Pro}	MMMU-Pro (`all` 3 configs) (Yue et al., 2024)	EXACT+JUDGE	`MMMU/MMMU_Pro`
MM-Vet	MM-Vet (v1) (Yu et al., 2023)	BENCHJUDGE	`lmms-lab/MMVet`
OCR^{EN}	OCRBench v2 (English aggregate of category scores) (Fu et al., 2025)	DET.	`ling99/OCRBench_v2`
OCR^{ZH}	OCRBench v2 (Chinese aggregate of category scores) (Fu et al., 2025)	DET.	`ling99/OCRBench_v2`
VisP	VisualPuzzles (Song et al., 2025)	EXACT+JUDGE	`neulab/VisualPuzzles`
VBlind	VLMs-are-Blind (BlindTest, all 7 tasks) (Rahmanzadehgervi et al., 2024)	EXACT+JUDGE	`XAI/vlmsareblind`
WeMath	WeMath (`testmini`) (Qiao et al., 2024)	EXACT+JUDGE	`We-Math/We-Math`

You FIRST think about the reasoning process as an internal monologue and then provide the final answer. The reasoning process MUST BE enclosed within `<think> </think>` *tags. The final answer MUST BE put in* `\boxed{}`.

Figure 8: Instruction appended once, after the question, at training and evaluation.

Table 8: The 10 LLM evaluation settings (Tab. 2, LLM block).

Abbreviation	Full name	Scoring	Data source (HF)
AIME25	AIME 2025 (`AIME2025-I/II`, 30 problems)	EXACT+JUDGE	`opencompass/AIME2025`
AIME26	AIME 2026 (30 problems)	EXACT+JUDGE	`MathArena/aime_2026`
CEval	C-Eval (`val`, all 52 subjects) (Huang et al., 2023)	EXACT+JUDGE	`ceval/ceval-exam`
GPQA-D	GPQA-Diamond (198 rows, gated) (Rein et al., 2023)	EXACT+JUDGE	`Idavidrein/gpqa`
HLE	Humanity's Last Exam (mostly text; small image-bearing subset) (Phan et al., 2025)	BENCHJUDGE	`cais/hle`
IMO-AB	IMO-AnswerBench (Luong et al., 2025)	EXACT+JUDGE	`OpenEvals/IMO-AnswerBench`
MMLU	MMLU (`test`, all subjects) (Hendrycks et al., 2020)	EXACT+JUDGE	`cais/mmlu`
MMLU-Pro	MMLU-Pro (Wang et al., 2024c)	EXACT+JUDGE	`TIGER-Lab/MMLU-Pro`
MMLU-Rd	MMLU-Redux 2.0 (all 57 subjects) (Gema et al., 2024)	EXACT+JUDGE	`edinburgh-dawg/mmlu-redux-2.0`
MultiCh	MultiChallenge (Sirdeshmukh et al., 2025)	BENCHJUDGE	`ScaleAI/MultiChallenge`

Table 9: The 5 Video evaluation settings, drawn from four suites (Tab. 2, Video block).

Abbreviation	Full name	Scoring	Data source (HF)
MMVU	MMVU (`validation`, 1,000 rows) (Zhao et al., 2025)	BENCHJUDGE	`yale-nlp/MMVU`
MVBench	MVBench (20 task configs) (Li et al., 2024a)	RULE+JUDGE	`OpenGVLab/MVBench`
VMME	Video-MME (without subtitles) (Fu et al., 2024)	RULE+JUDGE	`lmms-lab/Video-MME`
VMME^{S}	Video-MME (with subtitles) (Fu et al., 2024)	RULE+JUDGE	`lmms-lab/Video-MME`
VMMMU	Video-MMMU (Hu et al., 2025c)	RULE+JUDGE	`lmms-lab/VideoMMMU`

The judge runs only after the deterministic scorer gives 0. It receives the extracted snippet when a \boxed{} span parses but does not match. When no strict span parses, it falls back to the full response. The judge is Qwen3.5-27B-FP8, the same checkpoint as the frozen teacher. It runs on a separate pool at `temperature=0.0`, `top-p=1.0`, and `max-new-tokens=512`. Fig. 9 gives its template, whose {{ ... }} fields are filled at call time. Its parser accepts only {"verdict": "correct"} or {"verdict": "wrong"} and scores every other output 0. The checkpoint, template, parser, and fallback are the same for every method and for the untrained base. Appendix I.5 rescores the same fixed responses with seven other judges on the shared-template settings.

You are an expert judge evaluating whether a model's response is correct according to the ground truth.
Your Role (READ FIRST)
You are a COMPARATOR, not a solver. Your ONLY job is to decide how well the model's response matches the provided GROUND TRUTH.

- *Do NOT attempt to solve the problem yourself.*
- *Do NOT infer or derive what the "real" answer should be.*
- *Do NOT second-guess the ground truth, even if it seems wrong.*
- *The "Model Response" section below MAY be either (a) the model's full trajectory including any <think>...</think> reasoning and a concluding \boxed{...} answer, OR (b) a short EXTRACTED snippet that was already pulled out as the candidate final answer (e.g. a bare letter like B, a number like 42, an expression like x = 5/3). Treat both forms uniformly: the snippet (when given) IS the model's committed final answer, and a full trajectory's final answer is the content inside the last \boxed{...} or the concluding sentence AFTER any </think> tag.*
- *If the final answer is missing due to answer format error, truncation, or other reasons, look inside thinking or reasoning.*
- *You should OUTPUT JSON with the format {"verdict": "correct" or "wrong"} based on the model's response and the ground truth.*

Question
{{ question }}
Answer Options (rendered only when non-empty)
{{ options }}
Model Response
{{ model_response }}
Ground Truth Answer
{{ ground_truth }}
Output ONLY a valid JSON object with exactly this format (no prose, no code fences, no trailing commentary): {"verdict": "correct" or "wrong"}

Figure 9: Shared judge template. The trailing {"verdict": "correct" or "wrong"} is sent to the judge as written.

Benchmark-specific judge prompts. Five suites use templates adapted from their upstream evaluators: MM-Vet, MultiChallenge, MMVU, HLE, and CharXiv. CharXiv has separate Reasoning and Descriptive templates. We keep the upstream prompt logic and run it on the shared judge checkpoint, Qwen3.5-27B-FP8. The upstream evaluators use GPT-4o for CharXiv, MMVU, and MultiChallenge, GPT-4 for MM-Vet, and o3-mini for HLE. The figures below give the executed instruction text and runtime fields. For space, they shorten two fixed blocks: the rest of MM-Vet's few-shot table and the per-category CharXiv rubric bodies. The executed prompts contain both in full.

MM-Vet (Fig. 10) grades correctness on $[0, 1]$ using the dataset's <AND>/<OR> tags.

MultiChallenge (Fig. 11) uses the upstream YES/NO verifier. Its `response` field is the text after </think>, or the boxed answer if that text is empty. The shared fallback does not apply to this verifier.

MMVU (Fig. 12) uses the two-mode upstream template (Zhao et al., 2025): one mode for multiple-choice questions and one for open-ended questions.

HLE (Fig. 13) uses the upstream CAIS template. The judge reads the full response and returns four fields, with `correct` $\in$ {`yes`, `no`}.

Compare the ground truth and prediction from AI models, to give a correctness score for the prediction. <AND> in the ground truth means it is totally right only when all elements in the ground truth are present in the prediction, and <OR> means it is totally right when any one element in the ground truth is present in the prediction. The correctness score is 0.0 *(totally wrong),* 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, *or* 1.0 *(totally right). Just complete the last space of the correctness score.*
Question — Ground truth — Prediction — Correctness
— — — — — — —
What is x in the equation? — -1 <AND> -5 — x = 3 — 0.0
What is x in the equation? — -1 <AND> -5 — x = -1 — 0.5
What is x in the equation? — -1 <AND> -5 — x = -1 or x = -5 — 1.0
. . . (remaining fixed MM-Vet exemplars, present in the executed prompt)
{{ question }} — {{ ground_truth }} — {{ prediction }} —

Figure 10: MM-Vet judge prompt (excerpt).

You are tasked with evaluating a model response to see if it meets a specific criteria. The criteria will always be YES/NO evaluation.
The model response is as follows:
<MODEL_RESPONSE>
{{ response }}
</MODEL_RESPONSE>
The criteria that the model response must meet is as follows. Be VERY STRICT!:
<CRITERIA>
{{ target_question }}
</CRITERIA>
Print your reasoning followed by your verdict, either "YES" or "NO".

Figure 11: MultiChallenge YES/NO verifier template.

Evaluate whether the model's final answer is correct by comparing it to the ground-truth answer provided for the given question. You should first extract the final answer from the model's response, and then compare the extracted answer with the ground-truth answer to determine its accuracy.
(Open-ended branch only) The final answer generated by the model does not need to match the ground-truth answer word-for-word. However, it should ONLY be considered correct if it demonstrates the exact same technique or concept explicitly and unambiguously equivalent to the ground-truth answer.
Question {{ question }}
Options (MCQ only) {{ options }}
Ground Truth Answer {{ ground_truth }}
Model's Full Response {{ model_response }}
Output ONLY a valid JSON object with exactly this format (no prose, no code fences, no trailing commentary): {"verdict": "correct" or "wrong"}

Figure 12: MMVU two-mode judge template.

Judge whether the following [response] to [question] is correct or not based on the precise and unambiguous [correct_answer] below.
[question]: {{ question }}
[response]: {{ response }}
Your judgement must be in the format and criteria specified below:
extracted_final_answer: *The final exact answer extracted from the [response]. Put the extracted answer as 'None' if there is no exact, final answer to extract from the response.*
[correct_answer]: {{ correct_answer }}
reasoning: *Explain why the extracted answer is correct or incorrect based on [correct_answer], focusing only on whether there are meaningful differences. Do not solve the problem, do not argue for any answer different than [correct_answer], focus only on whether the answers match.*
correct: *Answer 'yes' if extracted_final_answer matches [correct_answer] (or is within a small margin of error for numerical problems), 'no' otherwise.*
confidence: *Confidence score* $\in [0\%, 100\%]$ *extracted from [response] (default* 100*).*
Respond with ONLY a single JSON object using exactly these keys: extracted_final_answer, reasoning, correct *("yes"/"no"),* confidence *(integer* 0–100*).*

Figure 13: HLE judge template.

CharXiv fills the runtime `rules` field of its Reasoning wrapper (Fig. 14) with one of four rubric bodies: text-in-chart, text-in-general, number-in-chart, or number-in-general. The runtime `rubric` field of its Descriptive wrapper (Fig. 15) takes one of seven classes: title, ocr, quant, bool, enum, trend, or layout.

You will be given a question, a ground truth answer and a model response. You need to extract the final answer from the model response, compare it with the ground truth answer, and then assign a binary score. Avoid providing explanations in your response. If there is no provided model response, please leave the extracted answer empty and give a score of 0.
Your response must follow JSON format with keys [extracted_answer, score] *where* score$\in \{0, 1\}$. *You must follow the scoring rules:*
{{ rules }}
Your Turn
* *Question:* {{ question }}
* *Ground Truth:* {{ ground_truth }}
* *Response:* {{ response }}
Respond with ONLY a single JSON object using exactly these keys: extracted_answer *(string),* score *(integer* 0 *or* 1*).*

Figure 14: CharXiv Reasoning judge-prompt wrapper.

You will be given a pair of ground truth answer and model response under an overarching question. You need to extract the final answer from the model response, compare it with the ground truth answer, and then assign a binary score. Avoid providing explanations in your response. If there is no provided model response, please leave the extracted answer empty and give a score of 0*. Your response must follow JSON format with keys* [extracted_answer, score] *where* score$\in \{0, 1\}$.
Overarching Question: {{ overarching_question }}
{{ rubric }}
Your Turn
Response: {{ response }}
Ground Truth: {{ ground_truth }}
Respond with ONLY a single JSON object using exactly these keys: extracted_answer *(string),* score *(integer* 0 *or* 1*).*

Figure 15: CharXiv Descriptive judge-prompt wrapper.

D ZPPO ALGORITHM

Algorithm 1 gives one ZPPO training step, where x denotes a question. Training is on-policy in the response-level sense of Sec. 3.2: every gradient-counted token is sampled by the student. Each group is sampled by the student policy of the current rollout step (π_{old} in Eq. (7)) under its own prompt. The frozen teacher π_{T} reaches the student only through the prompt. Its correct rollouts form the pool from which BCQ draws candidates. It also rewrites every BCQ and NCQ candidate before the candidate enters a prompt (candidate compression). No teacher response is ever a gradient target. We call x a *hard question* when the mean rollout accuracy $\bar{r}_x$ of its plain rollouts is below τ. Every run outside the sweep of Tab. 10 uses $\tau = 0.5$.

ZPPO is built on GRPO (Shao et al., 2024) with three DAPO ingredients (Yu et al., 2025). It uses the two-step REINFORCE++ estimator (Hu et al., 2025a) and $I = 4$ iterations per step (vs. DAPO's $I = 16$). Algorithm 1 tags each piece by its source. GRPO gives (G1) the group-relative advantage and (G2) a PPO-style clipped surrogate (Schulman et al., 2017). REINFORCE++ gives (R1) the two-step estimator of Eqs. (2)–(3). The DAPO ingredients are (Da1) clip-higher with $(\epsilon_{\text{low}}, \epsilon_{\text{high}}) = (0.20, 0.28)$, (Da2) a token-level policy gradient loss, and (Da3) removal of the KL penalty against a reference policy. (Y1) is the dual clip of Ye et al. (2020) with $c_{\text{dual}} = 10$ (Eq. (8)), which bounds the surrogate of a negative-advantage token. GRPO and GRPO† use it as well. (Z0) is *Norm w/o Zero*, which excludes zero-advantage groups from the batch statistics. (Z1)–(Z3) are BCQ, NCQ, and the prompt replay buffer. (Z4) marks the update that uses all three together.

Properties of the loop. BCQ needs both a correct teacher rollout and a parsed wrong student rollout, so a question the teacher also fails on can receive NCQ only. Appendix G.8 measures how often the 27B teacher succeeds. A hard question always has a wrong rollout, so every question in $X_{\text{aug}}^{\text{pre}}$ receives an NCQ instance in Step 4 whenever one of its wrong rollouts parses. A question with no parsed wrong rollout adds no instance, but its plain rollouts still update the buffer in Step 6. Teacher rollouts are sampled again on every visit ($G_{\text{T}} = 4$, Tab. 11). The student candidates come from the current plain group. Every visit the caps admit therefore builds BCQ and NCQ from fresh candidates. Only plain rollouts drive admission and graduation. A buffer entry's hardness comes from its most recent plain rollouts. The buffer stores only the original question x, never a response and never a BCQ or NCQ prompt. In ZPPO, the buffer serves to amplify BCQ and NCQ on questions the student has not yet mastered. GRPO† runs the same buffer without reformulation.

What "on-policy" means for a reformulated group. A plain group, on a new or a replayed question, trains on the original question x. A BCQ or NCQ group trains on its reformulated prompt x', whose teacher-compressed text is input only. All three group types apply the same clipped surrogate under their own prompt. The only ratio in the objective is the PPO ratio of Eq. (7), between the current parameters and those that sampled the group. No student–teacher ratio appears. Placing a teacher rollout in a group would instead treat a response drawn from $\pi_{\text{T}}(\cdot \mid x)$ as if it came from $\pi_\theta(\cdot \mid x)$, unless a student–teacher ratio corrects it. LUFFY (Yan et al., 2025) adds teacher traces to the group. Its objective includes this ratio, but in practice LUFFY sets the teacher probability to 1 and reshapes the student's token probability with regularized importance sampling. ZPPO never places a teacher rollout in a group, so a reformulated group is on-policy in the same response-level sense as a plain one. The plain and reformulated groups of the same question enter the same rollout step. Each group is on-policy for the prompt x' that sampled it. Every reported score is measured on the original question x, which the update on x' reaches through the shared parameters, and Appendix I.10 measures that transfer.

What one rollout step contains. Both ρ_{replay} and ρ_{aug} are fractions of the new-question count $|X_{\text{new}}|$. Replayed questions add to the new questions, and so do BCQ and NCQ instances. Each has its own budget. One step draws 384 new questions from $\mathcal{D}$ and up to 96 replayed questions from $\mathcal{B}$. It keeps at most 96 BCQ and NCQ instances after the cap (Tab. 11). In total, a step has at most 576 groups of $G_{\text{S}} = 8$ rollouts each. When the buffer supplies the full replay count and the cap is full, one group in six is a BCQ or NCQ group. The two caps count different objects. Step 2 of Algorithm 1 caps the *hard questions* that enter the reformulation branch at $\rho_{\text{aug}}|X_{\text{new}}| = 96$. Step 5 caps the BCQ and NCQ *instances* that reach the update at the same number. A question that keeps both its BCQ and its NCQ instance fills two slots. Whenever any question keeps both, fewer than 96 questions therefore pass the instance cap. Appendix G.8 reports the measured question and instance counts.

Algorithm 1: One executed ZPPO training step. G1–G2 come from GRPO (Shao et al., 2024), Da1–Da3 from DAPO (Yu et al., 2025), Y1 from dual-clip PPO (Ye et al., 2020), and R1 from REINFORCE++ (Hu et al., 2025a). Z0 is *Norm w/o Zero*, Z1–Z3 are BCQ, NCQ, and the prompt replay buffer, and Z4 marks their combination.

Input: Student π_θ, frozen teacher π_{T}, dataset $\mathcal{D}$, buffer $\mathcal{B}$; $G_{\mathrm{S}}, G_{\mathrm{T}}, \rho_{\mathrm{replay}}, \rho_{\mathrm{aug}}, |\mathcal{B}|_{\max}, I, \tau = 0.5$.
Output: Updated π_θ, $\mathcal{B}$.

// 1. Plain rollouts. *[Z3]*

Draw $X_{\mathrm{new}} \sim \mathcal{D}$. Before Step 6 refreshes $\mathcal{B}$, draw X_{replay} from $\mathcal{B}$ uniformly without replacement, with $|X_{\mathrm{replay}}| = \min(\rho_{\mathrm{replay}}|X_{\mathrm{new}}|, |\mathcal{B}|)$ (every buffer entry when $|\mathcal{B}|$ is below the full replay count, and none on step 1). Sampling does not remove an entry from $\mathcal{B}$. Step 6 decides whether it stays. Set $X \leftarrow \mathrm{concat}(X_{\mathrm{new}}, X_{\mathrm{replay}})$. For each $x \in X$, draw G_{S} *plain* rollouts $\{y_{\mathrm{S}}^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot|x)$, score them, compute $\bar{r}_x$, and collect the wrong ones $\{y_{\mathrm{S}}^{(-)}(x)\}$.

// 2. Hard questions and the question cap. *[Z1, Z2]*

Rank $X_{\mathrm{hard}} = \{x \in X : \bar{r}_x < \tau\}$ by ascending $\bar{r}_x$ alone, hardest first, and keep at most the $\rho_{\mathrm{aug}}|X_{\mathrm{new}}|$ hardest as $X_{\mathrm{aug}}^{\mathrm{pre}}$. This cap counts questions. Step 5 caps instances.

// 3. Teacher rollouts, drawn in parallel with Step 1. *[Z1]*

For each $x \in X$, new or replayed, draw $\{y_{\mathrm{T}}^{(g)}(x)\}_{g=1}^{G_{\mathrm{T}}} \sim \pi_{\mathrm{T}}(\cdot|x)$, score them with the same reward, and keep the correct ones $\{y_{\mathrm{T}}^{(+)}(x)\}$ as the BCQ pool. Step 4 uses the pool only on $X_{\mathrm{aug}}^{\mathrm{pre}}$.

// 4. Build BCQ and NCQ prompts. *[Z1, Z2]*

Set $\mathcal{A} \leftarrow \emptyset$. Write $\{y_{\mathrm{S}}^{(-)}\}_{\mathrm{p}}$ for the wrong rollouts on x whose final `\boxed{}` span parses. The others enter neither branch (Appendix E.1).

foreach $x \in X_{\mathrm{aug}}^{\mathrm{pre}}$ **do**

- **if** $\{y_{\mathrm{T}}^{(+)}\} \neq \emptyset$ *and* $\{y_{\mathrm{S}}^{(-)}\}_{\mathrm{p}} \neq \emptyset$ **then** sample one rollout from each set uniformly, teacher-compress both (shared prompt and token cap), anonymize them inside identical `<candidate>` tags, shuffle their order, and append them to x with the BCQ instruction as x_{BCQ}. Add (x_{BCQ}, x) to $\mathcal{A}$
- **if** $\{y_{\mathrm{S}}^{(-)}\}_{\mathrm{p}} \neq \emptyset$ **then** list the *distinct* parsed answers, teacher-compress *every* rollout in $\{y_{\mathrm{S}}^{(-)}\}_{\mathrm{p}}$ as its own `<candidate>` block, and append the list and the blocks to x with the NCQ instruction as x_{NCQ}. Add (x_{NCQ}, x) to $\mathcal{A}$

end

// 5. Instance cap and reformulated rollouts. *[Z1, Z2]*

if $|\mathcal{A}| > \rho_{\mathrm{aug}}|X_{\mathrm{new}}|$ **then** order $\mathcal{A}$ by the Step 2 rank of its source question, each question's BCQ instance before its NCQ instance, and keep the first $\rho_{\mathrm{aug}}|X_{\mathrm{new}}|$

For each $(x', x) \in \mathcal{A}$, draw G_{S} rollouts $\{y^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot|x')$ as their own group and score them with the same reward.

// 6. Update and buffer refresh. *[G1, G2, Da1–Da3, Y1, R1, Z0, Z3, Z4]*

The plain, BCQ, and NCQ rollouts on each x form up to three groups of G_{S} rollouts. Each group has its own mean $\bar{r}_g$ and standard deviation std_g. Split the groups across I optimizer mini-batches, keeping each group whole.

foreach *optimizer mini-batch* $b = 1, \ldots, I$ **do**

- Subtract each group's mean (Eq. (2)), then normalize by the statistics of the rollout-level A' over the non-trivial groups $\mathcal{G}^\star = \{g : \mathrm{std}_g > 0\}$ of b (Eq. (3)). Trivial groups keep $A^{(g,i)} = 0$.
- **if** $\mathcal{G}^\star = \emptyset$ **then** skip the update, since every advantage in b is zero
- Otherwise, update π_θ with the token-level dual-clip surrogate of Eq. (8) (with clip-higher and without a KL penalty against a reference policy).

end

Refresh $\mathcal{B}$ from the *plain*-rollout $\bar{r}_x$ only: admit or keep each x with $\bar{r}_x < \tau$, graduate the buffer entries with $\bar{r}_x \geq \tau$, and FIFO-evict the oldest entries until $|\mathcal{B}| \leq |\mathcal{B}|_{\max}$.

return π_θ, $\mathcal{B}$.

Step 6 splits all these groups across the $I=4$ optimizer mini-batches, so every response token enters exactly one of them.

What Fig. 5a reports. Fig. 5a summarizes the 200 rollout steps of one logged run per scale, read before the caps of Algorithm 1 apply. *Easy* counts the questions whose plain rollouts reach $\bar{r}_x \geq 0.5$ and need no reformulation. *BCQ* and *NCQ* count the reformulations that Step 4 could build on the hard questions. BCQ needs at least one correct teacher rollout among the $G_{\mathrm{T}}=4$, and NCQ needs only a parsed wrong student rollout. A hard question with a correct teacher rollout can yield both, so the two counts overlap. The three shares are therefore not a split of the questions. Each is a fraction of the summed Easy, BCQ, and NCQ count. Only BCQ depends on the teacher solving the question, so the BCQ share falls with student scale while the NCQ share stays near one third. After the caps, BCQ and NCQ groups make up at most one group in five of a step, and at most one in six once the buffer supplies the full replay count (*What one rollout step contains*). Appendix G.8 counts the instances and questions that pass the caps and measures the teacher's success rate on those questions.

Two-step advantage estimator (REINFORCE++) used by ZPPO. We use the two-step REINFORCE++ variant of Hu et al. (2025a), which they call *REINFORCE++ w/ Baseline* and define for $k>1$ samples per prompt. As in Sec. 3.1, Step 1 subtracts the per-group mean. Step 2 applies batch-level advantage normalization: it divides by a standard deviation taken across groups, not by each group's own std_g. Hu et al. (2025a) take this statistic over every group of the global batch. We take it over one optimizer mini-batch, and only over its non-trivial groups (Z0, *Norm w/o Zero*). Let $\mathcal{G}$ be the set of groups in one optimizer mini-batch, each holding G_{S} rollouts. Write $r^{(g,i)}$ for the i-th reward of group g, $\bar{r}_g$ and std_g for its within-group mean and standard deviation, and $\mathcal{G}^\star=\{g:\mathrm{std}_g>0\}$ for the *non-trivial* subset. The other groups, all-wrong or all-correct, are *trivial*. Plain, BCQ, and NCQ groups all enter these statistics. Groups in other mini-batches or other rollout steps do not. Below, mean and std are population statistics over the rollout-level entries in the subscript. The small constant $\epsilon>0$ is separate from ϵ_{low} and ϵ_{high}. Step 1 is

$$A'^{(g,i)} \;=\; r^{(g,i)} - \bar{r}_g, \tag{2}$$

which gives $A'=0$ for every rollout of a trivial group. Step 2 normalizes across a subset $\mathcal{S}\subseteq\mathcal{G}$ of groups:

$$\begin{aligned} A^{(g,i)} &= \frac{A'^{(g,i)} - \mu_{\mathcal{S}}}{\sigma_{\mathcal{S}} + \epsilon}, \\ \mu_{\mathcal{S}} &= \mathrm{mean}_{(g,i):g\in\mathcal{S}}\, A'^{(g,i)}, \\ \sigma_{\mathcal{S}} &= \mathrm{std}_{(g,i):g\in\mathcal{S}}\, A'^{(g,i)}. \end{aligned} \tag{3}$$

Step 1 makes every group sum to zero,

$$\sum_{i=1}^{G_{\mathrm{S}}} A'^{(g,i)} \;=\; \sum_{i=1}^{G_{\mathrm{S}}} \big(r^{(g,i)} - \bar{r}_g\big) \;=\; 0 \quad \text{for every } g\in\mathcal{G}, \tag{4}$$

so $\mu_{\mathcal{S}}=0$ for either choice of $\mathcal{S}$ below. The three settings of Sec. 4.4(ii) all run Step 1. They differ only in $\mathcal{S}$ and in whether Step 2 runs at all:

- *No norm*: Step 2 is skipped, so $A^{(g,i)}=A'^{(g,i)}$ and only the group mean is subtracted. This is Eq. (1) without its denominator.
- *Norm w/o Zero* (ZPPO default): $\mathcal{S}=\mathcal{G}^\star$, so Step 2 takes its statistics over the non-trivial groups only, and trivial groups keep $A^{(g,i)}=0$ exactly.
- *Norm w/ Zero*: $\mathcal{S}=\mathcal{G}$, so Step 2 also counts the trivial groups, whose $A'=0$. By Eq. (4), their zeros leave the mean at zero, but they pull the batch standard deviation down. With $f_{\mathrm{nt}}=|\mathcal{G}^\star|/|\mathcal{G}|$ and equal group size G_{S},

$$\begin{aligned} \sigma^2_{\mathcal{G}} &= \frac{1}{|\mathcal{G}|\,G_{\mathrm{S}}} \sum_{g\in\mathcal{G}}\sum_{i} \big(A'^{(g,i)}\big)^2 \\ &= \frac{|\mathcal{G}^\star|}{|\mathcal{G}|}\cdot\frac{1}{|\mathcal{G}^\star|\,G_{\mathrm{S}}} \sum_{g\in\mathcal{G}^\star}\sum_{i} \big(A'^{(g,i)}\big)^2 \\ &= f_{\mathrm{nt}}\,\sigma^2_{\mathcal{G}^\star}, \end{aligned} \tag{5}$$

so $\sigma_{\mathcal{G}} = \sqrt{f_{\text{nt}}}\,\sigma_{\mathcal{G}^\star}$. Substituting this into Eq. (3) gives

$$A^{(g,i)}_{\text{w/Zero}} = \frac{\sigma_{\mathcal{G}^\star} + \epsilon}{\sqrt{f_{\text{nt}}}\,\sigma_{\mathcal{G}^\star} + \epsilon} A^{(g,i)}_{\text{ZPPO}} \quad (g \in \mathcal{G}^\star),$$
$$A^{(g,i)}_{\text{w/Zero}} = 0 \quad (g \notin \mathcal{G}^\star). \tag{6}$$

Trivial groups have zero advantage under both settings. Every non-trivial advantage, however, is multiplied by a factor in $[1, 1/\sqrt{f_{\text{nt}}}]$ relative to ZPPO. The share f_{nt} is recomputed on each of the $I = 4$ mini-batches. Within one mini-batch this factor is a single scalar. It rescales the policy-gradient estimate without changing its direction. AdamW's moment estimates and weight decay can still change the resulting parameter update. AdamW normalizes each step by a running average of past squared gradients, so it cancels a scale that every mini-batch shares but not one that changes from one mini-batch to the next. Across mini-batches the factor moves with f_{nt}, so the size of each update depends on how many trivial groups its mini-batch happens to hold. The factor is largest on the mini-batches with the fewest non-trivial groups, whose gradient comes from the fewest rollouts. Both series of the normalization sweep, GRPO† and ZPPO, use the prompt replay buffer, which brings hard questions back on every step. ZPPO also builds its reformulated groups on the hardest of them. All-wrong groups therefore add to the all-correct ones, so f_{nt} is small and uneven across mini-batches. A pipeline that does not concentrate hard questions would have a larger, steadier f_{nt} and a factor closer to one. *Norm w/o Zero* computes σ on $\mathcal{G}^\star$ alone, so the scale of its non-trivial advantages does not depend on the share $1 - f_{\text{nt}}$ of trivial groups. *No norm* also rescales every non-trivial advantage relative to ZPPO, by $\sigma_{\mathcal{G}^\star} + \epsilon$. At $G_{\text{S}} = 8$, every non-trivial group has $\text{std}_g^2 \in [7/64, 1/4]$ (Eq. (9)), so $\sigma_{\mathcal{G}^\star} \in [0.33, 0.5]$, and this factor varies by at most $1.5\times$ across mini-batches. The *Norm w/ Zero* factor has no fixed bound, since it grows as f_{nt} falls. We read the drop of *Norm w/ Zero* in Fig. 6d–f as consistent with this varying scale, not as a controlled test of it.

Every RL run outside the normalization sweep of Fig. 6d–f uses *Norm w/o Zero*. Appendix G.5 reports that sweep in all six (series, family) cells.

Per-token clipped surrogate. For response token t, let ρ_t be the ratio between the current policy and the policy that sampled the rollout (Schulman et al., 2017):

$$\rho_t = \frac{\pi_\theta(y_t \mid x, y_{<t})}{\pi_{\text{old}}(y_t \mid x, y_{<t})}. \tag{7}$$

For BCQ and NCQ, x is the reformulated prompt. Only student response tokens enter Eq. (7) and the surrogate below. The loss is aggregated over those tokens (DAPO's token-level policy gradient loss). Write A for the rollout's advantage $A^{(g,i)}$ from Eq. (3), shared by all of its tokens. Let $q(\rho_t, A) = \min\big(\rho_t A,\ \text{clip}(\rho_t, 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}})A\big)$, with DAPO's clip-higher $(\epsilon_{\text{low}}, \epsilon_{\text{high}}) = (0.20, 0.28)$. The per-token surrogate adds the dual clip of Ye et al. (2020) on top of this asymmetric clip, with the dual-clip ratio $c_{\text{dual}} = 10$ of Tab. 11:

$$\ell(\rho_t, A) = \begin{cases} \max\big(q(\rho_t, A),\ c_{\text{dual}} A\big), & A < 0, \\ q(\rho_t, A), & A \geq 0. \end{cases} \tag{8}$$

The update maximizes ℓ, which for $A \geq 0$ is the asymmetric clip alone. For $A < 0$, $q(\rho_t, A)$ falls without bound as ρ_t grows. The second argument of the max keeps ℓ at or above $c_{\text{dual}} A$. This floor takes effect only where $\rho_t > c_{\text{dual}}$.

Why the threshold sits at one half. Sec. 3.1 sets the hard-question threshold at $\tau = 0.5$, the majority-failure boundary: below it, most rollouts on x fail. With binary rewards, a group's spread is fixed by its mean,

$$\text{std}_g^2 = \frac{1}{G_{\text{S}}} \sum_{i=1}^{G_{\text{S}}} \big(r^{(g,i)} - \bar{r}_g\big)^2 = \bar{r}_g\,(1 - \bar{r}_g), \tag{9}$$

which is zero at $\bar{r}_g \in \{0, 1\}$ and peaks at $\bar{r}_g = 0.5$. Eq. (3) divides every group by the same $\sigma_{\mathcal{S}}$, so the root-mean-square advantage of group g is $\text{std}_g/(\sigma_{\mathcal{S}} + \epsilon)$. Below τ, the plain group's signal therefore shrinks as the question gets harder and vanishes at $\bar{r}_x = 0$. Above τ, it shrinks as the question gets easier. The all-wrong case motivates BCQ and NCQ, but $\tau = 0.5$ also admits partially

solved questions ($\bar{r}_x \in \{\frac{1}{8}, \frac{2}{8}, \frac{3}{8}\}$), whose plain groups already carry a nonzero advantage. On those questions, a reformulated group adds to the plain group's signal instead of being the only signal.

At $G_S = 8$, a group's mean takes one of the nine values $k/8$, so τ only selects how many of these bins count as hard. Every $\tau \in (0.375, 0.5]$ admits the same four bins $\{0, \frac{1}{8}, \frac{2}{8}, \frac{3}{8}\}$, and the distinct settings are the bin counts from one to eight. Tab. 10 sweeps five of them at 2B. ZPPO is stable from $\tau = 0.3$ to 0.9: every family average stays within 0.7 pp of the $\tau = 0.5$ row. The default is not the best value on every family: $\tau = 0.7$ is higher on LLM and Video, and $\tau = 0.9$ is higher on VLM. We therefore treat $\tau = 0.5$ as a default, not a tuned value. The one-bin setting $\tau = 0.1$ lands lower. It admits all-wrong groups only, and an entry graduates as soon as one rollout in eight is correct. One correct rollout in eight is weak evidence that the student can solve a question. For example, a student that picks one of four options at random still answers correctly in at least one of eight rollouts with probability $1 - (3/4)^8 \approx 0.90$, but in at least four with probability only about 0.11. A single correct rollout can also be a lucky answer or a right answer reached through wrong reasoning, since the reward checks only the final answer. Under $\tau = 0.1$, a question with one correct rollout in eight is no longer hard. It leaves the reformulation branch and the buffer, and that rollout receives the largest advantage in its plain group. Under $\tau = 0.5$, it stays eligible for BCQ and NCQ and stays in the buffer until the student solves it in at least half of its rollouts. Every ZPPO row, $\tau = 0.1$ included, stays above GRPO† on all three families. Every run outside this sweep uses $\tau = 0.5$.

Table 10: ZPPO at 2B under five hard-question thresholds τ. Each cell is the macro average over the 10 LLM, 16 VLM, or 5 Video benchmarks. The five ZPPO rows admit 1, 3, 4, 6, and 8 of the $k/8$ bins. *Base* is the untrained Qwen3.5-2B checkpoint. GRPO has no τ. GRPO† uses τ for buffer admission and graduation and is shown at the default $\tau = 0.5$. Every other setting follows Tab. 11.

Method	τ	LLM	VLM	Video
Base	–	45.3	56.8	60.6
GRPO	–	46.7	58.7	61.2
GRPO†	0.5	47.4	59.2	61.9
ZPPO	0.1	47.9	59.9	62.1
	0.3	49.8	61.5	62.5
	0.5	50.4	62.0	63.2
	0.7	50.5	61.4	63.4
	0.9	50.1	62.3	62.8

Dynamic sampling: disabled. DAPO's dynamic sampling (Yu et al., 2025) drops every question whose rollouts are all correct or all wrong, and keeps sampling until the batch is full. ZPPO keeps those questions in the step, and their plain groups keep zero advantage. Questions whose plain group is all-wrong are the hardest, so they rank first for BCQ and NCQ. Step 6 also admits them to the buffer. Every RL run in this paper, GRPO and GRPO† included, has dynamic sampling disabled (`Online filtering: disabled` in Tab. 11).

E Hyperparameters and baseline implementations

E.1 ZPPO hyperparameters

Tab. 11 lists every optimization, rollout, and replay setting of ZPPO. Every RL row shares each setting that applies to it. Every ZPPO and baseline run uses the same GPU layout: 6 GPUs per node run student rollouts and optimization, and 2 run the frozen teacher, candidate compression, and the judge.

What the "Base" rows are. Every method starts from the same released Qwen3.5 checkpoint at each size. The checkpoint is natively multimodal, so there is no vision–language conversion stage. The *Base* row of every table is this checkpoint before our post-training, under the same evaluation setup. "Untrained" means untrained by us, since its developers have already post-trained the released checkpoint. Each Base cell uses the shared decoding of Appendix C.2.

Table 11: Training hyperparameters of ZPPO. Settings shared with GRPO and GRPO† stay fixed across every RL row of Tab. 1–Tab. 5. An appendix row changes only the setting or term it names. Distillation and supervised rows use the same $I = 4$ optimizer mini-batches of 96 new prompts per rollout step.

Group	Hyperparameter	Value
Student / teacher	Student initialization	`Qwen/Qwen3.5-{0.8,2,4,9}B` (HF)
	Teacher	`Qwen/Qwen3.5-27B-FP8` (HF), frozen
	GPUs per node (student / teacher)	6 / 2
	Total GPUs	64 NVIDIA H100-80 GB (8 nodes × 8)
Optimization	Optimizer	AdamW (`AnyPrecisionAdamW`, bf16 state)
	Learning rate	1.0×10^{-6} (constant for the full run)
	LR scheduler	constant, no warmup, no decay
	AdamW (β_1, β_2)	$(0.9,\ 0.999)$
	AdamW ϵ	10^{-8}
	Weight decay	0.1 (decoupled)
	Gradient clipping (max grad norm)	1.0
	PPO inner epochs (over a global batch)	1 (each global batch is used once)
	Mixed precision	bfloat16
	FSDP sharding	full-shard, bfloat16, rank-0 init
	Fused LM-head kernel	on, PyTorch backend
	KL penalty against reference policy	0 (following DAPO)
	Reference reset count	0
	Online filtering	disabled
Rollouts	Student group size G_{S}	8
	Teacher group size G_{T} (BCQ pool)	4
	Tensor-parallel size (student / teacher)	1 / 1
	Sampling temperature (student / teacher)	1.0 / 1.0 (training-time rollouts)
	Top-p (student / teacher)	1.0 / 1.0 (no nucleus truncation)
	Max prompt length	4,096 tokens, plain prompts only
	Max response length (student / teacher)	12,288 / 12,288 tokens
GRPO backbone	Reward	binary $\in \{0, 1\}$ (grader, then shared judge)
	Advantage estimator	two-step, REINFORCE++ (Hu et al., 2025a)
	Clip-lower ϵ_{low}	0.20
	Clip-higher ϵ_{high}	0.28 (DAPO clip-higher)
	Dual clip ratio	10.0
	Loss aggregation	token-level (DAPO)
Recipe choices	Iterations per step I	4 optimizer mini-batches per rollout step
	Batch advantage normalization	on (*Norm w/o Zero*)
ZPPO-specific	Hard-question threshold τ	0.5 on $\bar{r}_x$ (swept in Tab. 10)
	Replay fraction ρ_{replay}	0.25 of $\lvert X_{\mathrm{new}} \rvert$
	Prompt replay buffer capacity $\lvert\mathcal{B}\rvert_{\max}$	10,000 prompts
	Augmentation fraction ρ_{aug}	0.25 of $\lvert X_{\mathrm{new}} \rvert$
	Eviction policy	FIFO once $\lvert\mathcal{B}\rvert > \lvert\mathcal{B}\rvert_{\max}$
	Candidate compression	on, max 512 tokens
Schedule & batching	Data schedule	$200 \times 384 = 76{,}800$ draws from ZPPO-77K
	Micro-batch size per device (update / exp.)	4 / 4
	Global batch size	96 *new* prompts per update (4 accum. × 4 micro × 48 GPUs $/\ G_{\mathrm{S}}$); up to 144 with replayed and BCQ/NCQ prompts
	Rollout batch size	384 new, up to 96 replayed, up to 96 BCQ/NCQ
	Gradient accumulation per update	4 per device at 96 prompts, up to 6 at 144
	Total rollout steps per run	200
	Optimizer mini-batches per run	800 by default ($I \cdot 200$ per corpus pass)
	Padding-free training	on

The binary reward. We grade short verifiable answers with the math-aware grader of `mathruler.grader`. `extract_boxed_content` reads the last `\boxed{...}` span, and `grade_answer` compares it with the gold answer $a^\star$. The grader counts LaTeX and numerical equivalents, such as `1/2` and `0.5`, as matches. If that comparison fails and a boxed span is present, a second pass strips a degree or percent sign that both the prediction and the gold answer carry, then grades once more. The reward is 1 when either pass succeeds.

Free-form answers go to the shared LLM-as-a-judge (Zheng et al., 2023), in the same configuration as at evaluation (Appendix C.2). Both paths compare against $a^\star$. No judge token is ever a gradient target. The judge enters training only through the binary reward it assigns. Every method in every table uses this routing.

Candidate compression. Rollout scoring selects the candidates first. The frozen teacher then rewrites each selected response under one shared compression prompt and a 512-token output cap (Fig. 16). Prompt and cap are the same for teacher and student candidates. The compressor reads only the source response. The prompt tells it not to re-derive the problem and to end with the source's final answer. Its output enters the prompt as context, never as a gradient target. Each compressed candidate keeps the reward of its source rollout. The words "correct" and "wrong" in the BCQ instruction refer to that reward, and the compressed text is not graded again. Appendix G.8 reports how often a new rollout's final answer matches each candidate.

Compress the response below into a summary (in 5 lines max).
Rules:
- *Response is in* `<response>...</response>` *tags*
- *Summary should be in* `<summary>...</summary>` *tags and should be in 5 lines max*
- *Keep ONLY the essential reasoning steps and the final answer*
- *Remove ALL exploratory text, self-corrections, retries, and filler*
- *Do NOT re-derive or add new information*
- *End with the final answer in* `\boxed{}` *format*

`<response>`
⟨candidate response to be compressed – $y_{\mathrm{T}}^{(+)}$ *or* $y_{\mathrm{S}}^{(-)}$ *for BCQ,* $y_{\mathrm{S}}^{(-)}$ *for NCQ*⟩
`</response>`

Figure 16: Candidate compression prompt. The frozen teacher applies it, with a 512-token cap, to every BCQ and NCQ candidate before the candidate enters a prompt.

Reformulated prompts (BCQ and NCQ). Fig. 17 and Fig. 18 give the executed blocks that BCQ and NCQ append. We write the resulting prompts as x_{BCQ} and x_{NCQ}. A `<candidate>` block holds the returned summary text without the outer `<summary>` wrapper. In BCQ and NCQ alike, a wrong rollout whose final `\boxed{}` span does not parse never becomes a candidate. BCQ shuffles its two blocks, and no label, letter, or fixed order marks which one the teacher wrote. The instruction asks the student to use the two candidates as references. It does not ask the student to name one. The binary reward grades only the student's own final answer against $a^\star$. Appendix G.8 reports how often that answer matches one of the two candidate answers. Appendix I.4 reports which part of the block the benchmark gain needs.

BCQ (Fig. 17)**.** For each hard question with at least one correct teacher rollout, we draw one $y_{\mathrm{T}}^{(+)}$ and one $y_{\mathrm{S}}^{(-)}$ uniformly, teacher-compress both, shuffle their order, and append:

NCQ (Fig. 18)**.** For each hard question, we collect the wrong rollouts of the current group, parse each one's final boxed answer, and teacher-compress every rollout whose answer parses. Let $K \le G_{\mathrm{S}}$ be the number of parsed wrong rollouts and $M \le K$ the number of distinct parsed answers. The prompt lists each of the M answers once, joined with "`,  `", and keeps every compressed trace, so it holds K `<candidate>` blocks.

Both templates then receive the instruction of Fig. 8, appended once and last. The final order is the question text and image, the appended block, and that instruction. The student samples $y_{\mathrm{BCQ}}, y_{\mathrm{NCQ}} \sim \pi_\theta(\cdot|x_{\mathrm{BCQ/NCQ}})$ in the same think-then-boxed format as every other rollout. Dataset

⟨*original question* x⟩
Here are two candidate responses in `<candidate> </candidate>` *tags to the question above. One is correct and another is wrong. Use these as references to help you solve the problem.*
`<candidate>`
⟨*candidate*$_1$ *– compressed* $y_{\mathrm{T}}^{(+)}$ *or* $y_{\mathrm{S}}^{(-)}$*, shuffled*⟩
`</candidate>`
`<candidate>`
⟨*candidate*$_2$ *– the other one*⟩
`</candidate>`

Figure 17: BCQ reformulated-prompt template. Sec. 3.2 quotes its instruction in full. The two shuffled blocks hold one compressed correct teacher rollout and one compressed wrong student rollout.

⟨*original question* x⟩
The following answers are all WRONG: $\boxed{a_1}$, $\boxed{a_2}$, …, $\boxed{a_M}$. *Below are the incorrect reasoning processes in* `<candidate> </candidate>` *tags.*
`<candidate>`
⟨*compressed wrong rollout* 1⟩
`</candidate>`
`<candidate>`
⟨*compressed wrong rollout* 2⟩
`</candidate>`
…
`<candidate>`
⟨*compressed wrong rollout* K⟩
`</candidate>`

Figure 18: NCQ reformulated-prompt template. The prompt lists M distinct wrong answers and K candidate blocks.

filtering keeps every original prompt within $4{,}096$ tokens (Appendix C). The compression cap keeps each candidate within 512. Candidate text is therefore at most $1{,}024$ tokens for BCQ (two blocks) and at most $4{,}096$ for NCQ, whose K blocks reach $G_{\mathrm{S}} = 8$ on an all-wrong group. Reformulated prompts stay within the model's 262K context window, so no part of the appended block is truncated.

E.2 Baseline implementations

We give the executed step of every baseline and control. Each control takes named elements of its cited methods and runs them on our shared pipeline (Appendix B). The cited methods also have their own gates, gradient corrections, reward shaping, schedules, search procedures, or auxiliary models, with thresholds tuned to their own models and datasets. We keep every such setting at the shared value, so the tables compare *where* and in what form teacher knowledge enters training. Each control paragraph below lists what the control takes from each cited method and what it leaves out. Off-Distill, On-Distill, GRPO, and their † rows are the baselines. Hint, Prefix, Correction-SFT, and Error-reflection-SFT are the controls.

All rows share the initialization, corpus, rollout sizes, constant learning rate, and 200 rollout steps of Tab. 11. Every row therefore draws the same $200 \times 384 = 76{,}800$ new prompts and the same G_{S} student rollouts on each. The *Optimizer partitions* paragraph below states how each row packs them into optimizer updates. All rows also share the evaluation setup of Appendix C.2, except that Correction-SFT† keeps its training-time `<correct>` tag at evaluation.

The ablation rows run Algorithm 1 with branches switched off and nothing else changed. GRPO†+BCQ skips the NCQ branch of Step 4, and GRPO†+NCQ skips the BCQ branch. GRPO + Both keeps both branches at $\rho_{\mathrm{replay}} = 0$, so its hard questions come from the current data-loader batch alone. Caps, thresholds, and the advantage estimator keep the ZPPO values of Tab. 11.

Replay, admission, and graduation. Every † row draws its replay prompts uniformly and without replacement *before* the current step refreshes the buffer. The replay count is $\min(\rho_{\text{replay}}|X_{\text{new}}|, |\mathcal{B}|)$, so a buffer with fewer entries than the full replay count contributes all of them. The refresh reads $\bar{r}_x$ from the plain rollouts alone, never from a BCQ, NCQ, hint, or prefix group. Questions with $\bar{r}_x < \tau$ are admitted or stay in the buffer. Questions with $\bar{r}_x \geq \tau$ graduate. The oldest entries are then FIFO-evicted until $|\mathcal{B}| \leq |\mathcal{B}|_{\max}$. The buffer stores questions only, never a response or a reformulated prompt. Every algorithm below uses this replay rule and this buffer update. In each, $\text{concat}(X_{\text{new}}, X_{\text{replay}})$ joins the two prompt lists.

Hard questions used by the controls. Hint, Prefix, Correction-SFT, and Error-reflection-SFT generate their teacher output online, on the step that uses it. Each selects at most the $\rho_{\text{aug}}|X_{\text{new}}| = 96$ hardest questions with $\bar{r}_x < \tau$, ranked by ascending $\bar{r}_x$. A step with fewer hard questions selects all of them. A question with no usable teacher output adds no extra group or target, but its plain rollouts still update the buffer. The four rows are matched on teacher output: each uses it on at most 96 questions per step. Hint, Prefix, and Error-reflection-SFT build it from $G_{\text{T}} = 4$ teacher rollouts per question, as Step 4 of Algorithm 1 does. Correction-SFT makes one corrector call per question. In ZPPO, Step 3 draws the teacher rollouts on all of X in parallel with Step 1, and Step 4 uses only those on $X_{\text{aug}}^{\text{pre}}$ (Appendix D).

Gradient targets per step. An RL row backpropagates through at most 576 groups of $G_{\text{S}} = 8$ student responses per rollout step: 384 new prompts, up to 96 replayed prompts, and at most 96 BCQ/NCQ, hint, or prefix groups. Fewer groups enter when the buffer holds less than the full replay count or when the added groups do not fill their cap. Every gradient-counted token of an RL row is sampled from the current student. Off-Distill instead imitates the correct ones among the $G_{\text{T}} = 4$ teacher rollouts drawn on each prompt of its batch. On-Distill imitates the teacher's per-token distributions on all G_{S} student rollouts of each prompt. Correction-SFT and Error-reflection-SFT backpropagate through the verified targets of Algorithms 7 and 8.

Off-policy distillation. Off-Distill draws $G_{\text{T}} = 4$ fresh teacher rollouts on each prompt of its batch and grades them with the shared reward. It imitates the correct ones under the per-token generalized JSD of GKD (Agarwal et al., 2024) at $\beta = 0.5$. This loss compares the teacher and the student distributions with their mixture, which is β times the teacher plus $1-\beta$ times the student. The KL divergence from the teacher to the mixture gets weight β, and the one from the student to the mixture gets weight $1-\beta$. At $\beta = 0.5$ the mixture is the plain average, and the loss is symmetric. GKD itself has no correctness filter, so keeping only the correct teacher rollouts is our choice. A prompt with no correct teacher rollout adds no imitation target. We still draw and score $G_{\text{S}} = 8$ student rollouts on every prompt, as every row does. The † row reads their scores for its buffer update. No teacher rollout is cached, so the † row samples the teacher again on every replay visit.

On-policy distillation. The student samples $G_{\text{S}} = 8$ rollouts. The frozen teacher re-scores each one token by token. At every position it reads the student's partial response and gives a target distribution. This is the on-policy case of GKD (Agarwal et al., 2024), in which the student writes every training sequence. The loss is the same per-token generalized JSD at $\beta = 0.5$ as in Off-Distill, so the two distillation rows differ only in where the targets come from. Off-Distill imitates fresh teacher rollouts filtered by correctness, and On-Distill imitates the teacher's re-scoring of the student's own rollouts. The † row adds prompt replay only. Both distillation rows use the shared learning rate of 10^{-6} (Tab. 11). At a tenfold larger rate (10^{-5}), their VLM gain stayed about the same, but their LLM and Video averages fell much further. This is consistent with stronger overfitting to the 77K-example corpus, so we report 10^{-6}. More updates do not help On-Distill† either. A second corpus pass doubles its updates from 800 to 1,600 and still lowers every macro average at both scales (Appendix I.1).

GRPO and GRPO†. GRPO is ZPPO's pipeline without BCQ, NCQ, and the prompt replay buffer (Sec. 4). GRPO† adds the buffer alone, at $\rho_{\text{replay}} = 0.25$. Both use ZPPO's two-step estimator with zero-advantage groups excluded from the batch statistics, the token-level clipped surrogate, and $I = 4$.

Optimizer partitions and the budgets that move with I. Iterations per step I is the number of optimizer mini-batches that one rollout batch is split into. Every rollout enters exactly one mini-batch,

Algorithm 2: Executed Off-Distill step. The student imitates online teacher rollouts filtered by correctness. The † row adds prompt replay.

Input: Student π_θ, frozen teacher π_{T}, dataset $\mathcal{D}$, buffer $\mathcal{B}$ († only); $G_{\mathrm{T}}=4$, G_{S}, $\beta=0.5$, I; $\rho_{\mathrm{replay}}, |\mathcal{B}|_{\max}, \tau$ († only).

Sample $X_{\mathrm{new}} \sim \mathcal{D}$. For †, draw X_{replay} from $\mathcal{B}$ by the replay rule; otherwise set $X_{\mathrm{replay}}=\emptyset$. Set $X \leftarrow \mathrm{concat}(X_{\mathrm{new}}, X_{\mathrm{replay}})$.

foreach $x \in X$ **do**

  Draw G_{S} student rollouts $\{y_{\mathrm{S}}^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot|x)$ and score them; the † row keeps their mean $\bar{r}_x$.

  Draw G_{T} fresh teacher rollouts $\{y_{\mathrm{T}}^{(g)}\}_{g=1}^{G_{\mathrm{T}}} \sim \pi_{\mathrm{T}}(\cdot|x)$, grade them, and keep the N_x correct ones.

  if $N_x > 0$ **then**

    Accumulate the average per-token generalized JSD over the N_x correct teacher rollouts.

  end

end

Split X into I optimizer mini-batches of 96 new prompts each, as in the RL rows. Apply one AdamW update per mini-batch on its JSD, and skip a mini-batch with no correct teacher rollout.

(† only) Update $\mathcal{B}$ from $\bar{r}_x$: admit, graduate, and FIFO-evict.

return π_θ *(and $\mathcal{B}$ for †).*

Algorithm 3: Executed On-Distill step. The student imitates the teacher's re-scoring of its own rollouts. The † row adds prompt replay.

Input: Student π_θ, frozen teacher π_{T}, dataset $\mathcal{D}$, buffer $\mathcal{B}$ († only); G_{S}, $\beta=0.5$, I; $\rho_{\mathrm{replay}}, |\mathcal{B}|_{\max}, \tau$ († only).

Sample $X_{\mathrm{new}} \sim \mathcal{D}$. For †, draw X_{replay} from $\mathcal{B}$ by the replay rule; otherwise set $X_{\mathrm{replay}}=\emptyset$. Set $X \leftarrow \mathrm{concat}(X_{\mathrm{new}}, X_{\mathrm{replay}})$.

foreach $x \in X$ **do**

  Draw G_{S} student rollouts $\{y_{\mathrm{S}}^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot|x)$.

  Forward them through π_{T}, and accumulate the per-token generalized JSD between π_{T} and π_θ at every position of those student rollouts.

  († only) Score the group with the binary reward and keep its mean $\bar{r}_x$.

end

Split X into I optimizer mini-batches of 96 new prompts each, as in the RL rows, and apply one AdamW update per mini-batch on its JSD.

(† only) Update $\mathcal{B}$ from $\bar{r}_x$: admit, graduate, and FIFO-evict.

return π_θ *(and $\mathcal{B}$ for †).*

Algorithm 4: Executed GRPO step. It has no BCQ or NCQ branch. The † row adds prompt replay.

Input: Student π_θ, dataset $\mathcal{D}$, buffer $\mathcal{B}$ († only); G_{S}, I; $\rho_{\mathrm{replay}}, |\mathcal{B}|_{\max}, \tau$ († only).

Sample $X_{\mathrm{new}} \sim \mathcal{D}$. For †, draw X_{replay} from $\mathcal{B}$ by the replay rule; otherwise set $X_{\mathrm{replay}}=\emptyset$. Set $X \leftarrow \mathrm{concat}(X_{\mathrm{new}}, X_{\mathrm{replay}})$.

foreach $x \in X$ **do**

  Draw G_{S} student rollouts $\{y_{\mathrm{S}}^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot|x)$, score them, and compute $\bar{r}_x$.

end

Compute Eq. (2) within each complete rollout group, and split the groups across I optimizer mini-batches. In each mini-batch, apply Eq. (3) to the non-trivial groups, keep trivial-group advantages at zero, and apply the token-level clipped surrogate of Eq. (8). Skip a mini-batch whose advantages are all zero.

(† only) Update $\mathcal{B}$ from $\bar{r}_x$: admit, graduate, and FIFO-evict.

return π_θ *(and $\mathcal{B}$ for †).*

and none is used twice. The PPO inner-epoch count is 1 everywhere, so a run has $I \cdot 200$ optimizer mini-batches, with one update each. The default $I = 4$ splits each of the 200 rollout steps into four mini-batches of 96 new prompts. At the default I and one corpus pass, every row—RL, distillation, and supervised—takes 800 mini-batches per run, as in Tab. 11. A mini-batch with no usable target is skipped: an RL mini-batch with no non-trivial group, a distillation mini-batch with no imitation target, or a supervised mini-batch with no verified target. For any recipe, I is the ratio of the rollout batch to the optimizer mini-batch. DAPO's batch and mini-batch sizes give $I = 16$ (Yu et al., 2025). This is the $I = 16$ point of the sweep in Fig. 6a–c and the setting that Sec. 4.4 compares against. The new-prompt rollout batch is I times the per-update global batch of 96, so the sweep moves prompt draws together with the update count. At 200 steps the new-prompt draws are 19,200, 76,800, 153,600, and 307,200 at $I \in \{1, 4, 8, 16\}$, so $I = 8$ and $I = 16$ pass over the corpus twice and four times. Appendix G.5 reports a GRPO† pair at $I = 4$ and $I = 8$ with matched prompt draws and update count. Appendix I.1 runs a second corpus pass on On-Distill†, the one distillation row in that sweep. The pass takes the row from 800 to 1,600 mini-batches, the same packing as the RL rows. It does not raise the row's average on any family at either scale.

Hint control (Tab. 4). Hint gives an answer-free cue abstracted from a solution (Nath et al., 2025; Chen et al., 2026a; Wang et al., 2026; Xia et al., 2026; Wu et al., 2025). In our control, that solution is one graded-correct teacher rollout. We run the control on GRPO† (Sec. 4.5), with the cue in the prompt and never in the gradient target.

Guide-GRPO (Nath et al., 2025) uses hints that GPT-4o writes from the question and a reference solution. When all plain rollouts on a question fail, it samples an equal number of extra rollouts with the hint appended to the prompt. It pools the hinted and plain rollouts in one advantage computation. Each hinted token carries an importance weight whose numerator drops the hint, so the update optimizes the plain prompt. Guide-GRPO also reshapes this weight. To keep this training stable, it removes PPO ratio clipping. From Guide-GRPO we take extra hinted rollouts on a hard question, with the hint in the prompt and the whole response written by the student. We do not take its hint writer, its all-wrong gate, its pooled advantage, its importance weight and the reshaping of that weight, or its removal of clipping. Our gate is $\bar{r}_x < \tau$. Our hinted rollouts form their own group, which trains under the prompt that sampled it, with the two-step estimator and the clipped surrogate.

NuRL (Chen et al., 2026a) writes its hints offline, before training, with the policy itself. The initial policy reads the question and the gold answer and writes a chain of thought that explains why that answer is correct. It then abstracts this explanation into a hint that does not reveal the answer. A variant uses o4-mini as the hint writer. NuRL first trains with plain GRPO and turns hints on only after that stage converges. When all G plain rollouts on a question fail, it appends the hint after the question and samples a new batch in which $G-1$ rollouts see the hint and one stays hint-free. From NuRL we take the answer-free hint placed after the question and used in training only. We do not take its self-written hints, its use of the gold answer, its offline hint writing, its two-stage schedule, its all-wrong gate, or its hint-free rollout.

HINT (Wang et al., 2026) calls its cue a Meta-Hint: one short sentence that names the key insight without revealing the answer. Qwen2.5-72B-Instruct writes these hints for the whole training set before training, from the question and its final answer. The hint sits before the question in the prompt. HINT resamples a new group under the hint only when the first group fails completely. It trains that group with Affinity-Aware Policy Optimization, which scales the group's objective by an Affinity score. The score grows with the share of unclipped token updates and falls with their variability. HINT also computes the gradient against the original question rather than the hinted one. From HINT we take a short, answer-free cue in the prompt. We do not take its one-sentence limit, its pre-written hints, their place before the question, its all-wrong gate, its Affinity weighting, or its gradient on the original question. Our generator prompt (Fig. 19) asks for a concise hint and allows the key intermediate idea.

HiLL (Xia et al., 2026) writes hints online with a hinter policy that it trains jointly with the reasoner. It acts only on all-wrong groups. For each such group, the hinter reads the question, one wrong rollout from that group, and a reference solution. It then proposes several hints. Each hint goes after the question, and the reasoner samples a group under it. The hinter's reward favors hints that give mixed outcomes and whose correct rollouts depend little on the hint. When the best hint earns a positive reward, its group replaces the all-wrong group. The reasoner trains on that group under the

hinted prompt. From HiLL we take online hint writing from a correct solution and the training of the hinted group under the prompt that sampled it. We do not take its all-wrong gate, its trained hinter, its reward, its search over several hints, the hinter's reading of a wrong rollout, or the replacement of the plain group. Our generator is the frozen 27B teacher and reads one graded-correct teacher rollout only.

TAPO (Wu et al., 2025) guides every training question with high-level templates from a thought library. Each template is a sequence of generic reasoning actions, such as divide and conquer or self-reflection. The templates are abstracted before training from MCTS solutions of 500 seed problems, not from a solution of the question being trained. For each question, TAPO retrieves the templates closest to it in problem-condition complexity. Their action sequences guide the policy's step-by-step reasoning. TAPO samples rollouts under each retrieved template and treats all of them as one group, with no plain rollout. From TAPO we take high-level guidance instead of a reference solution. We do not take its thought library, its retrieval, its generic templates, its guidance on every question, or its single group across templates. Our cue is written for the question at hand, from one correct teacher rollout.

SocraticPO (Liu et al., 2026c) places teacher guidance in the student's context during RL. After each wrong answer, a teacher model reads the question and that answer, diagnoses the error, and writes short corrective guidance. The student then writes a new full answer, with its failed answer and the guidance in its context. A correct answer reached after guidance receives a decayed reward. From SocraticPO we take guidance that a teacher writes online and places in the student's context. As in SocraticPO, the student writes every gradient-counted token, and training runs under the guided context. We do not take its guidance after every wrong answer, its guidance conditioned on that answer, its retry with the failed answer in context, or its decayed reward. Our cue is written from a correct teacher rollout. The hinted group is sampled afresh and keeps the full binary reward.

Keeping these settings shared lets the Hint row isolate the effect of an answer-free cue in the prompt.

In the executed control, the generator is the frozen 27B teacher. On each of at most the 96 hardest questions with $\bar{r}_x < \tau$, the teacher draws $G_{\mathrm{T}} = 4$ fresh rollouts and grades them as ZPPO grades its BCQ pool. The generator reads one uniformly chosen graded-correct rollout and nothing else (Fig. 19). Like the candidate compressor, it sees neither the question nor the gold answer. Its prompt tells it not to reveal the final answer and not to include a `\boxed{}` span. A question with no correct teacher rollout gets no cue. The cue goes inside `<hint>`, after the question and before the instruction of Fig. 8. The student samples G_{S} rollouts on that prompt as a new group under the full binary reward. The plain group stays in the same rollout step. Each group trains under the prompt that sampled it, with the two-step estimator and the clipped surrogate. Every gradient-counted token is sampled from the current student, and the buffer is updated as in ZPPO. The cue appears in training only, and evaluation has no `<hint>` block. Algorithm 5 is the executed step.

Read the solved response below and write a concise hint that helps solve the original question. Do NOT reveal the final answer, do NOT include any `\boxed{}` *expression, and do NOT copy the full solution. Keep only high-level guidance or the key intermediate idea.*
<response>
$y_{\mathrm{T}}^{(+)}$
</response>

Figure 19: Hint-generation prompt for Algorithm 5. The generator reads only the solved response.

Prefix control (Tab. 4). Prefix gives the opening portion of the solution itself, and the student continues from it (Zhang et al., 2025d;b; Li et al., 2025d; Huang et al., 2025b;a). In our control, that solution is one graded-correct teacher rollout, and the prefix is its first 40% of tokens. We run the control on GRPO† (Sec. 4.5) and mask every prefix token from the loss.

Hint-GRPO (Huang et al., 2025a) takes its prefix from the solutions in its dataset, which GPT-4o splits into steps. It keeps the question unchanged, puts the prefix at the start of the response, and lets the model complete the remaining steps. Its adaptive hint length samples M groups with hint ratios $(i-1)/M$ for $i = 1, \ldots, M$, and it trains only the first group that has a correct answer. The first group has no hint. A hint is therefore used only when the plain group fails, and the hinted group then takes

Algorithm 5: Executed Hint-control step. The cue enters training prompts only.

Input: Student π_θ, frozen teacher π_{T}, dataset $\mathcal{D}$, buffer $\mathcal{B}$; $G_{\mathrm{T}}=4$, G_{S}, I, $\rho_{\mathrm{replay}}, \rho_{\mathrm{aug}}, |\mathcal{B}|_{\max}, \tau$.

Sample $X_{\mathrm{new}} \sim \mathcal{D}$, draw X_{replay} from $\mathcal{B}$ by the replay rule, and set $X \leftarrow \mathrm{concat}(X_{\mathrm{new}}, X_{\mathrm{replay}})$.
foreach $x \in X$ **do**
  Draw G_{S} student rollouts $\{y_{\mathrm{S}}^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot|x)$, score them, and compute $\bar{r}_x$.
end
Select X_{aug}: at most the $\rho_{\mathrm{aug}}|X_{\mathrm{new}}|$ hardest questions with $\bar{r}_x < \tau$, ranked by ascending $\bar{r}_x$.
foreach $x \in X_{\mathrm{aug}}$ **do**
  Draw G_{T} fresh teacher rollouts $\{y_{\mathrm{T}}^{(g)}\}_{g=1}^{G_{\mathrm{T}}} \sim \pi_{\mathrm{T}}(\cdot|x)$, grade them, and keep the N_x correct ones.
  if $N_x > 0$ **then**
    Choose one correct rollout $y_{\mathrm{T}}^{(+)}$ uniformly and query π_{T} with Fig. 19 on $y_{\mathrm{T}}^{(+)}$ alone to obtain the cue h.
    Form x_{Hint} by inserting `<hint>` h `</hint>` after the question and before the instruction of Fig. 8.
    Draw G_{S} student rollouts $\{y_{\mathrm{Hint}}^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot|x_{\mathrm{Hint}})$ and score them as a new group.
  end
end
Compute Eq. (2) within each complete plain and hint group, and split the groups across I optimizer mini-batches. In each mini-batch, apply Eq. (3) to the non-trivial groups, keep trivial-group advantages at zero, and apply the token-level clipped surrogate of Eq. (8), each group under the prompt that sampled it. Skip a mini-batch whose advantages are all zero.
Update $\mathcal{B}$ from the plain-rollout $\bar{r}_x$: admit, graduate, and FIFO-evict.
return π_θ, $\mathcal{B}$.

its place. At test time, Hint-GRPO shifts the logits by the gap between predictions with and without the image. From Hint-GRPO we take the place of the prefix: the question stays unchanged, and the prefix opens the response as a condition. We do not take its step split, its adaptive hint length, its single trained group, or its test-time shift.

BREAD (Zhang et al., 2025d) stores an expert trace with each question of its dataset. When the student's own group has no correct rollout, BREAD branches new student rollouts from a prefix of that trace. It splits the trace into episodes and binary-searches the prefix length. It shortens the prefix when every new rollout is correct, lengthens it when every new rollout is wrong, and stops when the group's success rate falls in a preset range. The prefix follows the question in the prompt, and the branched group replaces the failed one. Its loss covers the student's tokens only. From BREAD we take an expert prefix on a hard question, student rollouts that branch from it, and a loss on the student's tokens only. We do not take its stored traces, its length search, its all-wrong gate, its prefix in the prompt, or its replacement of the plain group. Our prefix length is fixed, and our prefix opens the response.

StepHint (Zhang et al., 2025b) collects verified-correct chains from stronger models before training. It splits each chain into steps using the model's end-of-thinking probability. Every training problem gets several hint levels, each a different number of opening steps. It also gets unhinted rollouts and the reference chain as an extra completion. The hint opens the response, and its tokens stay in the loss. On a wrong completion, the negative advantage on the hint tokens is clipped to zero. From StepHint we take the opening of a stronger model's correct chain as a prefix that opens the response, with plain rollouts in the same step. We do not take its offline chains, its step split, its several hint levels per problem, its hints on every problem, the reference chain as a completion, or the loss on prefix tokens. We mask every prefix token instead.

SEELE (Li et al., 2025d) appends a prefix of an annotated solution after every training problem. DeepSeek-V3 writes these solutions before training. SEELE samples several rounds of rollouts, all under the adaptive prefix, so it has no plain group and no gate. Before each round, it fits an item-response-theory model to pick the prefix length per instance, so that rollout accuracy stays near

50%. It computes advantages over the rollouts of all rounds. Its RL loss covers the generated tokens only, and a small supervised loss covers the prefix tokens. From SEELE we take part of a solution to lower a question's difficulty, with no policy-gradient term on the prefix. We do not take its pre-written solutions, its prefix on every rollout of every problem, its length search, its advantage pooled across rounds, or its supervised loss on the prefix. Our length is fixed.

Prefix-RFT (Huang et al., 2025b) cuts an offline DeepSeek-R1 demonstration at a fraction of its length. The fraction is drawn from a range whose lower end decays under a cosine schedule. The policy continues from the cut. On every prompt, this hybrid rollout replaces one on-policy rollout of the group. The prefix tokens stay in the PPO-style loss with the hybrid rollout's advantage, and only the highest-entropy ones are updated. From Prefix-RFT we take the cut at a fraction of the length, which we fix at $\alpha = 0.4$, and the prefix at the start of the response. We do not take its offline demonstrations, its schedule, its hybrid rollout on every prompt, its replacement of an on-policy rollout, or its loss on the prefix. We add a new group instead. We mask every prefix token, so the prefix receives no policy-gradient term and no supervised term.

LUFFY (Yan et al., 2025) guides RL with offline traces from a stronger model. The traces are correct DeepSeek-R1 outputs. On every question, one whole trace joins the on-policy rollouts of a GRPO group, and the advantage is computed over the whole group. The trace tokens stay in the loss. LUFFY reshapes their importance ratio with what it calls regularized importance sampling, and it removes the PPO clip. From LUFFY we take a correct trace from a stronger model as the source of guidance, used in the same rollout step as the student's own rollouts. We do not take its offline traces, the whole trace in the group, the gradient on teacher tokens, the reshaped ratio, or the removed clip. Our prefix is the opening part of one teacher rollout, and every prefix token is masked. ZPPO keeps teacher tokens out of the gradient for the reason given in Appendix D. The executed rows that put the teacher in the gradient are the two distillation rows, ReLIFT† and RSTG† (Appendix I.3), and KDRL†, RLKD†, and RLAD† (Appendix I.6).

In the executed control, the frozen 27B teacher draws $G_{\mathrm{T}} = 4$ rollouts on each of at most the 96 hardest questions with $\bar{r}_x < \tau$. We choose one graded-correct rollout $y_{\mathrm{T}}^{(+)} = (t_1, \ldots, t_L)$ uniformly and do not compress it. A question with no correct teacher rollout gets no prefix. The prefix $p = (t_1, \ldots, t_{\lfloor \alpha L \rfloor})$ with $\alpha = 0.4$ opens the response, and the question stays unchanged. The student samples G_{S} continuations from this shared prefix as a new group, and the plain group stays in the same rollout step. The reward grades the prefix and the continuation together. Prefix and continuation share the maximum response length of Tab. 11. Every prefix token is masked from the loss, so every gradient-counted token is sampled from the current student. The update uses the two-step estimator and applies the clipped surrogate to the unmasked tokens. The buffer is updated as in ZPPO. The prefix appears in training only. The row is evaluated with no prefix, as every ZPPO row is evaluated with no candidate block (Appendix I.10). Algorithm 6 is the executed step.

Correction-SFT control (Tab. 5). Correction-SFT follows LeMa and NAT, two supervised methods of the negative-example family (Sec. 4.5). From LeMa (An et al., 2023) we take the corrector protocol and its filter. The corrector names the wrong step, explains why it is wrong, and rewrites the solution from that step onward. A correction with a wrong final answer is discarded. From NAT (Wang et al., 2024b) we take the conditioning. A tag in the input marks the response as correct or incorrect, and both kinds of response train under the same cross-entropy. In NAT the tag is a short prompt that asks for a correct or an incorrect solution, and plain markers such as letters work about as well.

The corrector is the frozen 27B teacher. For each selected hard question, it reads the question and one uniformly sampled wrong student rollout (Fig. 20). A graded-correct correction becomes the SFT target. Each verified correction gives two training instances at equal count: the correction under `<correct>` and the sampled wrong response under `<incorrect>`. The SFT input is the question, the tag, and the instruction of Fig. 8. Prompt and tag tokens are masked, so cross-entropy covers response tokens only. Neither this prompt nor the refinement prompt of Fig. 21 asks for an output format beyond a final `\boxed{}` span. The teacher writes both targets in the same think-then-boxed format as its own rollouts.

As in NAT, evaluation uses the positive tag. Correction-SFT† keeps `<correct>`, the condition it trains under. It is the one row that departs from the shared evaluation prompt of Appendix C.2. In LeMa, the fine-tuning input is the question and the wrong solution, and the target is the correction. Our input follows NAT instead and holds only the question and the tag. The corrector writes from

Algorithm 6: Executed Prefix-control step. The prefix opens the response, and every prefix token is masked from the loss.

Input: Student π_θ, frozen teacher π_{T}, dataset $\mathcal{D}$, buffer $\mathcal{B}$; $G_{\mathrm{T}}=4$, G_{S}, $\alpha=0.4$, I, $\rho_{\mathrm{replay}}, \rho_{\mathrm{aug}}, |\mathcal{B}|_{\max}, \tau$.

Sample $X_{\mathrm{new}} \sim \mathcal{D}$, draw X_{replay} from $\mathcal{B}$ by the replay rule, and set $X \leftarrow \mathrm{concat}(X_{\mathrm{new}}, X_{\mathrm{replay}})$.
foreach $x \in X$ **do**
  Draw G_{S} student rollouts $\{y_{\mathrm{S}}^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot|x)$, score them, and compute $\bar{r}_x$.
end
Select X_{aug}: at most the $\rho_{\mathrm{aug}}|X_{\mathrm{new}}|$ hardest questions with $\bar{r}_x < \tau$, ranked by ascending $\bar{r}_x$.
foreach $x \in X_{\mathrm{aug}}$ **do**
  Draw G_{T} fresh teacher rollouts $\{y_{\mathrm{T}}^{(g)}\}_{g=1}^{G_{\mathrm{T}}} \sim \pi_{\mathrm{T}}(\cdot|x)$, grade them, and keep the N_x correct ones.
  if $N_x > 0$ **then**
    Choose one correct rollout $y_{\mathrm{T}}^{(+)} = (t_1, \ldots, t_L)$ uniformly and set the response prefix $p = (t_1, \ldots, t_{\lfloor \alpha L \rfloor})$.
    Draw G_{S} student continuations $\{c^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot|x, p)$ from the shared prefix.
    Score the responses $\{p \,\|\, c^{(g)}\}$ as a new group, and mask every token of p from the loss (no policy-gradient or supervised term on p).
  end
end
Compute Eq. (2) within each complete plain and prefix group, and split the groups across I optimizer mini-batches. In each mini-batch, apply Eq. (3) to the non-trivial groups, keep trivial-group advantages at zero, and apply the token-level clipped surrogate of Eq. (8) to the unmasked response tokens. Skip a mini-batch whose advantages are all zero.
Update $\mathcal{B}$ from the plain-rollout $\bar{r}_x$: admit, graduate, and FIFO-evict.
return π_θ, $\mathcal{B}$.

the first wrong step onward and does not restate the attempt, so a correction can refer to an attempt the student does not see. Error-reflection-SFT below has no such gap. Its input is the bare question, and its target is a complete, verified teacher solution revised with the student's own errors. This self-contained, error-informed target is what the Correction-SFT row lacks. Both rows land at or below the untrained base on LLM and Video (Tab. 5), so the comparison with NCQ does not rest on the Correction-SFT row.

Below is a question and a student's incorrect attempt. Identify the first step that is wrong, state briefly why it is wrong, then write a complete corrected solution from that point onward. Do not restate the incorrect attempt. End with the final answer in `\boxed{}` *format.*
`<question>`
x
`</question>`
`<incorrect_attempt>`
$y_{\mathrm{S}}^{(-)}$
`</incorrect_attempt>`

Figure 20: Correction-generation prompt for Algorithm 7. The frozen teacher reads the question and one wrong student rollout.

Error-reflection-SFT control (Tab. 5). Error-reflection-SFT follows ORION (Wu et al., 2026b) with teacher-side refinement (Sec. 4.5). In ORION, the student refines the teacher's chain of thought using its own wrong attempts. ORION refines once per wrong attempt, and each refinement opens with an analysis of that attempt's error. Its fine-tuning prompt also shows the wrong attempt. From ORION we take error-aware refinement and its filter: a teacher solution is rewritten using the student's own reasoning errors, a refinement with a wrong final answer is discarded, and the student is fine-tuned on the refined solution. Our control differs from ORION in four ways. The frozen teacher

Algorithm 7: Executed Correction-SFT step. The frozen teacher is the corrector, and its correction is graded before it becomes a target.

Input: Student π_θ, frozen teacher π_{T}, dataset $\mathcal{D}$, buffer $\mathcal{B}$; G_{S}, I, $\rho_{\mathrm{replay}}, \rho_{\mathrm{aug}}, |\mathcal{B}|_{\max}, \tau$.

Sample $X_{\mathrm{new}} \sim \mathcal{D}$, draw X_{replay} from $\mathcal{B}$ by the replay rule, and set $X \leftarrow \mathrm{concat}(X_{\mathrm{new}}, X_{\mathrm{replay}})$.
foreach $x \in X$ **do**
  Draw G_{S} student rollouts $\{y_{\mathrm{S}}^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot|x)$, score them, and compute $\bar{r}_x$.
end
Select X_{aug}: at most the $\rho_{\mathrm{aug}}|X_{\mathrm{new}}|$ hardest questions with $\bar{r}_x < \tau$, ranked by ascending $\bar{r}_x$.
foreach $x \in X_{\mathrm{aug}}$ **do**
  Choose one wrong rollout $y_{\mathrm{S}}^{(-)}$ uniformly, query π_{T} with Fig. 20 to obtain the correction y_{corr}, and grade it.
  if y_{corr} *is correct* **then**
    Form $(x \,\|\, \texttt{<correct>},\, y_{\mathrm{corr}})$ and $(x \,\|\, \texttt{<incorrect>},\, y_{\mathrm{S}}^{(-)})$, with each tag placed before the instruction of Fig. 8.
    Accumulate cross-entropy on both responses at equal instance count, with prompt and tag tokens masked.
  end
end
Split X into I optimizer mini-batches of 96 new prompts each, as in the RL rows. Apply one AdamW update per mini-batch on its cross-entropy, and skip a mini-batch with no verified pair.
Update $\mathcal{B}$ from the plain-rollout $\bar{r}_x$: admit, graduate, and FIFO-evict.
return π_θ, $\mathcal{B}$.

refines instead of the student. We refine once per question, using all wrong attempts at once. Our refinement holds only the revised solution. Our SFT input is the bare question.

The refiner is the frozen 27B teacher, the same model behind every teacher-assisted row of Tab. 5. On each selected hard question, it first writes $G_{\mathrm{T}} = 4$ drafts without seeing student errors. It then revises one graded-correct draft in a single call, using all K_x current wrong student rollouts (Fig. 21). Those rollouts are passed in full, without the candidate compression, parse filter, or answer list that NCQ applies to the same rollouts. The SFT input is the bare question, so the mistakes shape only the target. The refinement is graded before it becomes a target. Cross-entropy covers its response tokens only (Algorithm 8).

Below is a question, your own draft solution, and several incorrect attempts by a weaker student. Revise your draft so that it makes explicit the reasoning the student is missing: expand the steps the attempts get wrong, and keep the rest concise. Output only the revised solution, and end it with the final answer in `\boxed{}` *format.*
`<question>`
x
`</question>`
`<your_draft>`
$y_{\mathrm{T}}^{\mathrm{draft}}$
`</your_draft>`
`<student_attempts>`
$y_{\mathrm{S}}^{(-,1)}, \ldots, y_{\mathrm{S}}^{(-,K_x)}$
`</student_attempts>`

Figure 21: Error-conditioned refinement prompt for Algorithm 8. The frozen teacher revises one of its graded-correct drafts using the current wrong student rollouts.

E.3 Compute cost per run

Tab. 12 reports the measured end-to-end wall-clock time of one completed run of 200 rollout steps on the hardware of Tab. 11.

Algorithm 8: Executed Error-reflection-SFT step. The frozen teacher is the refiner, and its refinement is graded before it becomes a target.

Input: Student π_θ, frozen teacher π_{T}, dataset $\mathcal{D}$, buffer $\mathcal{B}$; $G_{\mathrm{T}}=4$, G_{S}, I, $\rho_{\mathrm{replay}}, \rho_{\mathrm{aug}}, |\mathcal{B}|_{\max}, \tau$.

Sample $X_{\mathrm{new}} \sim \mathcal{D}$, draw X_{replay} from $\mathcal{B}$ by the replay rule, and set $X \leftarrow \mathrm{concat}(X_{\mathrm{new}}, X_{\mathrm{replay}})$.
foreach $x \in X$ **do**
  Draw G_{S} student rollouts $\{y_{\mathrm{S}}^{(g)}\}_{g=1}^{G_{\mathrm{S}}} \sim \pi_\theta(\cdot|x)$, score them, and compute $\bar{r}_x$.
end
Select X_{aug}: at most the $\rho_{\mathrm{aug}}|X_{\mathrm{new}}|$ hardest questions with $\bar{r}_x < \tau$, ranked by ascending $\bar{r}_x$.
foreach $x \in X_{\mathrm{aug}}$ **do**
  Draw G_{T} fresh teacher drafts $\{y_{\mathrm{T}}^{(g)}\}_{g=1}^{G_{\mathrm{T}}} \sim \pi_{\mathrm{T}}(\cdot|x)$, grade them, and keep the N_x correct ones.
  Collect the K_x wrong student rollouts $\{y_{\mathrm{S}}^{(-,m)}\}_{m=1}^{K_x}$ from $\{y_{\mathrm{S}}^{(g)}\}$.
  if $N_x > 0$ **then**
    Choose one correct draft $y_{\mathrm{T}}^{\mathrm{draft}}$ uniformly and query π_{T} with Fig. 21 on x, $y_{\mathrm{T}}^{\mathrm{draft}}$, and all K_x wrong rollouts to obtain the refinement $y_{\mathrm{T}}^{\mathrm{ref}}$.
    Grade $y_{\mathrm{T}}^{\mathrm{ref}}$ with the shared outcome reward.
    if $y_{\mathrm{T}}^{\mathrm{ref}}$ *is correct* **then**
      Accumulate cross-entropy on $(x, y_{\mathrm{T}}^{\mathrm{ref}})$ over the refined response tokens only.
    end
  end
end
Split X into I optimizer mini-batches of 96 new prompts each, as in the RL rows. Apply one AdamW update per mini-batch on its cross-entropy, and skip a mini-batch with no verified target.
Update $\mathcal{B}$ from the plain-rollout $\bar{r}_x$: admit, graduate, and FIFO-evict.
return π_θ, $\mathcal{B}$.

Table 12: Measured end-to-end wall-clock hours of one completed training run, rounded to the nearest hour. Each run uses its own 64×NVIDIA H100-80 GB allocation. The time covers student generation and optimization, teacher generation and candidate compression where used, teacher scoring in the distillation rows, reward judging, synchronization, and data movement.

Method	0.8B	2B	4B	9B
Off-Distill	85	91	95	104
Off-Distill†	88	96	100	108
On-Distill	75	78	82	88
On-Distill†	78	82	83	90
GRPO	61	69	75	90
GRPO†	61	69	76	93
ZPPO	68	79	92	110

What ZPPO adds. On top of GRPO†, ZPPO adds teacher rollouts, candidate compression, and BCQ and NCQ groups, for at most 576 groups per rollout step. It takes 7/10/16/17 hours more than GRPO† at 0.8B/2B/4B/9B.

F Extended main-paper tables (4B and 9B)

Tab. 13 (16 VLM benchmarks) and Tab. 14 (10 LLM and 5 Video benchmarks) report the 4B and 9B blocks of Tab. 1 and Tab. 2. Tab. 15 does the same for Tab. 3. A per-setting Δ is ZPPO minus the strongest non-ZPPO value in that column at the same scale. An Avg Δ is ZPPO minus the strongest non-ZPPO macro average. The ablation tables print no Δ row. The generalization tables print only the † variant of each baseline.

Table 13: Main results on the 16 VLM benchmarks at 4B and 9B; † denotes augmentation with the prompt replay buffer, and Δ reports ZPPO minus the strongest non-ZPPO method in each column on the same model size. Bold marks the column maximum and is shared by every row that attains it; the colored Δ row is a difference, not a maximum. 0.8B and 2B results: Tab. 1. Benchmark abbreviations: Appendix C.2.

16 VLM Benchmarks	AI2D	BabyV	CharXiv	DynaM	EmbSp	InfoVQA	MVerse	MVision	MVista	MMMU^{Pro}	MM-Vet	OCR^{EN}	OCR^{ZH}	VisP	VBlind	WeMath	Avg
Qwen3.5-4B	88.7	16.0	78.7	59.3	82.1	88.5	83.1	64.3	83.9	64.0	78.4	50.8	28.9	47.4	66.4	89.2	66.9
+ Off-Distill	87.2	17.5	80.3	60.9	80.6	86.4	83.7	67.0	83.0	64.8	76.8	51.0	29.4	47.6	66.9	90.2	67.1
+ Off-Distill†	87.0	17.7	80.0	60.7	81.1	86.8	83.7	67.5	82.9	65.1	77.7	52.0	28.3	48.8	67.4	90.6	67.3
+ On-Distill	87.1	17.5	80.7	61.5	81.0	87.2	83.9	67.8	84.0	65.6	78.2	51.3	29.5	48.2	67.8	90.6	67.6
+ On-Distill†	88.2	17.4	80.5	61.6	80.5	87.7	84.5	67.6	83.8	65.6	78.2	52.0	30.0	48.7	67.8	91.4	67.8
+ GRPO	88.4	18.9	80.5	61.5	81.3	88.0	84.6	68.3	84.1	66.0	78.7	52.2	30.6	49.2	68.4	90.8	68.2
+ GRPO†	88.5	20.9	81.4	61.4	82.0	88.1	84.5	68.3	85.3	65.5	78.4	52.4	30.4	51.0	69.4	91.8	68.7
+ ZPPO	**89.6**	**27.8**	**82.8**	**63.9**	**82.4**	**88.8**	**85.7**	**72.2**	**85.4**	**67.1**	**79.8**	**54.6**	**33.0**	**54.5**	**74.9**	**92.6**	**70.9**
Δ	+0.9	+6.9	+1.4	+2.3	+0.3	+0.3	+1.1	+3.9	+0.1	+1.1	+1.1	+2.2	+2.4	+3.5	+5.5	+0.8	+2.2
Qwen3.5-9B	90.2	22.9	82.9	62.5	**82.1**	87.8	86.9	70.8	86.3	67.7	83.0	50.0	31.7	55.3	71.7	90.9	70.2
+ Off-Distill	89.1	23.0	82.0	64.7	80.9	88.1	85.7	73.2	85.5	67.9	81.4	51.1	32.4	54.4	73.4	91.0	70.2
+ Off-Distill†	89.2	23.4	82.6	64.8	80.7	88.6	86.1	72.8	85.5	68.6	81.1	51.6	32.2	55.2	73.7	91.5	70.5
+ On-Distill	89.5	24.3	82.2	64.6	81.0	89.5	86.5	72.9	86.4	69.1	82.8	51.6	32.6	54.1	73.3	91.5	70.7
+ On-Distill†	89.1	24.0	83.0	64.6	80.9	89.5	86.4	73.5	86.4	69.6	82.9	52.0	32.5	55.6	73.6	91.8	71.0
+ GRPO	89.8	24.7	83.4	65.3	80.4	89.6	87.4	74.3	86.0	69.1	83.2	52.2	32.9	55.9	73.3	91.8	71.2
+ GRPO†	90.2	24.0	83.2	65.1	81.4	89.6	87.9	75.0	87.3	68.9	83.0	52.2	32.9	55.7	74.3	92.2	71.4
+ ZPPO	**90.8**	**26.3**	**83.7**	**66.7**	**82.1**	**91.4**	**88.1**	**75.9**	**87.5**	**70.2**	**83.9**	**56.7**	**35.6**	**56.4**	**79.4**	**92.9**	**73.0**
Δ	+0.6	+1.6	+0.3	+1.4	0.0	+1.8	+0.2	+0.9	+0.2	+0.6	+0.7	+4.5	+2.7	+0.5	+5.1	+0.7	+1.6

Table 14: Evaluation over the 10 LLM and 5 Video benchmarks at 4B and 9B (probing generalization beyond the training data). Δ is ZPPO minus the strongest non-ZPPO row per column. Bold marks the column maximum, shared by every row attaining it; the colored Δ row is a difference. 0.8B and 2B: Tab. 2. Benchmark abbreviations: Appendix C.2.

	10 LLM Benchmarks											5 Video Benchmarks					
Method	AIME25	AIME26	CEval	GPQA-D	HLE	IMO-AB	MMLU	MMLU-Pro	MMLU-Rd	MultiCh	Avg	MMVU	MVBench	VMME	VMME^{S}	VMMMU	Avg
Qwen3.5-4B	53.3	63.3	83.6	69.7	7.8	35.5	85.3	79.4	89.5	48.9	61.6	69.2	**70.2**	68.1	69.9	**80.3**	71.5
+ Off-Distill†	53.3	63.3	83.0	68.2	6.3	35.7	83.7	77.3	88.1	48.1	60.7	67.9	67.4	66.4	68.8	78.2	69.7
+ On-Distill†	53.3	63.3	82.6	69.3	7.8	35.8	85.0	77.8	88.6	48.2	61.2	68.7	67.8	67.0	69.3	78.9	70.3
+ GRPO†	53.3	63.3	84.6	73.5	7.1	40.6	85.9	79.2	88.9	50.7	62.7	69.2	69.1	68.2	68.7	79.4	70.9
+ ZPPO	**56.7**	**66.7**	**86.1**	**77.3**	**8.3**	**48.0**	**86.0**	**79.7**	**90.1**	**56.0**	**65.5**	**70.6**	69.6	**68.5**	**70.2**	80.2	**71.8**
Δ	+3.4	+3.4	+1.5	+3.8	+0.5	+7.4	+0.1	+0.3	+0.6	+5.3	+2.8	+1.4	−0.6	+0.3	+0.3	−0.1	+0.3
Qwen3.5-9B	60.0	70.0	84.0	77.3	8.9	38.8	87.3	81.1	90.5	50.8	64.9	73.8	71.4	**70.5**	72.4	80.9	73.8
+ Off-Distill†	60.0	70.0	84.5	76.8	8.4	39.3	87.5	79.9	89.5	51.0	64.7	72.5	69.6	68.5	71.5	80.8	72.6
+ On-Distill†	60.0	70.0	84.2	77.4	9.2	39.4	87.2	81.2	90.5	52.1	65.1	73.0	70.4	69.6	71.6	80.7	73.1
+ GRPO†	63.3	70.0	85.3	77.8	8.8	43.6	87.4	81.3	90.9	53.0	66.1	73.1	71.5	69.3	71.2	82.0	73.4
+ ZPPO	**70.0**	**73.3**	**87.2**	**78.8**	**9.8**	**49.8**	**88.1**	**82.0**	**91.7**	**56.8**	**68.8**	**74.2**	**71.7**	70.2	**72.8**	**82.1**	**74.2**
Δ	+6.7	+3.3	+1.9	+1.0	+0.6	+6.2	+0.6	+0.7	+0.8	+3.8	+2.7	+0.4	+0.2	−0.3	+0.4	+0.1	+0.4

F.1 Setting-level win counts over all four scales

We count ZPPO against the untrained base and against GRPO† in each of the $4 \times 31 = 124$ (scale, setting) cells of Tab. 1, Tab. 2, Tab. 13, and Tab. 14. A cell is *above*, *level*, or *below* when the printed one-decimal difference between the two rows is positive, zero, or negative.

(i) ZPPO is above the untrained base in 117 cells, level in 4, and below in 3. The level cells are AIME25 and AIME26 at 0.8B, AIME25 at 2B, and EmbSp at 9B. All three below cells are Video settings: MVBench (−0.6 pp) and VMMMU (−0.1) at 4B, and VMME (−0.3) at 9B. No VLM or LLM cell falls below the base. The Video Avg stays above the base at every scale (+4.6/+2.6/+0.3/+0.4 pp at 0.8B/2B/4B/9B).

Table 15: Component ablation of ZPPO over the 16 VLM benchmarks at 4B and 9B. Rows add the prompt replay buffer (†), BCQ, and NCQ to GRPO, alone or combined. Bold marks the column maximum and is shared by every row that attains it. 0.8B and 2B: Tab. 3. Benchmark abbreviations: Appendix C.2.

16 VLM Benchmarks	AI2D	BabyV	CharXiv	DynaM	EmbSp	InfoVQA	MVerse	MVision	MVista	MMMU^{Pro}	MM-Vet	OCR^{EN}	OCR^{ZH}	VisP	VBlind	WeMath	Avg
Qwen3.5-4B	88.7	16.0	78.7	59.3	82.1	88.5	83.1	64.3	83.9	64.0	78.4	50.8	28.9	47.4	66.4	89.2	66.9
+ GRPO	88.4	18.9	80.5	61.5	81.3	88.0	84.6	68.3	84.1	66.0	78.7	52.2	30.6	49.2	68.4	90.8	68.2
+ GRPO†	88.5	20.9	81.4	61.4	82.0	88.1	84.5	68.3	85.3	65.5	78.4	52.4	30.4	51.0	69.4	91.8	68.7
+ GRPO + Both	89.0	20.7	81.3	61.0	82.1	87.8	84.2	67.0	84.7	65.9	79.0	51.2	30.6	50.5	69.5	91.3	68.5
+ GRPO† + BCQ	89.5	24.3	81.8	61.9	81.9	88.2	85.1	69.7	85.2	65.9	79.3	53.9	31.8	52.3	72.3	91.0	69.6
+ GRPO† + NCQ	89.3	24.9	81.4	62.7	81.7	88.7	85.2	70.2	84.9	66.3	79.4	54.2	32.1	52.2	72.8	92.1	69.9
+ ZPPO	**89.6**	**27.8**	**82.8**	**63.9**	**82.4**	**88.8**	**85.7**	**72.2**	**85.4**	**67.1**	**79.8**	**54.6**	**33.0**	**54.5**	**74.9**	**92.6**	**70.9**
Qwen3.5-9B	90.2	22.9	82.9	62.5	**82.1**	87.8	86.9	70.8	86.3	67.7	83.0	50.0	31.7	55.3	71.7	90.9	70.2
+ GRPO	89.8	24.7	83.4	65.3	80.4	89.6	87.4	74.3	86.0	69.1	83.2	52.2	32.9	55.9	73.3	91.8	71.2
+ GRPO†	90.2	24.0	83.2	65.1	81.4	89.6	87.9	75.0	87.3	68.9	83.0	52.2	32.9	55.7	74.3	92.2	71.4
+ GRPO + Both	90.3	24.3	83.3	64.5	80.9	89.8	87.7	73.5	86.0	69.3	83.6	52.7	32.5	56.0	74.5	92.2	71.3
+ GRPO† + BCQ	90.3	25.3	83.3	65.4	82.0	90.6	87.3	74.6	86.5	70.1	83.5	55.0	34.0	56.0	76.2	92.4	72.0
+ GRPO† + NCQ	90.7	26.0	82.9	66.4	81.6	90.7	87.9	75.4	86.9	70.0	83.3	56.2	35.2	56.2	78.4	92.8	72.5
+ ZPPO	**90.8**	**26.3**	**83.7**	**66.7**	**82.1**	**91.4**	**88.1**	**75.9**	**87.5**	**70.2**	**83.9**	**56.7**	**35.6**	**56.4**	**79.4**	**92.9**	**73.0**

(ii) ZPPO is above GRPO† in 121 cells, level in 3, and below in 0. ZPPO and GRPO† both use the replay buffer, so their gap shows what BCQ and NCQ add. The level cells are AIME25 and AIME26 at 0.8B and AIME25 at 2B. Because of these ties, the LLM *Win* entries of the ZPPO − GRPO† column in Tab. 20 read $8/10$ at 0.8B and $9/10$ at 2B. The other ten (scale, family) blocks read $16/16$ on VLM and $5/5$ on Video at every scale, and $10/10$ on LLM at 4B and 9B. At those two scales, ZPPO is therefore above GRPO† in all 31 settings.

F.2 Numbers behind the introduction figure and box

Fig. 1 has one panel each for the 10 LLM (left), 16 VLM (middle), and 5 Video (right) benchmarks at the four Qwen3.5 student scales. Each bar is a method's Avg on one family minus the Avg of the untrained base at the same scale. The base is the $y = 0$ line, which the legend calls *Original*. The LLM and Video panels use the Avg columns of Tab. 2 (0.8B/2B) and Tab. 14 (4B/9B). The VLM panel uses those of Tab. 1 and Tab. 13. The four series are the buffer-augmented rows Off-Distill†, On-Distill†, GRPO†, and ZPPO. The figure does not plot the buffer-free Off-Distill, On-Distill, and GRPO rows of Tab. 1 and Tab. 13 (Appendix A). Tab. 16 prints every plotted value.

(i) In our setting, distillation *hurts generalization beyond the training corpus*, consistent with prior reports (Sec. 1). The best distillation variant stays within 1.3 pp of the original on VLM (On-Distill† at 0.8B). Off-Distill† and On-Distill† *lose* accuracy on Video and *degrade* the pooled LLM–Video average of Eq. (10) at every scale (Tab. 17), sharpest at 0.8B.

(ii) ZPPO *improves* LLM and Video. Every ZPPO bar in Fig. 1 is above the original, including the LLM and Video families beyond the training corpus. GRPO†'s Video bars fall below the original at 4B and 9B (-0.6/-0.4 pp), while ZPPO's stay above ($+0.3$/$+0.4$ pp).

(iii) The VLM gains quoted in Sec. 1 are the 0.8B and 9B ZPPO bars. The $+9.3$ pp gain over the untrained base at 0.8B is $50.3 - 41.0$ in Tab. 1. The $+2.8$ pp gain at 9B is $73.0 - 70.2$ in Tab. 13. Tab. 16 prints both in its ZPPO row.

Table 16: Values plotted in Fig. 1: macro-average gain Δ (pp) over the untrained base of the same scale. LLM and Video from Tab. 2 and Tab. 14; VLM from Tab. 1 and Tab. 13. † denotes augmentation with the prompt replay buffer.

Plotted series (table row)	10 LLM				16 VLM				5 Video			
	0.8B	2B	4B	9B	0.8B	2B	4B	9B	0.8B	2B	4B	9B
Off-Policy Distill (Off-Distill†)	−2.7	−2.2	−0.9	−0.2	+0.5	+0.5	+0.4	+0.3	−3.3	−2.0	−1.8	−1.2
On-Policy Distill (On-Distill†)	−2.0	−1.5	−0.4	+0.2	+1.3	+1.2	+0.9	+0.8	−2.5	−1.4	−1.2	−0.7
GRPO (GRPO†)	+3.5	+2.1	+1.1	+1.2	+4.5	+2.4	+1.8	+1.2	+2.2	+1.3	−0.6	−0.4
ZPPO	+7.9	+5.1	+3.9	+3.9	+9.3	+5.2	+4.0	+2.8	+4.6	+2.6	+0.3	+0.4

The boxed takeaway of Sec. 1. Tab. 17 derives every number in the box from the gains over the untrained base in Tab. 16. The second part of the box pools the 10 LLM and 5 Video benchmarks by

setting count,

$$\Delta_{\text{LLM+Video}} \;=\; \frac{10\,\Delta_{\text{LLM}} + 5\,\Delta_{\text{Video}}}{15}, \tag{10}$$

so each of the 15 settings carries the same weight.

Table 17: Numbers in the boxed takeaway of Sec. 1, as gains (pp) over the untrained base from Tab. 16. The LLM+Video block pools by Eq. (10). The box quotes the one-decimal rows.

Series	16 VLM				10 LLM + 5 Video			
	0.8B	2B	4B	9B	0.8B	2B	4B	9B
Off-Distill†	+0.50	+0.50	+0.40	+0.30	−2.90	−2.13	−1.20	−0.53
On-Distill†	+1.30	+1.20	+0.90	+0.80	−2.17	−1.47	−0.67	−0.10
Mean of the two	+0.90	+0.85	+0.65	+0.55	−2.53	−1.80	−0.93	−0.32
as quoted	+0.9	+0.9	+0.7	+0.6	−2.5	−1.8	−0.9	−0.3
ZPPO	+9.30	+5.20	+4.00	+2.80	+6.80	+4.27	+2.70	+2.73
as quoted	+9.3	+5.2	+4.0	+2.8	+6.8	+4.3	+2.7	+2.7

F.3 Where the LLM-family gain sits, by setting type

We decompose the LLM-family gain of ZPPO over GRPO† into two groups of settings. The first holds four knowledge exams (CEval, MMLU, MMLU-Pro, MMLU-Redux). The second holds six derivation, short-answer, or instruction-following settings (AIME25, AIME26, GPQA-Diamond, HLE, IMO-AnswerBench, MultiChallenge). Inside each group, we sum the printed ZPPO − GRPO† differences of Tab. 2 and Tab. 14 and divide by ten. Each group then gives its share of the ten-setting mean. The knowledge exams add +1.47/+0.50/+0.33/+0.41 pp at 0.8B/2B/4B/9B. The other six settings add +2.97/+2.52/+2.45/+2.20 pp. Together they give the +4.44/+3.02/+2.78/+2.61 pp LLM Δ of the ZPPO − GRPO† column in Tab. 20. The six-setting group is larger at every scale. It also holds the three AIME ties with GRPO† (Appendix F.1), which add zero. The largest single gap is GPQA-Diamond at 0.8B (+16.9 pp). GRPO† and ZPPO train under the same instruction and reward. Dropping GPQA-Diamond still leaves +3.06 pp at 0.8B (Appendix G.7).

F.4 Where the VLM-family gain sits, by setting

We subtract the printed GRPO† cell from the ZPPO cell in each of the 16 VLM settings of Tab. 1 and Tab. 13. The difference is positive in all 64 VLM cells of the 124 counted in Appendix F.1. Its mean over the 16 settings is +4.79/+2.76/+2.24/+1.54 pp at 0.8B/2B/4B/9B. These are the 16/16 *Win* and Δ entries of the ZPPO − GRPO† column in Tab. 20.

G Additional empirical results

G.1 Buffer dynamics across student scales

We track how the prompt replay buffer fills and drains over training in one logged ZPPO run at each student scale. As in Sec. 3.3, admission and graduation use the mean rollout accuracy $\bar{r}_x$ of the plain rollouts on the original question, never a BCQ or NCQ group. Fig. 5a gives the shares of Easy questions and of the BCQ and NCQ reformulations before the caps (Appendix D). Fig. 22 instead follows the buffer entries at each step.

(i) The buffer reaches capacity earlier at smaller scales. The 0.8B run reaches the 10,000-entry capacity first, then 2B, 4B, and 9B. All four reach it within 200 steps (Fig. 22).

(ii) Questions admitted at 0% graduate at every scale. Fig. 23 counts the graduations of entries admitted at 0%: 293/432/562/514 at 0.8B/2B/4B/9B, or 13/16/21/21% of each run's graduations. Over all four admission accuracies, the runs graduate 2,334/2,627/2,645/2,455 entries and FIFO-evict 17,963/7,251/1,910/896 (Fig. 5b). Most entries are FIFO-evicted before they graduate at the small scales, and most graduate before they are FIFO-evicted at the large ones.

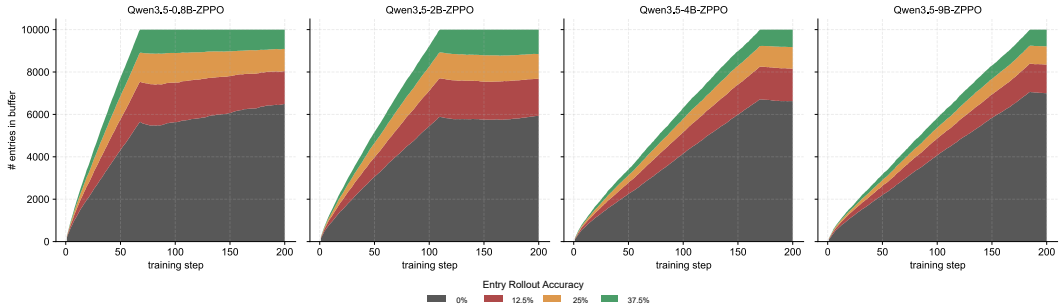


Figure 22: Buffer entries over training for ZPPO at every student scale, split by rollout accuracy at admission. Each panel stacks the four admission accuracies, so the total height is the number of buffer entries.

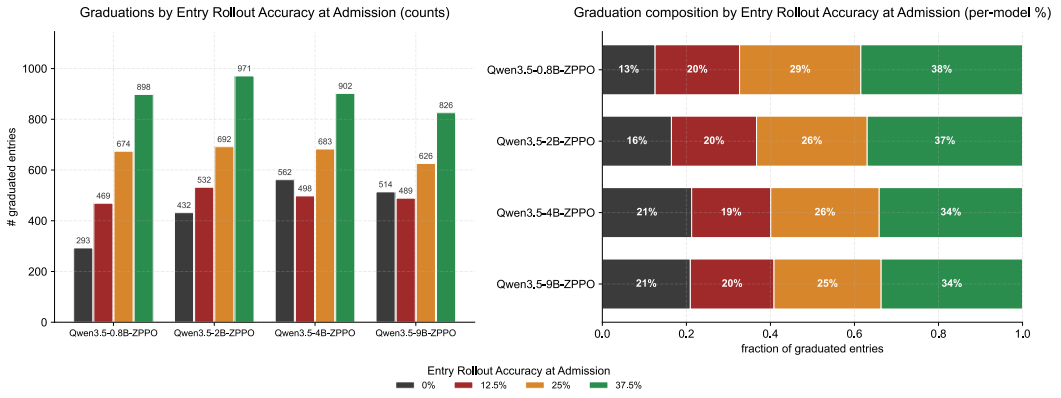


Figure 23: **(Left)** Graduate counts per scale, split by rollout accuracy at admission (0%, 12.5%, 25%, 37.5%). **(Right)** Share of each admission accuracy among the graduates, per scale.

G.2 ZPPO VS. GRPO† TRAINING DYNAMICS AT 2B

We compare one logged ZPPO run with one logged GRPO† run at 2B. The two share every setting of Tab. 11 and differ only in whether BCQ and NCQ are on. Each run admits the questions on which its own plain rollouts give $\bar{r}_x < \tau$, so the two buffers hold different entries.

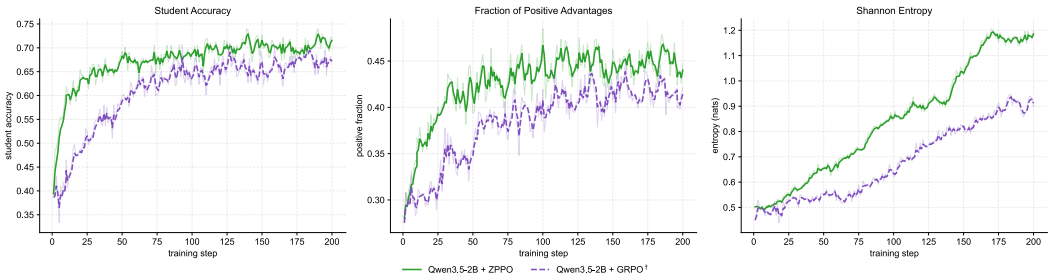


Figure 24: Training dynamics of ZPPO vs. GRPO† at 2B. **(Left)** Student rollout accuracy. **(Center)** Fraction of student rollouts with positive group-relative advantage. **(Right)** Actor entropy.

(i) ZPPO's positive-advantage fraction is above GRPO†'s after the shared start (Fig. 24). The fraction counts every student rollout of the step, and ZPPO's batch adds BCQ and NCQ groups to the plain groups. A reformulated group has nonzero advantage as soon as its own rollouts include both a correct and a wrong one, even when the plain group on the same question is all-wrong.

(ii) ZPPO's actor entropy ends above GRPO†'s. Both curves rise over the 200 steps, and ZPPO is above GRPO† after the early phase (Fig. 24). Both runs use clip-higher, which DAPO introduced to keep entropy from dropping quickly (Yu et al., 2025).

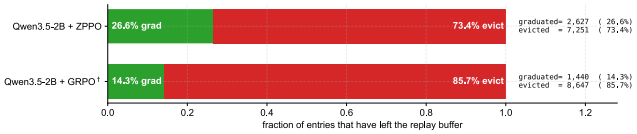


Figure 25: Buffer entries that graduated or were FIFO-evicted, for ZPPO and GRPO† at 2B, as in Fig. 5b.

(iii) ZPPO's graduation rate is higher under the same buffer settings. An entry graduates when a later plain group reaches $\bar{r}_x \geq \tau$. In Fig. 25, ZPPO graduates 2,627 entries and FIFO-evicts 7,251 (26.6/73.4%). GRPO† graduates 1,440 and FIFO-evicts 8,647 (14.3/85.7%). Fig. 4 splits the same graduate totals by rollout accuracy at admission: $432+532+692+971 = 2{,}627$ for ZPPO and $73+160+450+757 = 1{,}440$ for GRPO†.

Fig. 25, like Fig. 5b, counts every entry that has left the buffer by step 200, by graduation or by FIFO eviction. Fig. 4 instead divides the graduates by the entries admitted at each accuracy and re-sampled at least once. It prints 432/1566 vs. 73/2035 at 0% and 532/981 vs. 160/1158 at 12.5%, which Sec. 4.2(i) quotes as 28% vs. 4% and 54% vs. 14%.

(iv) The largest gaps in graduate count are at the two hardest admission accuracies. From the counts in (iii), ZPPO graduates 359/372/242/214 more entries than GRPO† at 0%/12.5%/25%/37.5% (Fig. 26).

G.3 MULTI-METRIC DYNAMICS ACROSS SCALES

Fig. 27 plots six per-step metrics from one logged ZPPO run at each student scale: the policy gradient loss, BCQ-prompt accuracy, NCQ-prompt accuracy, the fraction of positive advantages, student rollout accuracy, and actor entropy. In each panel, the light trace is the per-step value and the heavy trace is its running mean. The BCQ-prompt panel plots the mean outcome reward of the BCQ groups drawn at each step. This is a different population from the audit in Appendix G.8.

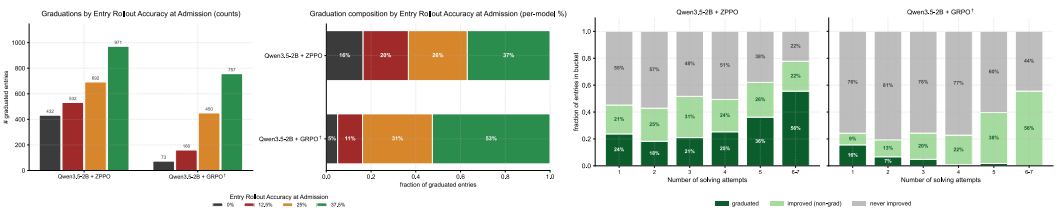


Figure 26: ZPPO vs. GRPO† at 2B. **(Left two panels)** Graduate counts by rollout accuracy at admission, and each method's share of graduates per admission accuracy. **(Right two panels)** For each number of later attempts, the fraction of entries that graduated, improved without graduating, or never improved.

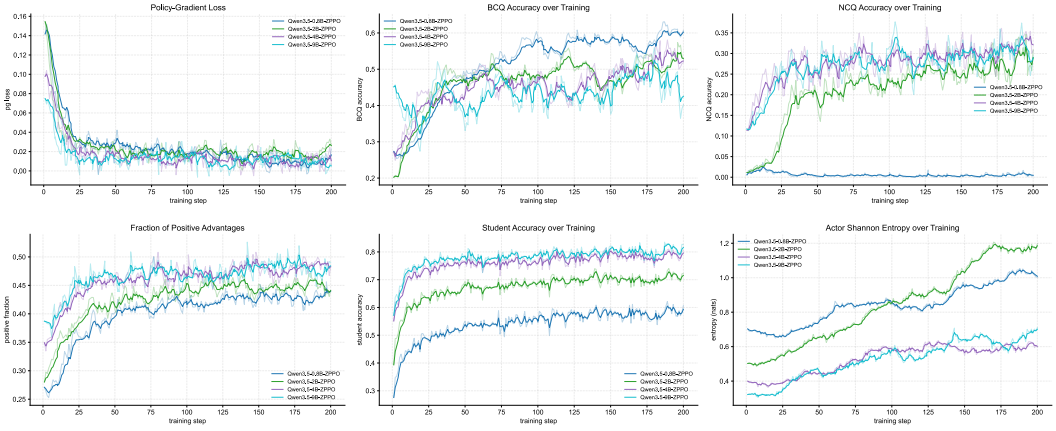


Figure 27: ZPPO training dynamics at all four student scales. **(Top row)** Policy gradient loss, BCQ-prompt accuracy, NCQ-prompt accuracy. **(Bottom row)** Fraction of positive advantages, student rollout accuracy, actor entropy.

Table 18: Component ablation of ZPPO over the 10 LLM and 5 Video benchmarks at all four scales. Rows and formatting follow Tab. 3: the rows add the prompt replay buffer (†), BCQ, and NCQ to GRPO, alone or combined, and *Both* = BCQ + NCQ without the buffer. Distillation baselines: Tab. 2 and Tab. 14. Benchmark abbreviations: Appendix C.2.

	10 LLM Benchmarks											5 Video Benchmarks					
Method	AIME25	AIME26	CEval	GPQA-D	HLE	IMO-AB	MMLU	MMLU-Pro	MMLU-Rd	MultiCh	Avg	MMVU	MVBench	VMME	VMME^S	VMMMU	Avg
Qwen3.5-0.8B	**3.3**	0.0	52.3	12.1	2.2	3.3	57.7	41.7	62.5	16.6	25.2	39.0	51.6	50.3	51.6	48.8	48.3
+ GRPO	0.0	0.0	53.9	20.0	3.2	5.1	59.4	44.8	64.5	19.7	27.1	39.9	53.2	51.4	52.7	50.8	49.6
+ GRPO†	**3.3**	0.0	54.5	25.5	3.5	7.0	59.9	47.1	65.1	20.8	28.7	41.3	53.5	52.2	53.0	52.5	50.5
+ GRPO + Both	**3.3**	0.0	54.8	25.2	2.6	5.8	60.4	45.9	64.2	20.7	28.3	40.3	53.5	51.0	51.8	51.3	49.6
+ GRPO† + BCQ	**3.3**	0.0	56.7	35.8	4.4	8.9	62.1	50.0	67.9	25.9	31.5	41.8	55.8	53.3	54.4	54.8	52.0
+ GRPO† + NCQ	**3.3**	0.0	55.2	27.0	4.1	7.4	60.6	47.5	66.0	22.5	29.4	41.6	54.3	52.5	53.6	52.9	51.0
+ ZPPO	**3.3**	0.0	**57.6**	**42.4**	**5.0**	**10.5**	**63.0**	**51.7**	**69.0**	**28.6**	**33.1**	**42.5**	**56.6**	**53.9**	**54.9**	**56.4**	**52.9**
Qwen3.5-2B	**26.7**	30.0	71.1	53.5	3.5	11.3	74.6	66.2	80.4	35.8	45.3	53.7	60.7	59.9	61.2	67.6	60.6
+ GRPO	**26.7**	33.3	72.0	55.3	3.8	15.8	74.9	66.8	81.1	37.0	46.7	55.1	61.3	60.7	62.1	66.8	61.2
+ GRPO†	**26.7**	33.3	72.4	56.5	4.2	19.3	75.4	67.3	81.4	37.3	47.4	56.7	61.7	61.2	62.6	67.3	61.9
+ GRPO + Both	**26.7**	33.3	72.7	56.1	4.3	18.3	75.4	66.6	80.4	36.7	47.1	55.7	61.0	60.3	61.6	67.3	61.2
+ GRPO† + BCQ	**26.7**	36.7	73.4	58.7	4.9	24.6	76.0	68.0	82.3	39.3	49.1	57.6	62.3	61.9	63.4	67.9	62.6
+ GRPO† + NCQ	**26.7**	36.7	73.0	57.8	4.6	22.3	75.8	67.8	81.9	38.7	48.5	56.9	62.0	61.8	63.1	67.5	62.3
+ ZPPO	**26.7**	**40.0**	**74.0**	**60.6**	**5.1**	**29.5**	**76.4**	**68.2**	**82.9**	**40.6**	**50.4**	**59.0**	**62.7**	**62.4**	**63.9**	**68.1**	**63.2**
Qwen3.5-4B	53.3	63.3	83.6	69.7	7.8	35.5	85.3	79.4	89.5	48.9	61.6	69.2	**70.2**	68.1	69.9	**80.3**	71.5
+ GRPO	53.3	63.3	84.1	70.9	7.5	38.9	85.3	78.1	88.6	50.5	62.1	69.1	68.3	66.8	69.2	79.6	70.6
+ GRPO†	53.3	63.3	84.6	73.5	7.1	40.6	85.9	79.2	88.9	50.7	62.7	69.2	69.1	68.2	68.7	79.4	70.9
+ GRPO + Both	53.3	63.3	84.2	72.7	8.2	39.6	85.5	79.1	89.7	51.5	62.7	70.2	68.1	67.7	69.2	79.5	70.9
+ GRPO† + BCQ	53.3	**66.7**	85.2	74.9	8.2	44.4	85.4	78.9	89.4	54.3	64.1	70.3	69.0	68.3	69.3	79.4	71.3
+ GRPO† + NCQ	**56.7**	**66.7**	85.5	75.4	8.2	44.5	85.6	79.5	89.7	54.2	64.6	70.2	69.2	68.4	70.0	80.0	71.6
+ ZPPO	**56.7**	**66.7**	**86.1**	**77.3**	**8.3**	**48.0**	**86.0**	**79.7**	**90.1**	**56.0**	**65.5**	**70.6**	69.6	**68.5**	**70.2**	80.2	**71.8**
Qwen3.5-9B	60.0	70.0	84.0	77.3	8.9	38.8	87.3	81.1	90.5	50.8	64.9	73.8	71.4	**70.5**	72.4	80.9	73.8
+ GRPO	63.3	70.0	84.3	77.4	8.9	41.5	87.5	80.6	90.9	52.2	65.7	73.5	70.9	69.2	71.6	80.9	73.2
+ GRPO†	63.3	70.0	85.3	77.8	8.8	43.6	87.4	81.3	90.9	53.0	66.1	73.1	71.5	69.3	71.2	82.0	73.4
+ GRPO + Both	63.3	70.0	85.3	77.9	9.2	42.6	87.6	81.0	90.8	52.9	66.1	73.1	71.1	69.5	72.1	81.4	73.4
+ GRPO† + BCQ	66.7	**73.3**	86.7	77.8	9.0	45.8	87.9	81.8	91.0	54.5	67.5	73.8	71.5	68.8	72.6	81.5	73.6
+ GRPO† + NCQ	**70.0**	**73.3**	86.9	78.1	9.5	48.3	87.8	81.7	91.6	55.6	68.3	73.9	71.6	69.7	72.5	81.7	73.9
+ ZPPO	**70.0**	**73.3**	**87.2**	**78.8**	**9.8**	**49.8**	**88.1**	**82.0**	**91.7**	**56.8**	**68.8**	**74.2**	**71.7**	70.2	**72.8**	**82.1**	**74.2**

NCQ at 0.8B is sparse, but it recurs on every step. Every step at 0.8B has hard questions with parsed wrong rollouts, so every step builds NCQ instances. On the audited steps, they fill 68.7 of the 96 instance slots (Appendix G.8). Their accuracy is far below that of the other scales but stays above zero throughout training (Fig. 27). On the audited steps, 1.2% of 0.8B NCQ rollouts are correct and 82.7% repeat one of the listed wrong answers (Tab. 23). Across several hundred NCQ rollouts per step, correct ones therefore keep arriving, a few at a time. The cap keeps the hardest questions first. At 0.8B, most buffer entries were admitted at 0% (Fig. 22). Many NCQ instances therefore come from questions whose plain group is all-wrong and has zero advantage. The teacher also fails on 60% of the 0.8B post-cap questions (Appendix G.8), so these questions have no BCQ instance. On them, a correct NCQ rollout is the only positive example the student gets, and it receives the largest advantage in its group ($+7/8$ before batch normalization when it is the only correct one). A 0.8B student has the least to draw on and the most all-wrong questions, so these small but recurring signals weigh more for it than for the larger students. NCQ's marginal contribution at 0.8B is $+1.7$ pp on VLM and $+1.6$ pp on LLM (Sec. 4.3, Appendix G.4). One reading is that it is the cumulative effect of these signals. Extra groups without NCQ content do not reproduce it. Raising GRPO†'s replay fraction to 0.50 adds up to 96 replayed groups per step and moves its 0.8B averages by $+0.4/+0.2/-0.1$ pp on LLM/VLM/Video (Appendix I.2). The NCQ prompt contains no correct answer, so no success copies a candidate. On multiple-choice questions, the listed answers can narrow the options (Appendix I.11). No control isolates the sparse successes from the rest of the NCQ branch.

G.4 Component ablation on the LLM and Video benchmarks

Tab. 18 repeats the component ablation of Tab. 3 on the 10 LLM and 5 Video benchmarks at all four scales.

(i) Neither component suffices alone. The buffer on its own (GRPO†) and BCQ+NCQ without the buffer (GRPO + *Both*) are each at or above GRPO in every block. Yet GRPO + *Both* never beats the buffer alone. Over the 12 (scale, family) blocks of Tab. 3, Tab. 15, and Tab. 18, it is below GRPO† in 8, tied in 4, and above in 0.

(ii) Replay $\times$ reformulation is super-additive. Tab. 3, Tab. 15, and Tab. 18 together hold 24 single-branch cells (four scales $\times$ three families $\times$ two branches). In *all* 24, adding the branch on top of the buffer lifts the macro average above GRPO†. Four rows size the interaction: GRPO (neither), GRPO† (buffer only), GRPO + *Both* (reformulation only), and ZPPO (both). On LLM, the buffer alone

gains $+1.6/+0.7/+0.6/+0.4$ pp over GRPO at 0.8B/2B/4B/9B, and the reformulation alone gains $+1.2/+0.4/+0.6/+0.4$ pp. ZPPO gains $+6.0/+3.7/+3.4/+3.1$ pp, which is $+3.2/+2.6/+2.2/+2.3$ pp above the sum of the two parts. The same excess is $+3.4/+2.6/+1.9/+1.5$ pp on VLM and $+2.4/+1.3/+0.6/+0.6$ pp on Video. Three parts of the loop can contribute to this interaction: replay brings hard questions back into the step, the cap reformulates the hardest questions first, and each visit draws fresh candidates. The four rows size their joint effect.

(iii) The marginal contributions of BCQ and NCQ change with scale. Dropping one branch from ZPPO leaves the other branch's row. ZPPO − GRPO†+NCQ therefore measures what BCQ adds, and ZPPO − GRPO†+BCQ measures what NCQ adds. The three rows share the caps of Algorithm 1. In ZPPO, BCQ and NCQ instances compete for the 96 instance slots of a step, and each question's BCQ instance comes before its NCQ instance. A single-branch row leaves every slot to its one branch. Adding a branch can therefore push out instances of the other branch on lower-ranked questions. Each marginal contribution thus measures one branch under the same cap. On LLM, BCQ adds $+3.75/+1.87/+0.89/+0.47$ pp and NCQ adds $+1.61/+1.34/+1.42/+1.30$ pp at 0.8B/2B/4B/9B (Tab. 21). Both stay positive, so dropping either branch lowers the macro average at every scale. NCQ applies to every hard question with a parsed wrong rollout, but a BCQ instance also needs a correct teacher response (Step 4 of Algorithm 1). Appendix G.8 measures how often the teacher supplies one. GRPO†+BCQ leads GRPO†+NCQ on all three families at 0.8B and 2B, and GRPO†+NCQ leads at 4B and 9B. Full ZPPO has the highest macro average in every block.

G.5 RL RECIPE SWEEP RESULTS

Tab. 19 prints every point of Fig. 6. Both series use the prompt replay buffer. The dashed gray line marks the untrained 2B base, at 45.3/56.8/60.6 on LLM/VLM/Video. Every column other than the default changes one entry of Tab. 11.

Table 19: Values plotted in Fig. 6: macro averages over the 10 LLM, 16 VLM, or 5 Video benchmarks at 2B. The $I=4$ and *Norm w/o Zero* columns are the same configuration, so their entries are identical.

Series	Family	Iterations per step I (Fig. 6a–c)				Batch-level advantage normalization (Fig. 6d–f)		
		1	4	8	16	No norm	Norm w/o Zero	Norm w/ Zero
GRPO†	LLM	46.8	47.4	46.3	46.1	46.0	47.4	43.0
	VLM	58.4	59.2	57.7	57.5	58.3	59.2	54.2
	Video	61.5	61.9	61.3	61.1	61.1	61.9	57.6
ZPPO	LLM	49.8	50.4	49.2	49.0	49.9	50.4	46.1
	VLM	61.4	62.0	60.7	60.4	61.1	62.0	57.1
	Video	62.7	63.2	62.5	62.2	62.8	63.2	59.1

(i) $I=4$ is the sweet spot on all three families. Against the untrained 2B base, GRPO† gains $+2.1/+2.4/+1.3$ pp on LLM/VLM/Video at $I=4$ but only $+0.8/+0.7/+0.5$ pp at $I=16$. ZPPO gains $+5.1/+5.2/+2.6$ pp at $I=4$ and $+3.7/+3.6/+1.6$ pp at $I=16$. Compared with $I=4$, the $I=8$ runs draw $2\times$ the new prompts and the $I=16$ runs $4\times$, yet both stay below $I=4$ in all six (series, family) cells. The $I=1$ runs draw a quarter as many (Appendix E.2). Larger I grows in-step drift, and $I=1$ removes drift but undertrains at matched step count. The sweep moves the new-prompt budget with I, from $0.25\times$ to $4\times$ the default. The budget-matched pair at the end of this subsection holds the budget and the update count fixed for GRPO† at $I=4$ and $I=8$.

(ii) *Norm w/o Zero* is above *No norm* in all six (series, family) cells. The margin is $+1.4/+0.9/+0.8$ pp for GRPO† and $+0.5/+0.9/+0.4$ pp for ZPPO on LLM/VLM/Video.

(iii) *Norm w/ Zero* is below *No norm* in all six cells. The gap is $-3.0/-4.1/-3.5$ pp for GRPO† and $-3.8/-4.0/-3.7$ pp for ZPPO on LLM/VLM/Video. Against each series' *Norm w/o Zero* entry, the gap is $-4.4/-5.0/-4.3$ pp for GRPO† and $-4.3/-4.9/-4.1$ pp for ZPPO, so the drop differs by at most 0.2 pp between the two series. Against the untrained base, *Norm w/ Zero* is $-2.3/-2.6/-3.0$ pp for GRPO† and $+0.8/+0.3/-1.5$ pp for ZPPO. Eq. (6) gives the rescaling that *Norm w/ Zero* applies to each mini-batch (Appendix D). The sweep shows the drop but does not isolate its cause.

(iv) ZPPO stays above GRPO† at every swept setting. In every column of Tab. 19, ZPPO is above GRPO† on all three families, by $+1.1$ to $+3.9$ pp. Every ZPPO row of the τ sweep is also above GRPO† (Tab. 10). Every RL row shares these recipe choices, so the ZPPO − GRPO† gap does not rest on the default values.

Budget-matched I comparison. This comparison runs GRPO† only, at matched prompt draws and update count. At $I=8$, a run of 200 rollout steps with 8 updates each draws $200\times8\times96=153{,}600$ new prompts and runs 1,600 updates. The $(G_{\mathrm{S}}, \text{passes})=(8,2)$ row of Tab. 29 runs the same GRPO† recipe at the default $I=4$ over 400 rollout steps. It draws the same $400\times4\times96=153{,}600$ new prompts and runs the same 1,600 updates. The $I=8$ run scores 46.3/57.7/61.3 on LLM/VLM/Video and the $I=4$ run 47.2/59.4/61.7, so $I=4$ stays 0.9/1.7/0.4 pp ahead.

G.6 Hint and Prefix results

Tab. 4 reports two controls on the shared GRPO† pipeline. Hint (Algorithm 5) gives an answer-free cue abstracted from a solution, and Prefix (Algorithm 6) gives the opening portion of the solution itself. Appendix E.2 gives the executed step of each control. Hint helps VLM but transfers less to the LLM/Video suite, and Prefix transfers even less. GRPO†+BCQ is above both on all three families at both scales. One difference is that BCQ places the correct candidate beside a wrong one, whereas Hint and Prefix pass on correct material only. Without the wrong candidate and the BCQ instruction, BCQ falls below GRPO† on all three families at 2B (Appendix I.4).

At 0.8B, GRPO† scores 28.7/45.5/50.5 on LLM/VLM/Video, Hint 28.8/47.2/50.0, Prefix 27.1/45.5/49.1, GRPO†+BCQ 31.5/48.6/52.0, and ZPPO 33.1/50.3/52.9. At 2B, GRPO† scores 47.4/59.2/61.9, Hint 47.6/60.4/61.6, Prefix 46.3/59.5/61.0, GRPO†+BCQ 49.1/60.8/62.6, and ZPPO 50.4/62.0/63.2. GRPO†+BCQ leads Hint by $+2.7/+1.4/+2.0$ pp at 0.8B and $+1.5/+0.4/+1.0$ pp at 2B. Full ZPPO leads Hint by $+4.3/+3.1/+2.9$ pp at 0.8B and $+2.8/+1.6/+1.6$ pp at 2B.

G.7 Benchmark-selection robustness via paired bootstrap

Inside each (scale, family) block, we redraw the reported settings 10,000 times at random, with repeats allowed. Each time, we recompute the paired macro-average delta from their printed cells. The comparison columns of a row share the same resamples, so all of them are drawn on the same settings. Tab. 20 and Tab. 21 report the full-sample Δ, a percentile 95% interval, and the number of settings on which ZPPO is higher. The full-sample Δ is the mean of the per-setting differences. It can differ slightly from the difference of two printed *Avg* cells, because each *Avg* cell is rounded to one decimal. Tab. 38 does the same for the two other student backbones of Appendix I.9, but uses the exact resample distribution instead of sampling it. Each reported setting is one resampling unit, so VMME and VMME$^{\mathrm{S}}$ enter separately, as do OCR$^{\mathrm{EN}}$ and OCR$^{\mathrm{ZH}}$. These intervals test the choice of benchmarks, not training runs or evaluation items.

(i) The ZPPO − GRPO† interval excludes zero in all 12 (scale, family) cells (Tab. 20). ZPPO is higher on $16/16$ settings in every VLM cell, on $5/5$ in every Video cell, and on 8–10 of 10 on LLM. Every LLM setting where ZPPO is not higher is an exact tie on AIME25 or AIME26 (Appendix F.1). **(ii) The ZPPO − GRPO†+BCQ and ZPPO − GRPO†+NCQ intervals exclude zero in every cell** (Tab. 21). **(iii) The ZPPO − Off-Distill† and ZPPO − On-Distill† intervals exclude zero in all 12 cells.** Together with (i), the interval excludes zero against every trained comparator of Tab. 20.

(iv) Dropping one setting leaves a positive mean. We drop the setting with the largest ZPPO − GRPO† gap from each block and take the paired mean over the remaining 9 LLM, 15 VLM, or 4 Video settings. This leaves $+3.06/+2.22/+2.27/+2.16$ pp on LLM, $+4.50/+2.47/+1.93/+1.31$ pp on VLM, and $+1.98/+1.08/+0.75/+0.58$ pp on Video at 0.8B/2B/4B/9B, all above zero. The dropped LLM setting is GPQA-Diamond at 0.8B, IMO-AnswerBench at 2B and 4B, and AIME25 at 9B. **(v) Dropping AIME leaves a positive mean.** Every tie against GRPO† in Appendix F.1 falls on AIME25 or AIME26. Dropping both still leaves a ZPPO − GRPO† gap of $+5.55/+2.94/+2.63/+2.01$ pp at 0.8B/2B/4B/9B, averaged over the remaining eight LLM settings.

G.8 BCQ and NCQ candidate-match audit

We measure candidate-match rates on the last three logged sets of instances of one ZPPO run per scale (Appendix A gives the populations). A match compares normalized final `\boxed` strings.

BCQ candidate availability. Fig. 5a covers all 200 steps, before the caps. Here we count after the caps, on the audited steps. BCQ gives 27.3/18.7/10.3/7.3 instances per audited step at 0.8B/2B/4B/9B,

Table 20: Paired bootstrap 95% intervals over benchmark settings for the ZPPO − comparator macro-average delta (pp) of Tab. 1, Tab. 2, Tab. 13, and Tab. 14, computed as in Appendix G.7. Δ is the full-sample paired mean, and *Win* counts the settings on which ZPPO is above the comparator.

Scale	Family	ZPPO − Off-Distill† Δ [95% CI] (Win)	ZPPO − On-Distill† Δ [95% CI] (Win)	ZPPO − GRPO† Δ [95% CI] (Win)
0.8B	VLM	+8.78 [+6.87, +10.61] (16/16)	+7.93 [+6.18, +9.63] (16/16)	+4.79 [+3.45, +6.12] (16/16)
0.8B	LLM	+10.56 [+5.87, +16.47] (9/10)	+9.88 [+5.30, +15.87] (9/10)	+4.44 [+2.02, +7.77] (8/10)
0.8B	Video	+7.86 [+6.80, +9.44] (5/5)	+7.04 [+5.62, +8.96] (5/5)	+2.36 [+1.54, +3.30] (5/5)
2B	VLM	+4.72 [+3.69, +5.83] (16/16)	+4.02 [+3.03, +5.09] (16/16)	+2.76 [+1.89, +3.74] (16/16)
2B	LLM	+7.27 [+4.40, +10.85] (9/10)	+6.63 [+4.24, +9.82] (10/10)	+3.02 [+1.36, +5.19] (9/10)
2B	Video	+4.62 [+3.34, +6.18] (5/5)	+4.06 [+2.92, +5.42] (5/5)	+1.32 [+0.96, +1.84] (5/5)
4B	VLM	+3.61 [+2.60, +4.84] (16/16)	+3.10 [+2.00, +4.45] (16/16)	+2.24 [+1.41, +3.19] (16/16)
4B	LLM	+4.79 [+2.86, +7.11] (10/10)	+4.32 [+2.28, +6.74] (10/10)	+2.78 [+1.52, +4.23] (10/10)
4B	Video	+2.08 [+1.70, +2.44] (5/5)	+1.48 [+1.16, +1.78] (5/5)	+0.90 [+0.48, +1.32] (5/5)
9B	VLM	+2.50 [+1.90, +3.18] (16/16)	+2.01 [+1.39, +2.77] (16/16)	+1.54 [+0.92, +2.29] (16/16)
9B	LLM	+4.06 [+2.16, +6.25] (10/10)	+3.63 [+1.65, +5.91] (10/10)	+2.61 [+1.36, +3.99] (10/10)
9B	Video	+1.62 [+1.38, +1.86] (5/5)	+1.14 [+0.86, +1.32] (5/5)	+0.78 [+0.30, +1.26] (5/5)

Table 21: Paired bootstrap 95% intervals over benchmark settings for the ZPPO − comparator macro-average delta (pp) of Tab. 3, Tab. 15, and Tab. 18, computed and printed as in Tab. 20.

Scale	Family	ZPPO − GRPO† Δ [95% CI] (Win)	ZPPO − GRPO†+BCQ Δ [95% CI] (Win)	ZPPO − GRPO†+NCQ Δ [95% CI] (Win)
0.8B	VLM	+4.79 [+3.45, +6.12] (16/16)	+1.64 [+1.13, +2.16] (16/16)	+4.05 [+2.84, +5.22] (16/16)
0.8B	LLM	+4.44 [+2.02, +7.77] (8/10)	+1.61 [+0.69, +2.92] (8/10)	+3.75 [+1.60, +6.80] (8/10)
0.8B	Video	+2.36 [+1.54, +3.30] (5/5)	+0.84 [+0.58, +1.22] (5/5)	+1.88 [+1.16, +2.78] (5/5)
2B	VLM	+2.76 [+1.89, +3.74] (16/16)	+1.21 [+0.79, +1.67] (16/16)	+1.86 [+1.26, +2.52] (16/16)
2B	LLM	+3.02 [+1.36, +5.19] (9/10)	+1.34 [+0.52, +2.42] (9/10)	+1.87 [+0.81, +3.35] (9/10)
2B	Video	+1.32 [+0.96, +1.84] (5/5)	+0.60 [+0.32, +1.02] (5/5)	+0.96 [+0.62, +1.54] (5/5)
4B	VLM	+2.24 [+1.41, +3.19] (16/16)	+1.31 [+0.86, +1.80] (16/16)	+1.06 [+0.69, +1.47] (16/16)
4B	LLM	+2.78 [+1.52, +4.23] (10/10)	+1.42 [+0.71, +2.22] (9/10)	+0.89 [+0.29, +1.63] (8/10)
4B	Video	+0.90 [+0.48, +1.32] (5/5)	+0.56 [+0.32, +0.80] (5/5)	+0.26 [+0.16, +0.36] (5/5)
9B	VLM	+1.54 [+0.92, +2.29] (16/16)	+0.94 [+0.62, +1.35] (16/16)	+0.44 [+0.31, +0.56] (16/16)
9B	LLM	+2.61 [+1.36, +3.99] (10/10)	+1.30 [+0.54, +2.16] (9/10)	+0.47 [+0.20, +0.79] (8/10)
9B	Video	+0.78 [+0.30, +1.26] (5/5)	+0.56 [+0.24, +1.00] (5/5)	+0.32 [+0.20, +0.44] (5/5)

and NCQ gives 68.7/77.3/85.7/88.7. Together they fill the combined instance cap of 96 at every audited step. Among the questions that pass this cap, the teacher succeeds on $82/206$, $56/232$, $31/257$, and $22/266$, or 39.8/24.1/12.1/8.3%. Each denominator is that scale's post-cap question count over the three audited steps. On these steps, every question that passes the cap has an NCQ instance, and every such question the teacher succeeds on also has a BCQ instance. Questions plus BCQ instances therefore add up to the cap over the three steps: $206+82 = 232+56 = 257+31 = 266+22 = 288 = 3\times 96$. BCQ candidates become scarce as the student scales up, since more of the remaining hard questions are ones the teacher also fails on. NCQ fills the freed slots.

BCQ audit. Tab. 22 covers 656/448/248/176 BCQ rollouts at 0.8B/2B/4B/9B. For each rollout, it checks whether the normalized final answer matches the teacher-derived candidate (*match-pos*), the student-derived candidate (*match-neg*), or neither (*other*). It splits the counts by graded correctness. In the BCQ instruction, "correct" and "wrong" refer to the reward of each candidate's source rollout (Appendix E.1). The table rows instead give the graded correctness of the student's new rollout. Graded-correct rollouts match the teacher-derived candidate at 78.4–90.7% across the four scales. Graded-wrong rollouts match the student-derived candidate at 77.4–97.4%. Each rate is a share of its own row's N. The *BCQ acc* column covers the audited steps only. This window differs from the first and last ten recorded steps of Tab. 32 and from the per-step curve of Fig. 27. Appendix I.4 varies what the candidate block contains.

Matches across the candidate labels. Two cells of Tab. 22 cross the labels. At 0.8B, 15 of the 450 graded-correct rollouts (3.3%) end on the answer of the student-derived candidate. At 9B, 3 of the 112 graded-wrong rollouts (2.7%) end on the answer of the teacher-derived candidate. The other six crossing cells are 0.0%. The audit compares the new rollout's normalized boxed string with the boxed answer of the compressed candidate, whose label is the reward of its source rollout (Appendix E.1). Such a match can arise when the compression changes a candidate's final answer, when the source rollout's reward is wrong, when the normalization merges two answers that the reward keeps apart, or when the judge scores two equivalent free-form answers differently.

NCQ audit. Step 4 of Algorithm 1 builds x_{NCQ} from the student's wrong rollouts $\{y_{\mathrm{S}}^{(-)}\}$ alone. The NCQ block holds their distinct parsed answers and their teacher-compressed rewrites, and no teacher rollout. Tab. 23 splits the NCQ rollouts at each scale into three shares of that scale's N. *Match-neg* is the share that repeats one of the listed wrong answers. *Match-pos* is the share that matches a separately logged teacher-derived string, which the prompt does not contain. *Other* is the rest. Match-neg is 82.7/2.2/0.7/0.2% at 0.8B/2B/4B/9B, match-pos 0.4/4.8/3.7/1.8%, and other 16.9/92.9/95.5/98.0%. At 0.8B, most NCQ rollouts repeat a listed answer and are graded wrong, so most 0.8B NCQ groups are all-wrong. The few correct rollouts are the sparse but recurring signal of Appendix G.3.

Two patterns across scales. NCQ's marginal contribution is largest at 0.8B (Tab. 21), where NCQ-prompt accuracy is lowest. At 1.2% of 1,648 rollouts, at most 20 of the 206 audited 0.8B NCQ groups hold a correct rollout. Appendix G.3 relates the contribution to these few non-trivial groups, many of which fall on questions with no other signal. Match-neg also falls from 82.7% at 0.8B to 2.2% at 2B, consistent with the 0.8B student copying answers from its context despite the instruction that they are wrong.

H TEACHER CAPABILITY ON THE SAME BENCHMARKS

We score the frozen 27B teacher (Qwen3.5-27B-FP8) on the 10 LLM, 16 VLM, and 5 Video benchmarks of the main tables. The teacher uses the same decoding and scoring as the students (Appendix C.2). Appendix H.3 also prints every point of the teacher-size sweep in Fig. 7.

H.1 ZPPO STUDENTS VS. THE 27B TEACHER

Tab. 24 compares the untrained base and the ZPPO student at each scale with the 27B teacher, in one block per benchmark family. The teacher row and the *Base* columns come from frozen checkpoints, and the ZPPO columns from trained ones (Appendix A).

Table 22: BCQ candidate-match audit over the last three logged sets of instances of one ZPPO run per scale (1,528 rollouts in total). Each rollout's normalized final `\boxed` string matches the teacher-derived candidate (*match-pos*), the student-derived candidate (*match-neg*), or neither (*other*). The rows split the rollouts by graded correctness, and *BCQ acc* is rollout accuracy over the same rollouts.

Scale	Subset	N	match-pos	match-neg	other	BCQ acc
0.8B	Correct	450	90.7%	3.3%	6.0%	68.6%
	Wrong	206	0.0%	90.3%	9.7%	
2B	Correct	271	79.7%	0.0%	20.3%	60.5%
	Wrong	177	0.0%	77.4%	22.6%	
4B	Correct	97	78.4%	0.0%	21.6%	39.1%
	Wrong	151	0.0%	97.4%	2.6%	
9B	Correct	64	84.4%	0.0%	15.6%	36.4%
	Wrong	112	2.7%	86.6%	10.7%	

Table 23: NCQ candidate-match audit over the last three logged sets of instances of one ZPPO run per scale (7,688 rollouts in total). *match-neg* is the share of rollouts whose normalized final answer repeats one of the listed wrong answers. *match-pos* is the share that matches a separately logged teacher-derived string, which the prompt does not contain. *other* is the rest, and *NCQ acc* is rollout accuracy over the same rollouts.

Scale	N	match-neg	match-pos	other	NCQ acc
0.8B	1,648	82.7%	0.4%	16.9%	1.2%
2B	1,856	2.2%	4.8%	92.9%	22.1%
4B	2,056	0.7%	3.7%	95.5%	35.2%
9B	2,128	0.2%	1.8%	98.0%	30.5%

Table 24: Macro average (Avg) on the 10 LLM, 16 VLM, and 5 Video benchmarks: the untrained base and the ZPPO student at four scales, and the 27B teacher. $\Delta = \text{ZPPO} - \text{Base}$. Bold marks the higher of Base and ZPPO in each row and family.

	LLM Avg			VLM Avg			Video Avg		
Model	Base	ZPPO	Δ	Base	ZPPO	Δ	Base	ZPPO	Δ
Qwen3.5-0.8B	25.2	**33.1**	**+7.9**	41.0	**50.3**	**+9.3**	48.3	**52.9**	**+4.6**
Qwen3.5-2B	45.3	**50.4**	**+5.1**	56.8	**62.0**	**+5.2**	60.6	**63.2**	**+2.6**
Qwen3.5-4B	61.6	**65.5**	**+3.9**	66.9	**70.9**	**+4.0**	71.5	**71.8**	**+0.3**
Qwen3.5-9B	64.9	**68.8**	**+3.9**	70.2	**73.0**	**+2.8**	73.8	**74.2**	**+0.4**
Qwen3.5-27B		71.8			75.5			77.4	

H.2 PER-BENCHMARK RESULTS: ZPPO STUDENTS VS. THE 27B TEACHER

Tab. 25, Tab. 26, and Tab. 27 give the per-benchmark scores behind Tab. 24 for the four ZPPO students and the 27B teacher. The ZPPO rows repeat Tab. 1, Tab. 2, Tab. 13, and Tab. 14. The 27B rows appear only here.

Table 25: Per-benchmark accuracy on the 10 LLM benchmarks: ZPPO students at four scales and the 27B teacher. Benchmark abbreviations: Appendix C.2.

Model	AIME25	AIME26	CEval	GPQA-D	HLE	IMO-AB	MMLU	MMLU-Pro	MMLU-Rd	MultiCh	Avg
Qwen3.5-0.8B + ZPPO	3.3	0.0	57.6	42.4	5.0	10.5	63.0	51.7	69.0	28.6	33.1
Qwen3.5-2B + ZPPO	26.7	40.0	74.0	60.6	5.1	29.5	76.4	68.2	82.9	40.6	50.4
Qwen3.5-4B + ZPPO	56.7	66.7	86.1	77.3	8.3	48.0	86.0	79.7	90.1	56.0	65.5
Qwen3.5-9B + ZPPO	**70.0**	73.3	87.2	78.8	9.8	49.8	88.1	82.0	91.7	56.8	68.8
Qwen3.5-27B	**70.0**	**76.7**	**91.2**	**83.3**	**16.0**	**51.3**	**91.6**	**86.3**	**94.3**	**57.5**	**71.8**

Table 26: Per-benchmark accuracy on the 16 VLM benchmarks: ZPPO students at four scales and the 27B teacher. Benchmark abbreviations: Appendix C.2.

Model	AI2D	BabyV	CharXiv	DynaM	EmbSp	InfoVQA	MVerse	MVision	MVista	MMMU^{Pro}	MM-Vet	OCR^{EN}	OCR^{ZH}	VisP	VBlind	WeMath	Avg
Qwen3.5-0.8B + ZPPO	76.5	13.9	63.9	31.1	71.5	75.3	59.3	29.2	73.2	37.6	59.9	42.5	18.7	35.0	44.7	71.7	50.3
Qwen3.5-2B + ZPPO	85.3	18.6	73.9	52.7	79.5	84.6	76.0	50.5	80.5	53.2	77.1	48.8	26.0	42.0	60.5	82.6	62.0
Qwen3.5-4B + ZPPO	89.6	27.8	82.8	63.9	82.4	88.8	85.7	72.2	85.4	67.1	79.8	54.6	33.0	54.5	74.9	92.6	70.9
Qwen3.5-9B + ZPPO	90.8	26.3	83.7	66.7	82.1	91.4	88.1	75.9	87.5	70.2	83.9	**56.7**	**35.6**	56.4	79.4	92.9	73.0
Qwen3.5-27B	**92.1**	**30.4**	**86.0**	**70.5**	**84.2**	**91.5**	**89.1**	**80.0**	**88.2**	**74.0**	**84.9**	55.7	33.5	**61.9**	**92.1**	**93.8**	**75.5**

Table 27: Per-benchmark accuracy on the 5 Video benchmarks: ZPPO students at four scales and the 27B teacher. VMME and VMME^{S} are the same Video-MME questions without and with subtitles. Each counts as one setting in the macro average. Benchmark abbreviations: Appendix C.2.

Model	MMVU	MVBench	VMME	VMME^{S}	VMMMU	Avg
Qwen3.5-0.8B + ZPPO	42.5	56.6	53.9	54.9	56.4	52.9
Qwen3.5-2B + ZPPO	59.0	62.7	62.4	63.9	68.1	63.2
Qwen3.5-4B + ZPPO	70.6	69.6	68.5	70.2	80.2	71.8
Qwen3.5-9B + ZPPO	74.2	71.7	70.2	72.8	82.1	74.2
Qwen3.5-27B	**75.7**	**74.6**	**76.1**	**76.0**	**84.7**	**77.4**

The 27B teacher is above all four ZPPO students on every family macro average. ZPPO's gain over the untrained base is largest at 0.8B in every family (Tab. 24). On the audited hard questions, the teacher has a correct rollout on 82 of 206 at 0.8B and on 22 of 266 at 9B (Appendix G.8).

H.3 WHAT THE TEACHER-SIZE FIGURE PLOTS

Fig. 7 plots the mean family gain Δ_{all} for each student scale and teacher size. Δ_{all} averages the three family gains over the untrained base of the same size:

$$\Delta_{\text{all}} \;=\; \tfrac{1}{3}\big(\Delta_{\text{LLM}} + \Delta_{\text{VLM}} + \Delta_{\text{Video}}\big), \qquad \Delta_{\text{family}} \;=\; \text{Avg}_{\text{family}}(\text{method}) - \text{Avg}_{\text{family}}(\text{Base}). \tag{11}$$

Δ_{all} weights the LLM, VLM, and Video families equally. The pooled LLM+Video gain of the boxed takeaway instead weights each setting equally (Eq. (10)). We train the 0.8B and 2B students with each of the 4B, 9B, and 27B teachers. Tab. 28 prints one Δ_{all} per pair.

With the 27B teacher, ZPPO's Δ_{all} is $\frac{1}{3}(7.9{+}9.3{+}4.6) = 7.3$ pp at 0.8B and $\frac{1}{3}(5.1{+}5.2{+}2.6) = 4.3$ pp at 2B, from the Δ cells of Tab. 24. The reference in Fig. 7 is GRPO†+NCQ. We train it with the 27B teacher only. Its Δ_{all} is $\frac{1}{3}(4.2{+}5.2{+}2.7) = 4.0$ pp at 0.8B and $\frac{1}{3}(3.2{+}3.3{+}1.7) = 2.7$ pp at 2B, from the *Avg* cells of Tab. 3 and Tab. 18. The figure draws it as a hatched band. The upper edge of the band marks the value.

The 4B and 9B teachers are the released Qwen3.5-4B and Qwen3.5-9B checkpoints. Their accuracy is therefore the untrained *Base* row of Tab. 13 and Tab. 14. The 27B teacher's accuracy is the last row of Tab. 24. Teacher accuracy rises from 4B to 9B to 27B on all three family macro averages.

A larger teacher widens ZPPO's gain at both student scales (Tab. 28). A 4B teacher brings it down to the GRPO†+NCQ level: $+4.4$ vs. $+4.0$ pp at 0.8B and $+2.8$ vs. $+2.7$ pp at 2B. The drop is steepest

Table 28: Mean family gain Δ_{all} of Eq. (11) for every point of the teacher-size sweep in Fig. 7. Each gain is in pp over the untrained base of the same size. The GRPO†+NCQ row uses the 27B teacher and marks the upper edge of the hatched reference band in the figure. Bold marks the largest gain in each ZPPO row.

Student	Recipe	4B teacher	9B teacher	27B teacher
Qwen3.5-0.8B	GRPO†+NCQ	–	–	+4.0
	ZPPO	+4.4	+5.1	**+7.3**
Qwen3.5-2B	GRPO†+NCQ	–	–	+2.7
	ZPPO	+2.8	+3.1	**+4.3**

at 0.8B, where ZPPO's gain is most BCQ-dominated (Sec. 4.3). The gain falls from $+7.3$ to $+4.4$ pp at 0.8B and from $+4.3$ to $+2.8$ pp at 2B.

In each teacher column of Tab. 28, that checkpoint supplies the correct teacher response in BCQ and compresses every BCQ and NCQ candidate (Tab. 11). Every other hyperparameter stays at its headline value. The judge of free-form answers stays the 27B checkpoint (Appendix I.11).

I Additional comparisons and robustness checks

This appendix collects nine controls and two diagnostics; none of the controls changes the ranking. The controls vary the budget (Appendix I.1), the replay (Appendix I.2), the zero-variance method (Appendix I.3), the candidate content (Appendix I.4), the judge (Appendix I.5), the teacher-side method (Appendix I.6), the reward (Appendix I.7), the policy objective (Appendix I.8), and the student backbone (Appendix I.9). The two diagnostics cover the plain and reformulated groups of one question (Appendix I.10) and the conditions the method needs (Appendix I.11). Apart from what each subsection changes, all rows share the initialization, corpus, schedule, and evaluation setup. Each row that adds a cited method differs from GRPO† only by that method's named term (Appendix B), and the term keeps the method's default coefficients and settings; we change none of them. RSTG† is the one row that takes only part of its method: it keeps the distillation term and leaves out the fine-tuning term (Appendix I.3). Every macro average covers the 10 LLM, 16 VLM, or 5 Video benchmarks, with two exceptions that the captions state. Tab. 33 averages over the settings its judge sweep scores, and Tab. 37 over the 3 LLM, 7 VLM, and 3 Video settings it lists.

I.1 Group-size and corpus-pass controls

ZPPO adds BCQ and NCQ rollouts, teacher rollouts, and candidate compression on top of GRPO†. We therefore sweep the budget of two baselines, GRPO† and On-Distill†, and two controls, Hint and Prefix (Tab. 29). Each of the four varies the student group size $G_{\text{S}} \in \{8, 16\}$ and the number of corpus passes (1 or 2), and ZPPO stays at its default ($G_{\text{S}} = 8$, 1 pass). Every other setting follows Tab. 11.

(i) Extra rollouts do not close the gap. Doubling the group size at one pass moves a macro average by $+0.0$ to $+0.5$ pp across the 24 swept (method, scale, family) cells, and no swept row reaches ZPPO. **(ii) At $G_{\text{S}} = 8$, a second corpus pass lowers LLM and Video.** On these two families, every (method, scale) cell moves by -0.1 to -1.1 pp, and VLM moves by at most 0.3 pp in either direction. For On-Distill†, the second pass also doubles the updates per run from 800 to 1,600, the same packing as the RL rows (Appendix E.2). It still lowers every macro average at both scales.

I.2 Extra replay from each method's own buffer

The sweep above adds budget to every question alike, and this control adds it only on hard questions. We raise GRPO†'s replay fraction ρ_{replay} from 0.25 to 0.50 (Tab. 30). The extra groups come from GRPO†'s own prompt replay buffer, and no teacher is involved. A GRPO† step then runs 384 new and up to 192 replayed groups. A ZPPO step at $\rho_{\text{replay}} = 0.25$ runs 384 new, up to 96 replayed, and at most 96 reformulated groups (Appendix D), so both are capped at 576 groups per rollout step. Each method admits the questions its own plain rollouts mark as hard, so the two buffers hold different questions. Every other setting follows Tab. 11.

Table 29: Group-size and corpus-pass sweep. Each compared method varies the student group size G_S and the number of corpus passes, and ZPPO stays at its default ($G_S = 8$, 1 pass). The $(8, 1)$ row of each block is that method's row in Tab. 1, Tab. 2, or Tab. 4.

Method	(G_S, passes)	Qwen3.5-0.8B			Qwen3.5-2B		
		LLM	VLM	Video	LLM	VLM	Video
GRPO†	$(8, 1)$	28.7	45.5	50.5	47.4	59.2	61.9
	$(16, 1)$	29.0	46.0	50.6	47.6	59.7	62.0
	$(8, 2)$	28.6	45.6	50.3	47.2	59.4	61.7
	$(16, 2)$	28.9	45.7	50.7	47.4	59.6	61.9
On-Distill†	$(8, 1)$	23.2	42.3	45.8	43.8	58.0	59.2
	$(16, 1)$	23.6	42.8	46.0	44.0	58.4	59.2
	$(8, 2)$	22.4	42.2	45.1	43.0	57.9	58.6
	$(16, 2)$	22.9	42.5	45.4	43.4	58.2	58.8
GRPO†+Hint	$(8, 1)$	28.8	47.2	50.0	47.6	60.4	61.6
	$(16, 1)$	29.1	47.6	50.3	47.9	60.9	61.8
	$(8, 2)$	28.1	47.3	49.4	47.0	60.5	61.1
	$(16, 2)$	28.6	47.4	49.7	47.4	60.6	61.3
GRPO†+Prefix	$(8, 1)$	27.1	45.5	49.1	46.3	59.5	61.0
	$(16, 1)$	27.5	45.8	49.1	46.7	59.8	61.1
	$(8, 2)$	26.0	45.2	48.1	45.3	59.3	60.1
	$(16, 2)$	26.5	45.4	48.3	45.8	59.4	60.4
ZPPO	$(8, 1)$	**33.1**	**50.3**	**52.9**	**50.4**	**62.0**	**63.2**

Table 30: Replay control: GRPO† and ZPPO at two replay fractions ρ_{replay}. At $\rho_{\text{replay}} = 0.50$, a GRPO† step is capped at 576 groups, the same cap as ZPPO at 0.25. Bold marks the column maximum.

Method	ρ_{replay}	Qwen3.5-0.8B			Qwen3.5-2B		
		LLM	VLM	Video	LLM	VLM	Video
GRPO†	0.25	28.7	45.5	50.5	47.4	59.2	61.9
GRPO†	0.50	29.1	45.7	50.4	47.6	59.9	62.1
ZPPO	0.25	33.1	50.3	52.9	50.4	62.0	63.2
ZPPO	0.50	**33.8**	**50.5**	**53.1**	**50.5**	**62.3**	**63.5**

Extra replay does not close the gap. It moves GRPO†'s LLM/VLM/Video averages by $+0.4/+0.2/-0.1$ pp at 0.8B and $+0.2/+0.7/+0.2$ pp at 2B. ZPPO is above GRPO† by $+4.4/+4.8/+2.4$ pp at 0.8B. Doubling GRPO†'s group size instead moves the same 6 cells by at most $+0.5$ pp (Tab. 29). Plain replay gives an all-wrong group no gradient, since at $\bar{r}_x = 0$ the group has zero advantage on that visit. A partially solved replayed question ($\bar{r}_x \in \{0.125, \ldots, 0.375\}$ at $G_S = 8$) still gives a gradient. BCQ and NCQ apply to both kinds of hard question. A reformulated group has nonzero advantage whenever its own rollouts include both a correct and a wrong one, even when the plain rollouts on the same question are all wrong (Appendix G.2).

Raising ZPPO's own ρ_{replay} to 0.50 lifts its full replay count from 96 to 192 questions and its per-step cap from 576 to 672 groups. It moves LLM/VLM/Video by $+0.7/+0.2/+0.2$ pp at 0.8B and $+0.1/+0.3/+0.3$ pp at 2B. The headline ZPPO row stays at $\rho_{\text{replay}} = 0.25$, whose cap matches GRPO† at 0.50.

I.3 The zero-variance line: the same diagnosis, a different intervention

RL-ZVP (Le et al., 2026), LENS (Feng et al., 2025b), ReLIFT (Ma et al., 2026), and RSTG (Han et al., 2026) start from the same diagnosis as ZPPO: an all-wrong group has zero advantage. They change the loss or the target, never the prompt (Appendix B). We add each named term to the GRPO† pipeline (Tab. 31), and every other setting follows Tab. 11.

In GRPO† and ZPPO, an all-wrong group keeps zero advantage (Appendix D). RL-ZVP and LENS are *teacher-free*. Both replace that zero advantage with a signal from the student's own rollouts, so the group still contributes to the gradient. **RL-ZVP** shapes the advantage from the student's token-level entropy, and **LENS** penalizes a wrong answer more as the student's confidence in it grows. ReLIFT and RSTG instead put teacher targets in the loss on those questions. **ReLIFT** interleaves online supervised fine-tuning on teacher solutions for the hardest questions. **RSTG** applies on-policy distillation only on all-wrong groups. It weights each sample by the teacher's confidence and distills only selected tokens: those with high student entropy or a large teacher–student divergence. It also fine-tunes on teacher-generated correct trajectories. This row keeps the distillation term with its sample weights and token selection, and leaves out the fine-tuning. ZPPO also uses the teacher, but only as prompt text, never as a gradient target.

Table 31: Zero-variance adaptations on the GRPO† pipeline. Each adds a named signal or teacher target and leaves the prompt unchanged; † denotes augmentation with the prompt replay buffer. *Teacher target in loss* records whether the row puts a teacher target in the loss.

Method	Teacher target in loss	Qwen3.5-0.8B			Qwen3.5-2B		
		LLM	VLM	Video	LLM	VLM	Video
Base	–	25.2	41.0	48.3	45.3	56.8	60.6
GRPO†	No	28.7	45.5	50.5	47.4	59.2	61.9
On-Distill†	Yes	23.2	42.3	45.8	43.8	58.0	59.2
RL-ZVP†	No	29.0	45.8	50.7	47.5	59.5	62.0
LENS†	No	29.3	46.1	50.9	47.7	59.8	62.2
ReLIFT†	Yes	28.1	46.9	49.6	47.1	60.4	61.5
RSTG†	Yes	27.5	47.6	49.1	46.8	60.7	61.0
ZPPO	No	**33.1**	**50.3**	**52.9**	**50.4**	**62.0**	**63.2**

The two teacher-free rows are above GRPO† in all 12 of their cells, by $+0.1$ to $+0.6$ pp. The two teacher-target rows split by family. They are above GRPO† on VLM by $+1.2$ to $+2.1$ pp, but below it on LLM by -0.3 to -1.2 pp and on Video by -0.4 to -1.4 pp. None of the four changes the ranking.

I.4 BCQ candidate-content controls

A default BCQ candidate block holds a teacher-compressed trace that ends in its boxed final answer. We vary the candidate content with three variants on top of GRPO† (Tab. 32). The first drops the wrong student response and keeps the compressed correct teacher response as the only candidate. It appends that one `<candidate>` block without the BCQ instruction, since that instruction describes a correct–wrong pair. The prompt therefore carries a full correct solution, a stronger cue than the

answer-free hint of Tab. 4. The *boxed-answer-masked* variant masks the boxed span and keeps the reasoning. The *final-answer-only* variant drops the reasoning and keeps the answer. These two keep the BCQ template of Fig. 17, the shared compression prompt, and the shuffled, anonymized candidate pair.

Table 32: BCQ candidate-content controls at 2B on top of GRPO†. Δ is each row minus the GRPO† row. *BCQ acc* is the rollout accuracy on the reformulated prompt over the first ten and the last ten recorded training steps of one logged run.

BCQ candidate	Qwen3.5-2B			Δ vs. GRPO†			BCQ acc
	LLM	VLM	Video	LLM	VLM	Video	
GRPO† (no candidate)	47.4	59.2	61.9	–	–	–	–
+BCQ (default)	49.1	60.8	62.6	**+1.7**	**+1.6**	**+0.7**	0.23→0.56
+BCQ without wrong student response	46.3	57.6	60.2	**−1.1**	**−1.6**	**−1.7**	0.88→0.95
+boxed-answer-masked candidates	48.5	60.0	62.0	**+1.1**	**+0.8**	**+0.1**	0.22→0.55
+final-answer-only candidates	47.1	58.9	62.0	**−0.3**	**−0.3**	**+0.1**	0.45→0.48
ZPPO	**50.4**	**62.0**	**63.2**	**+3.0**	**+2.8**	**+1.3**	0.24→0.53

(i) A lone correct candidate turns BCQ's gain into a loss. With the correct teacher response as the only candidate, the row is $-1.1/-1.6/-1.7$ pp against GRPO† on LLM/VLM/Video, while default BCQ is $+1.7/+1.6/+0.7$. The correct response does not help on its own. This variant also drops the pair instruction, so it does not separate the effect of the wrong candidate from that of the instruction. **(ii) Dropping the reasoning removes BCQ's gain.** With only the final answer in each block, the row is $-0.3/-0.3/+0.1$ pp against GRPO†. Masking the boxed span and keeping the reasoning still leaves $+1.1/+0.8/+0.1$ pp. **(iii) Accuracy on the reformulated prompt does not track the benchmark gain.** Without the wrong response, this accuracy is already 0.88 over the first recorded steps and ends at 0.95. Yet this row is the lowest in the table. The final-answer-only block shows just the two answers, one correct and one wrong, so copying one of them at random gives accuracy $1/2$. That block stays near this level, $0.45 \to 0.48$. Default BCQ and the masked block start well below $1/2$ and end above it, $0.23 \to 0.56$ and $0.22 \to 0.55$. These two are the only variants above GRPO† on all three families. They are also the only ones whose accuracy starts well below $1/2$ and ends above it. **(iv) Handing over more of the solution lowers the row.** Against GRPO† at 2B, Hint (Tab. 4) gives an answer-free cue and is $+0.2/+1.2/-0.3$ pp on LLM/VLM/Video. Prefix gives the opening 40% of a correct teacher rollout and is $-1.1/+0.3/-0.9$ pp. The correct response alone is $-1.1/-1.6/-1.7$ pp. It is the lowest of the three on VLM and Video and ties with Prefix on LLM. Prefix also opens the response and is not compressed, so we read this as a trend, not a controlled sweep. On VLM, the correct response alone (57.6) lands within 0.4 pp of the four supervised and distillation rows at 2B: Off-Distill† (57.3), Correction-SFT† (57.6), Error-reflection-SFT† (57.9), and On-Distill† (58.0). On LLM and Video it stays above them, 46.3 against 43.1–44.6 and 60.2 against 58.6–59.6. **(v) BCQ hands over the same solution but does not say which candidate it is.** At 2B, 79.7% of the graded-correct BCQ rollouts end on the teacher-derived answer (Tab. 22). To reach it, the student must pick the right candidate, and BCQ starts at 0.23, below the $1/2$ that copying one at random gives. Without the wrong response, the prompt holds only the correct candidate, so there is nothing to choose between. Accuracy on that prompt is already 0.88 over the first recorded steps. Placing a correct teacher solution in the prompt is therefore not enough by itself. BCQ's gain appears only with the wrong candidate and the pair instruction in place.

I.5 Robustness to the judge model

The judge for free-form answers is Qwen3.5-27B-FP8, the same checkpoint as the frozen teacher. We change only the judge and keep the scored student responses fixed. The shared judge template decides $8/10$ LLM, $11/16$ VLM, and $4/5$ Video settings whenever the deterministic scorer gives no credit. Of the other eight settings, three are scored deterministically only, and five use a benchmark-specific judge prompt (Appendix C.2). Neither group enters the sweep. We recompute each macro average over the swept settings under the default judge and seven others (Tab. 33). All eight judges use the instruction text, runtime fields, strict-JSON parser, and fallback of Fig. 9.

(i) Seven other judges leave the ordering intact. All eight give ZPPO $>$ GRPO† $>$ On-Distill† on every family at both scales. The population standard deviation over the eight judges is at most 0.6 pp for every method and family. **(ii) The default judge is not the most generous one.** At 0.8B

Table 33: Robustness to the judge model. Each average covers the $8/10$ LLM, $11/16$ VLM, and $4/5$ Video settings that the shared judge template decides. Only the judge changes across rows, and the scored responses stay fixed. The first row of each block is the default judge, and we query the other seven through their providers' APIs in one evaluation window.

Judge	ZPPO			GRPO†			On-Distill†		
	LLM	VLM	Video	LLM	VLM	Video	LLM	VLM	Video
Qwen3.5-0.8B student									
Qwen3.5-27B-FP8 (default)	37.2	49.4	55.5	32.8	43.5	52.8	26.8	40.4	48.1
Gemini 3.1 Pro	37.0	48.8	55.6	32.6	43.0	52.7	26.5	39.6	48.1
Gemini 3.5 Flash	38.0	50.0	55.9	33.4	43.9	53.2	27.7	40.9	48.7
Claude Sonnet 5	37.5	48.7	55.7	33.1	43.0	53.0	27.1	39.5	48.3
GPT-5.6 Luna	37.6	49.5	55.6	33.3	43.6	52.9	27.2	40.6	48.2
GPT-5 Nano	37.8	49.8	55.9	33.4	43.7	53.3	27.5	41.0	48.6
Kimi-K2.6	36.8	48.6	55.4	32.5	42.8	52.6	26.1	39.4	47.8
GLM 5.2	37.3	48.9	55.8	32.9	43.1	53.2	26.9	40.0	48.4
Mean ± pop. std	$37.4_{\pm 0.4}$	$49.2_{\pm 0.5}$	$55.7_{\pm 0.2}$	$33.0_{\pm 0.3}$	$43.3_{\pm 0.4}$	$53.0_{\pm 0.2}$	$27.0_{\pm 0.5}$	$40.2_{\pm 0.6}$	$48.3_{\pm 0.3}$
Qwen3.5-2B student									
Qwen3.5-27B-FP8 (default)	57.3	61.9	64.3	54.0	58.7	63.2	50.0	57.8	60.9
Gemini 3.1 Pro	56.7	61.7	63.9	53.5	58.6	62.8	49.3	57.4	60.4
Gemini 3.5 Flash	57.9	62.6	64.5	54.4	59.3	63.3	50.6	58.6	61.2
Claude Sonnet 5	56.5	61.8	63.8	53.3	58.7	62.7	49.1	57.5	60.4
GPT-5.6 Luna	57.4	62.4	64.2	54.1	59.1	63.1	50.1	58.3	60.8
GPT-5 Nano	57.8	62.5	64.4	54.4	59.2	63.3	50.6	58.6	61.1
Kimi-K2.6	56.4	61.6	63.7	53.3	58.5	62.7	49.0	57.3	60.0
GLM 5.2	56.8	62.3	64.0	53.6	59.0	62.9	49.5	58.2	60.6
Mean ± pop. std	$57.1_{\pm 0.5}$	$62.1_{\pm 0.4}$	$64.1_{\pm 0.3}$	$53.8_{\pm 0.4}$	$58.9_{\pm 0.3}$	$63.0_{\pm 0.2}$	$49.8_{\pm 0.6}$	$58.0_{\pm 0.5}$	$60.7_{\pm 0.4}$

on LLM, it scores ZPPO at 37.2, against a mean of 37.4. **(iii) All eight judges share one template.** They use the same fallback of Fig. 9, which lets the judge look inside the reasoning when no final answer parses, so the sweep isolates the choice of judge. Appendix I.11 reports the settings that do not use this fallback.

I.6 Teacher-side alternatives: self-distillation and hybrid RL+distillation

Tab. 34 groups its rows by where teacher knowledge enters training, as Sec. 2 does for prior work. Each added row puts a named term on the GRPO† pipeline. The external-teacher rows use the frozen teacher, Qwen3.5-27B-FP8, and every other setting follows Tab. 11.

The two self-teacher rows build the teacher from the student itself, so no external model enters them. **OPSD** (Zhao et al., 2026) makes the student its own teacher. As teacher, the same model sees extra context that the student's prompt lacks. The student then matches this teacher's per-token distributions on its own rollouts. **SDPO** (Hübotter et al., 2026) conditions a self-teacher on rollout feedback and distills it back into the student on the student's own rollouts. Besides On-Distill†, three hybrid RL+distillation rows route an external teacher back through the gradient. **KDRL** (Xu et al., 2025) adds a reverse-KL term toward the teacher on the student's rollouts. **RLKD** (Xu et al., 2026) adds a reward derived from the teacher's reasoning structure. **RLAD** (Zhang et al., 2026b) replaces the policy ratio with a trust-region ratio anchored to a mixture of the old student policy and the teacher.

On-Distill†, OPSD†, and SDPO† all land below the untrained base on LLM and Video at both scales. The two self-teacher rows are -2.2 to -3.7 pp below it in those 8 (method, scale, family) cells. KDRL† and RLKD† stay above the base on all three families but split against GRPO†. They are above it on VLM by $+0.2$ to $+0.8$ pp, but below it on LLM by -0.6 to -1.2 pp and on Video by -0.6 to -0.9 pp. RLAD† is above GRPO† in all 6 cells, by $+0.3/+0.8/+0.2$ pp at 0.8B and $+0.5/+1.1/+0.2$ pp at 2B. Hybrid RL+distillation is at most $+1.1$ pp above GRPO†.

I.7 Alternative multimodal RL rewards

BCQ and NCQ change the prompt and leave the reward alone. Each row of Tab. 35 instead adds a named reward or credit-assignment term to the GRPO† pipeline, and every other setting follows Tab. 11. Two rows add a term beside the correctness reward. **Perception-R1** (Xiao et al., 2026) adds

Table 34: Teacher-side adaptations on the GRPO† pipeline, grouped by where teacher knowledge enters training; † denotes augmentation with the prompt replay buffer. The hybrid RL+distillation rows add a reverse-KL term, a reasoning-structure reward, or a trust-region ratio anchored to a student–teacher mixture. The untrained *Base* and GRPO† are the reference rows.

	Qwen3.5-0.8B			Qwen3.5-2B		
Method	LLM	VLM	Video	LLM	VLM	Video
Base	25.2	41.0	48.3	45.3	56.8	60.6
Self-teacher (no external model)						
+OPSD†	21.8	41.2	44.6	42.4	57.0	58.1
+SDPO†	22.1	40.9	44.9	42.7	56.7	58.4
External-teacher target, term, or reward in the loss						
+On-Distill†	23.2	42.3	45.8	43.8	58.0	59.2
+KDRL†	27.8	46.0	49.8	46.8	60.0	61.3
+RLKD†	27.5	45.7	49.6	46.5	59.6	61.1
+RLAD†	29.0	46.3	50.7	47.9	60.3	62.1
No teacher						
+GRPO†	28.7	45.5	50.5	47.4	59.2	61.9
Teacher in the prompt						
+ZPPO	**33.1**	**50.3**	**52.9**	**50.4**	**62.0**	**63.2**

a visual-perception reward that an LLM judges against textual annotations of the image. **GRPO-CARE** (Chen et al., 2026c) adds an adaptive consistency bonus. The third row changes credit assignment: **SRPO** (Jiang & Zou, 2026) keeps the reward and turns the group-relative advantage into role-aware token weights for perception and reasoning tokens.

Table 35: Multimodal reward and credit-assignment adaptations on the GRPO† pipeline. The three rows add a perception reward, a consistency bonus, or role-aware token weights; † denotes augmentation with the prompt replay buffer.

	Qwen3.5-0.8B			Qwen3.5-2B		
Method	LLM	VLM	Video	LLM	VLM	Video
GRPO†	28.7	45.5	50.5	47.4	59.2	61.9
Perception-R1†	29.1	47.0	50.7	47.5	60.3	62.0
GRPO-CARE†	29.2	46.2	51.4	47.8	59.9	62.7
SRPO†	29.0	47.2	50.9	47.6	60.6	62.2
ZPPO	**33.1**	**50.3**	**52.9**	**50.4**	**62.0**	**63.2**

The three rows are above GRPO† in all 18 (method, scale, family) cells, by $+0.1$ to $+1.7$ pp. At both scales, Perception-R1† and SRPO† gain most on VLM, and GRPO-CARE† gains most on Video. None of the three changes the ranking.

I.8 Compatibility with alternative policy objectives

We keep BCQ, NCQ, and the prompt replay buffer fixed and change only the policy objective of Step 6 in Algorithm 1. Tab. 36 reports three other objectives. Each uses its named policy term on the GRPO† pipeline, and every other setting follows Tab. 11. **GSPO** (Zheng et al., 2025) replaces the token-level importance ratio with one sequence-level ratio per rollout. **CISPO** (Chen et al., 2025a) clips the importance weight instead of the update. **DISPO** (Karaman et al., 2026) sets the upper and lower clipping of the importance weight separately for correct and for wrong responses, which gives four clipping regimes. The reference is GRPO with the DAPO ingredients, the Step 6 objective in every other table. Each other objective alone is above GRPO† in all 6 of its (scale, family) cells, by $+0.5$ to $+1.2$ pp at 0.8B and $+0.6$ to $+1.4$ pp at 2B. We therefore measure ZPPO's gain within each objective.

ZPPO gives a positive Δ on all three families at both scales under each of the four objectives, from $+1.2$ to $+5.0$ pp across the 24 (objective, scale, family) cells. The three other objectives keep the family ordering of the GRPO† row: the largest Δ is on VLM at 0.8B, and the smallest is on Video at 2B.

Table 36: ZPPO with GSPO (Zheng et al., 2025), CISPO (Chen et al., 2025a), or DISPO (Karaman et al., 2026) as the Step 6 objective, with BCQ, NCQ, and the prompt replay buffer fixed. Δ is ZPPO minus the same objective alone.

Scale	Objective	Objective alone			ZPPO on that objective			Δ		
		LLM	VLM	Video	LLM	VLM	Video	LLM	VLM	Video
0.8B	GRPO† (default)	28.7	45.5	50.5	33.1	50.3	52.9	**+4.4**	**+4.8**	**+2.4**
	GSPO†	29.4	46.1	51.5	33.7	50.8	53.9	**+4.3**	**+4.7**	**+2.4**
	CISPO†	29.9	46.3	51.0	34.5	51.3	53.5	**+4.6**	**+5.0**	**+2.5**
	DISPO†	29.7	46.7	51.2	34.2	51.4	53.8	**+4.5**	**+4.7**	**+2.6**
2B	GRPO† (default)	47.4	59.2	61.9	50.4	62.0	63.2	**+3.0**	**+2.8**	**+1.3**
	GSPO†	48.1	60.0	62.9	51.0	62.7	64.1	**+2.9**	**+2.7**	**+1.2**
	CISPO†	48.7	60.2	62.5	51.9	63.1	63.9	**+3.2**	**+2.9**	**+1.4**
	DISPO†	48.4	60.6	62.7	51.4	63.2	64.2	**+3.0**	**+2.6**	**+1.5**

I.9 Two student backbones outside Qwen3.5

Appendix I.8 swaps the policy objective and keeps the Qwen3.5 student. Here we run the recipe on Ovis2.5-2B (Lu et al., 2025) and InternVL3.5-2B (Wang et al., 2025c) with the same Qwen3.5-27B-FP8 teacher (Tab. 37). Both differ from the Qwen3.5 student in the vision encoder, the modality connector, and the training recipe. Ovis2.5 pairs a native-resolution ViT initialized from SigLIP2 with a learnable visual embedding table in place of an MLP projector. InternVL3.5 pairs InternViT-300M with an MLP projector. Both build on the Qwen3-1.7B language model, so the pair varies the vision stack and the multimodal recipe and holds the language model fixed. ZPPO needs no shared output vocabulary, because the teacher reaches the student only as prompt text and no teacher distribution is matched. Evaluation covers 3 of our 10 LLM, 7 of our 16 VLM, and 3 of our 5 Video settings. It uses the unchanged evaluation setup of Appendix C.2, the same setup that measures the matching Qwen3.5-2B cells of Tab. 1 and Tab. 2. That setup includes `max-model-len=262144`, which only caps the sequence length. We add no RoPE scaling to the 40,960 positions that both language models are configured for. Both released checkpoints are already RL-tuned: Ovis2.5 with DPO followed by GRPO, and InternVL3.5 with cascaded MPO and GSPO. ZPPO's gain here therefore comes on top of an existing RL stage.

Table 37: ZPPO on Ovis2.5-2B (Lu et al., 2025) and InternVL3.5-2B (Wang et al., 2025c) with the Qwen3.5-27B-FP8 teacher, over 3 of our 10 LLM, 7 of our 16 VLM, and 3 of our 5 Video settings (abbreviations in Appendix C.2). Bold marks the column maximum inside each backbone block, and Δ is ZPPO minus GRPO†. Tab. 38 gives paired bootstrap intervals over settings for these deltas.

		3 LLM Benchmarks				7 VLM Benchmarks								3 Video Benchmarks			
Backbone	Method	AIME25	CEval	MMLU	Avg	AI2D	DynaM	MVerse	MVision	MVista	MM-Vet	WeMath	Avg	MVBench	VMME	VMME^S	Avg
Ovis2.5-2B	Base	40.0	60.3	60.9	53.7	85.5	26.5	64.8	37.4	81.4	67.9	50.8	59.2	66.7	60.2	63.6	63.5
	+GRPO†	43.3	61.2	62.2	55.6	85.2	29.6	67.0	40.7	81.9	68.8	53.1	60.9	68.3	62.3	65.3	65.3
	+ZPPO	**46.7**	**62.6**	**64.1**	**57.8**	**87.6**	**32.8**	**69.3**	**43.5**	**83.2**	**71.3**	**56.1**	**63.4**	**70.2**	**64.2**	**66.9**	**67.1**
	Δ	**+3.4**	**+1.4**	**+1.9**	**+2.2**	**+2.4**	**+3.2**	**+2.3**	**+2.8**	**+1.3**	**+2.5**	**+3.0**	**+2.5**	**+1.9**	**+1.9**	**+1.6**	**+1.8**
InternVL3.5-2B	Base	43.3	61.0	61.9	55.4	78.8	31.5	53.4	42.8	71.8	71.7	48.5	56.9	65.9	58.4	61.9	62.1
	+GRPO†	46.7	62.1	63.1	57.3	80.2	34.7	56.0	45.7	71.4	72.8	50.8	58.8	67.7	59.9	63.2	63.6
	+ZPPO	**50.0**	**63.5**	**64.7**	**59.4**	**81.5**	**38.6**	**58.8**	**49.4**	**74.0**	**74.3**	**53.9**	**61.5**	**69.3**	**62.0**	**64.9**	**65.4**
	Δ	**+3.3**	**+1.4**	**+1.6**	**+2.1**	**+1.3**	**+3.9**	**+2.8**	**+3.7**	**+2.6**	**+1.5**	**+3.1**	**+2.7**	**+1.6**	**+2.1**	**+1.7**	**+1.8**

ZPPO is above GRPO† by +2.2/+2.5/+1.8 pp on the LLM/VLM/Video subsets for Ovis2.5-2B and by +2.1/+2.7/+1.8 pp for InternVL3.5-2B. On the same subsets of Tab. 1 and Tab. 2, ZPPO on Qwen3.5-2B is above GRPO† by +0.9/+3.5/+1.2 pp. Both backbones therefore show a smaller VLM gain and larger LLM and Video gains than Qwen3.5-2B. ZPPO is above GRPO† on all 13 settings of both backbones. GRPO† falls below the base on one setting per backbone: AI2D on Ovis2.5-2B and MVista on InternVL3.5-2B. ZPPO is above the base on both of those settings.

Tab. 38 resamples these deltas over the reported settings, as in Appendix G.7. The ZPPO − GRPO† interval excludes zero in all six (backbone, family) blocks.

Table 38: Paired bootstrap 95% intervals over benchmark settings for the ZPPO − comparator macro-average delta (pp) of Tab. 37, following Appendix G.7. We compute the resample distribution of the mean exactly instead of sampling it. It covers $1{,}716$ weighted count vectors for a seven-setting block and 10 for a three-setting block. For a three-setting block, the interval spans the smallest to the largest per-setting delta. Δ is the full-sample paired mean, and *Win* counts the settings on which ZPPO is above the comparator.

Backbone	Family	ZPPO − Base Δ [95% CI] (Win)	ZPPO − GRPO† Δ [95% CI] (Win)
Ovis2.5-2B	VLM	+4.21 [+2.96, +5.43] (7/7)	+2.50 [+2.04, +2.89] (7/7)
Ovis2.5-2B	LLM	+4.07 [+2.30, +6.70] (3/3)	+2.23 [+1.40, +3.40] (3/3)
Ovis2.5-2B	Video	+3.60 [+3.30, +4.00] (3/3)	+1.80 [+1.60, +1.90] (3/3)
InternVL3.5-2B	VLM	+4.57 [+3.17, +5.94] (7/7)	+2.70 [+1.99, +3.36] (7/7)
InternVL3.5-2B	LLM	+4.00 [+2.50, +6.70] (3/3)	+2.10 [+1.40, +3.30] (3/3)
InternVL3.5-2B	Video	+3.33 [+3.00, +3.60] (3/3)	+1.80 [+1.60, +2.10] (3/3)

I.10 Reformulated prompts in training, the original prompt at evaluation

The original question and the BCQ or NCQ prompt built from it enter the same rollout step (Algorithm 1). The plain rollouts form one group, and the reformulated prompt adds a further group under the same reward and the same parameters. Evaluation uses the prompt of the plain group: the question text and image, followed by the instruction in Fig. 8. Every reported score is therefore measured on a candidate-free prompt. The candidate block serves only as context for the added group. Correction-SFT† is the one departure from that shared prompt: it keeps its training-time `<correct>` tag at evaluation, the condition it trains under (Appendix E.2).

Training and evaluation prompts differ only in the candidate block. Every prompt, plain or reformulated, ends with the same instruction (Fig. 8), and every rollout is graded by the same reward in the same think-then-boxed format. A reformulated prompt adds only the candidate block before that instruction. Decoding does differ: training rollouts sample at temperature 1.0, and evaluation uses the configuration of Appendix C.2. The plain group of the same question stays in the same rollout step. BCQ and NCQ groups make up at most one group in five of a step, and one in six once the buffer supplies the full replay count (Appendix D). At least four in five groups of every step are therefore trained on a prompt in the evaluation format.

Prior guidance methods train across the same gap. Guide-GRPO, NuRL, HINT, and HiLL write their hints from a reference solution or answer (Nath et al., 2025; Chen et al., 2026a; Wang et al., 2026; Xia et al., 2026). BREAD and Prefix-RFT cut their prefixes from expert solutions (Zhang et al., 2025d; Huang et al., 2025b). None of these sources exists at test time, so these methods train with the guidance and evaluate without it. HiLL, BREAD, and Prefix-RFT train the guided rollouts under the guided context, as ZPPO trains each reformulated group under its own prompt. Guide-GRPO and HINT instead correct the gradient toward the original prompt. Our Hint and Prefix controls train under the guided context (Appendix E.2), so Tab. 4 compares them with BCQ under the same gap.

When the plain group is all-wrong, it has zero advantage. A BCQ or NCQ group built from that question then uses a second prompt that still contains the original question. The student samples the response, the reward grades it, and the update changes the shared parameters under that prompt. GRPO† keeps the plain group and the buffer but drops these groups. On entries admitted at 0% rollout accuracy at 2B, later plain rollouts graduate 432 entries under ZPPO and 73 under GRPO† (Appendix G.2).

(i) Graduation is measured on plain rollouts. Step 6 of Algorithm 1 refreshes the buffer with the mean rollout accuracy $\bar{r}_x$ of the plain rollouts only. An entry admitted at 0% therefore graduates when a later plain group moves from $0/8$ to at least $4/8$ (Appendix G.2). A $0/8$ group does not mean the student cannot solve the question, so some entries graduate on resampling alone. GRPO† runs the same buffer without BCQ and NCQ and graduates 4% of its re-sampled entries admitted at 0% (73/2035), against 28% for ZPPO (432/1566). The gap is consistent with the reformulated groups lifting the plain rollouts on the same question. The two buffers hold different questions, however, so the gap is not a per-question estimate. Because BCQ shows a correct answer to the same question,

a later plain success can also reflect that answer, so we treat graduation as a training-side signal. Point (iii) gives the evidence on the held-out benchmarks.

(ii) Accuracy on the reformulated prompt alone does not explain the gain. With final-answer-only candidates, this accuracy stays near $1/2$, the rate of copying one of them at random, yet the benchmark gain disappears. With the correct response as the only candidate, this accuracy reaches 0.95, yet all three macro averages fall below GRPO† (Appendix I.4). Doing well on a reformulated prompt therefore does not carry over to the original question by itself. BCQ does not say which of its two candidates is correct, so the student has to work it out. Point (iii) measures the result on the original question.

(iii) The gain holds on the original question. Over the four scales' 124 (scale, setting) cells, ZPPO is above GRPO† in 121, level in 3, and below in 0 (Appendix F.1).

(iv) Extra replay without reformulation does not close the gap. Raising GRPO†'s replay fraction from 0.25 to 0.50 re-exposes its hard questions without reformulation. It moves the three macro averages by at most $+0.4$ pp at 0.8B (Appendix I.2).

I.11 Scope of the reported measurements

ZPPO needs a binary correctness reward and a teacher that solves some of the student's hard questions. On questions that both teacher and student fail, BCQ has no candidate and NCQ carries the signal alone. Such questions become more common with student scale (Appendix G.8). NCQ alone lifts GRPO† on every family at every scale but stays below the full recipe (Appendix G.4). At 9B, where the measured teacher-success rate is lowest, ZPPO is still above GRPO† in all 31 settings (Appendix F.1). Appendix E.3 gives the wall-clock time that ZPPO adds.

One Qwen3.5-27B-FP8 checkpoint serves as the BCQ teacher, as the compressor of every BCQ and NCQ candidate, and as the judge that assigns the binary reward on free-form answers. Every method is scored through that same reward path (Appendix E.1). On the evaluation side, the judge sweep of Appendix I.5 leaves the ordering intact.

The controlled comparison is ZPPO $-$ GRPO†, which varies BCQ and NCQ alone (Appendix G.7). The untrained *Base* row is the comparator for the gains in Fig. 1 and Fig. 7, and elsewhere only where the text or a caption says so.

Reading the results. Eight points bear on how the results are read.

(i) The judge fallback. The shared judge template lets the judge read the reasoning when a response gives no final answer (Fig. 9). The judge sees the full response only when no `\boxed{}` span parses. Otherwise, it sees only the extracted answer (Appendix C.2). The rule applies to every method and to the untrained base. Seven other judges under the same template preserve the ordering (Appendix I.5). The eight settings outside the shared template do not use this rule. Three have no judge (InfoVQA, OCR$^{\text{EN}}$, and OCR$^{\text{ZH}}$), and the five benchmark-specific judge prompts (CharXiv, MM-Vet, HLE, MultiCh, and MMVU) do not tell the judge to read the reasoning. On these eight settings, ZPPO is above GRPO† in all 32 (scale, setting) cells, by $+2.8$/$+1.9$/$+2.0$/$+2.0$ pp on average at 0.8B/2B/4B/9B. On the three settings with no judge, it is above in all 12 cells. Every trained row also learns under the instruction of Fig. 8, which replaces each benchmark's own format instruction. Gains over the untrained base therefore include adaptation to that instruction, while the ZPPO $-$ GRPO† comparison holds it fixed.

(ii) Elements the controls leave out. Our Hint control takes elements of Guide-GRPO, NuRL, HINT, HiLL, TAPO, and SocraticPO (Liu et al., 2026c), and our Prefix control takes elements of Hint-GRPO, BREAD, StepHint, SEELE, Prefix-RFT, and LUFFY (Yan et al., 2025), under the shared settings (Appendix E.2). The Hint control leaves out Guide-GRPO's importance weight toward the plain prompt and its pooled advantage, HINT's Affinity weighting and its gradient on the original question, and SocraticPO's guidance conditioned on the failed attempt, its retry with that attempt in context, and its decayed reward. The Prefix control leaves out LUFFY's teacher traces in the rollout group under regularized importance sampling. RSTG† leaves out RSTG's fine-tuning term (Appendix I.3).

(iii) NCQ's mechanism. The NCQ answer list can rule out options on multiple-choice questions. Two observations bear on how much this contributes. (a) Over the audited NCQ rollouts, NCQ-prompt

accuracy stays at 1.2/22.1/35.2/30.5% at 0.8B/2B/4B/9B (Tab. 23). At 0.8B, 82.7% of NCQ rollouts repeat one of the listed wrong answers instead of avoiding them. (b) We compute NCQ's marginal contribution from the printed cells of Tab. 3, Tab. 15, and Tab. 18. At 2B/4B/9B, it is larger on the eight open-answer settings, which have no options to rule out, than on the thirteen multiple-choice settings: +1.35/+1.49/+1.80 against +0.73/+0.98/+0.46 pp. The order is reversed at 0.8B (+1.13 against +1.73 pp). The multiple-choice settings are CEval, GPQA-D, MMLU, MMLU-Pro, MMLU-Rd, AI2D, EmbSp, MMMU^{Pro}, VisP, WeMath, MVBench, VMME, and VMME^{S}. The open-answer settings are IMO-AB, MultiCh, InfoVQA, OCR^{EN}, OCR^{ZH}, MM-Vet, CharXiv, and VBlind. AIME25 and AIME26, where one problem moves a cell by 3.3 pp, and the eight mixed-format settings are left out. Both observations are indirect, and a control without the list would test this directly.

(iv) Candidate labels after compression. Each compressed candidate keeps its source rollout's reward and is not graded again (Appendix E.1). In the BCQ audit, a new rollout crosses the candidate labels in 15 of 450 graded-correct rollouts at 0.8B, in 3 of 112 graded-wrong rollouts at 9B, and in none at 2B and 4B (Appendix G.8).

(v) Video at 4B and 9B. There, ZPPO is +0.3/+0.4 pp above the untrained base and below it on two and one of the five settings (Appendix F.1). Against $\text{GRPO}^{\dagger}$, it is +0.9/+0.8 pp and above it on all five settings (Tab. 14).

(vi) The Correction-SFT input. Its input holds the question and the tag, as in NAT, while its target corrects an attempt that the input does not include (Appendix E.2). It keeps its training-time `<correct>` tag at evaluation, as NAT does.

(vii) Scope of the distillation result. Off-Distill and On-Distill use one loss, the generalized JSD at $\beta = 0.5$, on one corpus. We tried two learning rates and, for $\text{On-Distill}^{\dagger}$, two group sizes and two corpus passes (Appendix E.2, Appendix I.1). Their loss of accuracy beyond the training corpus is an observation in this setting, consistent with the prior reports cited in Sec. 1.

(viii) Training-side diagnostics. Graduation (Appendix I.10), the replay × reformulation interaction (Appendix G.4), the sweep over I (Appendix G.5), and the *Norm w/ Zero* drop (Appendix D) are read from training-side measurements, and each of those appendices states what its measurement covers.