

PaceReasoner: Label-Free Learning to Answer Briefly from Long Reasoning

Changguo Jia^{1*} Zhiyou Xiao^{1*} Xinbei Ma^{2*} Geng Chen¹
Weiming Zhang³ Yiming Xu¹ Tianqi Zhao⁴ Minghui Zhou^{1†}

¹Peking University ²National University of Singapore

³Shanghai Jiao Tong University ⁴Zhongguancun Laboratory

jiachangguo@stu.pku.edu.cn xiaozhiyou@stu.pku.edu.cn

xinbei.ma@nus.edu.sg zhmh@pku.edu.cn

Abstract

Extended chain-of-thought reasoning improves problem-solving performance but increases inference cost and latency, motivating more accurate answers with shorter reasoning. We introduce PaceReasoner, a label-free test-time reinforcement learning framework that uses a model’s LONG responses to improve its SHORT-mode accuracy. Both modes share a trainable policy, creating a need for reliable supervision, sufficient SHORT training samples, and continued LONG responses to guide learning. PaceReasoner combines all-response and LONG-only answer consensus through progressive, confidence-weighted supervision. A behavior reward gradually shifts from encouraging SHORT exploration to rewarding agreement with LONG answers, while a ratio controller regulates the balance between modes. Across three model backbones and four benchmarks, PaceReasoner improves average SHORT accuracy over the baselines by 7.4–13.5 percentage points, with fewer than 25 reasoning tokens per response on average. Ablations support the key components, and cross-dataset results demonstrate transfer beyond the adaptation set.

1 Introduction

Large reasoning models improve problem-solving performance by devoting additional test-time computation to extended chain-of-thought reasoning (Guo et al., 2025; Qwen Team, 2025). However, longer reasoning increases token usage and response latency, while unnecessarily extended reasoning may yield limited additional gains (Chen

et al., 2025; Ma et al., 2025). For simpler questions or applications with latency or cost constraints, these gains may not justify the additional computation. Existing approaches manage this trade-off by learning when to invoke explicit reasoning (Fang et al., 2025; Zhang et al., 2025; Kim et al., 2026), compressing reasoning traces through self-distillation (Sang et al., 2026), or controlling and allocating reasoning budgets (Li et al., 2026a; Lin et al., 2026). These approaches primarily optimize the reasoning cost rather than the problem-solving capability of a low-cost mode itself. Therefore, they motivate a complementary question: how can we improve the model’s problem-solving accuracy when it operates in shorter reasoning mode?

Our starting point is the accuracy gap between LONG and SHORT modes of the same model, where zero-shot LONG achieves higher average accuracy than SHORT across all three evaluated backbones. This gap suggests that the model’s own LONG responses may provide useful supervision for improving SHORT without reference answers. Motivated by this observation, we formulate the problem as **label-free test-time adaptation**: given a set of test questions without access to their reference answers, our goal is to **improve the problem-solving accuracy of SHORT mode while keeping its reasoning brief**.

We introduce PaceReasoner, a test-time reinforcement learning framework that jointly optimizes LONG and SHORT within a shared policy, using LONG answers to guide SHORT learning without reference answers. This approach faces three challenges. First, **supervision reliability**: LONG responses may disagree or converge on an incorrect answer. Second, **SHORT learning op-**

*Equal contribution.

†Corresponding author.

opportunities: the initially stronger LONG mode may dominate generation, limiting the samples available for improving SHORT. Third, **continued LONG supervision:** encouraging SHORT too strongly may leave too few LONG responses to support further learning.

PaceReasoner addresses these challenges through coordinated answer and behavior rewards. To improve **supervision reliability**, progressive supervision combines answer consensus across all responses with consensus from LONG responses alone, while confidence weighting reduces reliance on uncertain LONG consensus. To provide **SHORT learning opportunities**, a behavior reward initially encourages SHORT exploration and gradually shifts toward rewarding agreement with LONG answers. To preserve **continued LONG supervision**, a ratio controller adjusts this reward according to the proportion of SHORT responses in the batch, counteracting mode imbalance. Finally, Group Relative Policy Optimization (GRPO) updates the complete responses from both modes, including mode selection and subsequent generation.

Our evaluation on three model backbones and four benchmarks shows that PaceReasoner improves average SHORT-mode accuracy over TTRL by 7.4–13.5 percentage points, while using fewer than 25 reasoning tokens per response on average for each backbone. Our further ablation studies support the contributions of progressive supervision and behavior control, while cross-dataset experiments show that SHORT-mode improvements transfer beyond the adaptation dataset.

In summary, our contributions are threefold:

- We introduce PaceReasoner, which improves short reasoning using answer supervision from the same model’s LONG responses on unlabeled target questions.
- We coordinate progressive consensus supervision, an exploration-to-agreement reward, and ratio control to address noisy supervision, insufficient SHORT coverage, and loss of LONG evidence during joint training.
- Across three models and four benchmarks, PaceReasoner improves average SHORT accuracy over baselines at fewer than 25 reasoning tokens on average. Ablations support each key component, while cross-dataset results

demonstrate transfer beyond the adaptation set.

2 Related Work

2.1 Reasoning Compression and Distillation

Length-constrained reinforcement learning and pruning improve the accuracy–cost trade-off (Aggarwal and Welleck, 2025a; Hou et al., 2025a). Progressive CoT removal internalizes intermediate steps (Deng et al., 2024), while on-policy distillation learns from teacher distributions on student-generated trajectories (Agarwal et al., 2024). Teacher conditioning includes privileged solutions in OPSD (Zhao et al., 2026a), conciseness instructions in CRISP (Sang et al., 2026), and hidden reasoning prefixes in Masked Self-Distillation (Kalwar et al., 2026). Alternatively, LONG answers guide learning without distribution matching. PaceReasoner supervises SHORT using sequence-level rewards from LONG answer consensus.

Adaptive Reasoning. Qwen3 supports hybrid reasoning (Qwen Team, 2025). Thinkless decouples mode-selection and response optimization, while AdaptThink balances mode exploration through importance sampling (Fang et al., 2025; Zhang et al., 2025). RPO optimizes only the root token (Kim et al., 2026). Greater SHORT usage need not improve capability. Mode allocation governs supervision: LONG dominance restricts SHORT updates, whereas SHORT dominance reduces LONG evidence. PaceReasoner coordinates mixed-mode sampling to sustain SHORT learning and LONG supervision.

2.2 Label-Free Test-Time Adaptation

Self-consistency aggregates answers at inference (Wang et al., 2023), and TTRL converts majority agreement into reinforcement-learning rewards (Zuo et al., 2025); RENT and Intuitor use intrinsic confidence (Prabhudesai et al., 2025; Zhao et al., 2026b). U-OPSD distills disagreeing completions using consensus-supported pseudo-solutions (Li et al., 2026b), whereas TTPO distills agreeing rollouts and selectively penalizes disagreeing ones (Wang et al., 2026). Both improve non-thinking performance; TTPO studies thinking-to-non-thinking transfer. Joint adaptation couples supervision and mode allocation: updates change LONG evidence, while mode frequencies affect voting pools and training opportunities. All-response consensus offers broader coverage but

risks domination by weaker SHORT predictions; LONG-only consensus isolates deliberative evidence but may be sparse or incorrect. PaceReasoner combines progressive, confidence-weighted dual-consensus rewards with mode regulation to sustain supervision during joint adaptation.

3 Problem Definition and Preliminaries

3.1 Test-time Adaptation

Test-time adaptation aims to update a policy at inference time without additional ground-truth supervision. Given an initial policy π_{θ_0} and a set of test questions $\mathcal{D} = \{q_j\}_{j=1}^M$, the adaptation stage updates the policy to π_θ using rewards derived from its own inference. After the learning stage, we evaluate the updated policy on the same questions $\mathcal{D}$, where the generated final answers are scored against the reference answer. In this study, our goal is to improve performance in the short chain-of-thought setting, defined in the following section.

3.2 Reasoning Mode

Chain-of-thought largely facilitates the reasoning capabilities of an LLM. Given a question q , an LLM with parameters θ generates reasoning text z followed by a final answer a . We denote the resulting output by $o = (z, a)$:

$$o = (z, a) \sim \pi_\theta(\cdot \mid q). \quad (1)$$

To enable efficient reasoning, we use system prompts to elicit two reasoning modes, LONG and SHORT. The LLM is prompted either to choose between the two reasoning modes or to follow a specified mode, and to format its response accordingly.

(1) In LONG mode, the response contains explicit reasoning followed by the final answer:

<reason> z </reason><answer> a </answer>.

(2) In SHORT mode, the response gives the final answer directly, without an explicit reasoning trace:

<not></not><answer> a </answer>.

Thus, under the specified output format, $z = \emptyset$ in SHORT mode. The full prompt used during adaptation is provided in Appendix A.5. Therefore, the reasoning policy is denoted as

$$(z, a) \sim \pi_\theta(\cdot \mid q, m),$$

where L and S denote LONG and SHORT and both modes share the same policy parameters θ .

We define the reasoning cost of a response as the number of generated reasoning tokens before the final answer, excluding answer tokens. Let $\ell(z)$ denote this count. For mode m , the expected reasoning cost is

$$C_m = \mathbb{E}_{(z,a) \sim \pi_\theta(\cdot \mid q, m)}[\ell(z)]. \quad (2)$$

3.3 Group Relative Policy Optimization

Group Relative Policy Optimization (GRPO) is a reinforcement learning algorithm for LLM post-training that has been shown to be effective for reasoning tasks with outcome-based reward (Shao et al., 2024), where only the answer a is compared with a verifiable reference, without requiring supervision for the intermediate reasoning process.

Given a question q , GRPO samples a group of K responses $\{o_i\}_{i=1}^K$ from the policy. Each response receives a scalar reward r_i , and its advantage is computed relative to the other responses in the group, used to update the policy optimization,

$$\hat{A}_i = \frac{r_i - \text{mean}_j(r_j)}{\text{std}_j(r_j) + \epsilon}. \quad (3)$$

In our test-time adaptation setting, the reward r_i , is derived from the policy’s LONG mode of reasoning, transforming self-generated signals to improve token-efficient reasoning of SHORT mode.

4 PaceReasoner

4.1 Method Overview

Building on GRPO-based test-time adaptation, the primary idea of PaceReasoner is to construct a reward that encourages SHORT to generate the same answer as LONG, enhancing short reasoning of the policy model. We propose a novel reward that uses answer consensus from LONG. As is shown in Figure 1, for each question, the current policy performs K times of rollout, where it chooses a reasoning mode and follows the format to generate K responses. Since LONG always reasons to superior answer accuracy than SHORT, we use the predicted answer of LONG’s consensus to align SHORT. Another key is the mode-choosing behavior: because both modes are updated by the same policy, the training requires enough SHORT for learning while preserving enough LONG as a source of supervision. Therefore, we have to prevent the reasoning mode from collapsing into any single one.

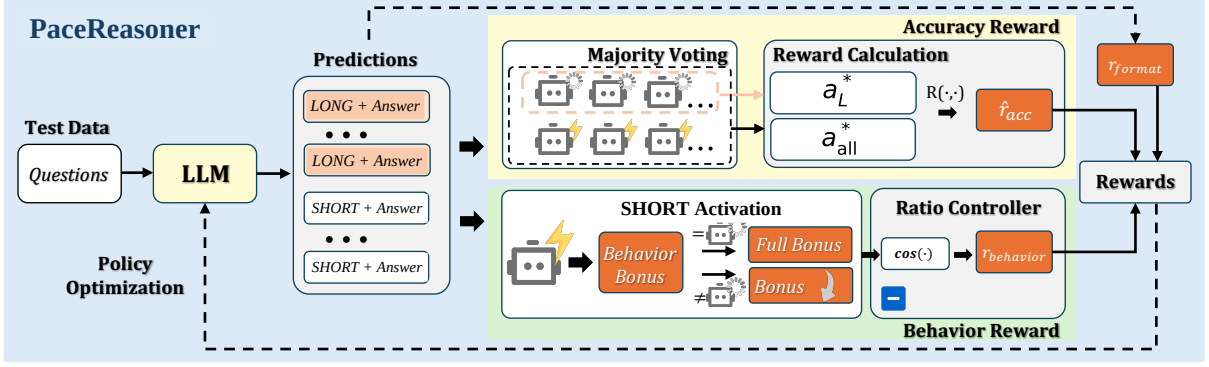


Figure 1: Overview of PaceReasoner. LONG and SHORT responses supply all-response and LONG-only answer consensus. The accuracy-reward block implements progressive supervision with the additional confidence weight in Eq. (7). SHORT activation and ratio control construct the behavior reward; format, answer, and behavior rewards jointly drive policy optimization.

The reward for $o_i = (m_i, z_i, a_i)$ is defined as

$$r_i = r_{fmt}(o_i) + r_{acc}(q, a_i, t) + r_{beh}(o_i, t). \quad (4)$$

The format reward is 1 for a valid LONG or SHORT output and 0 otherwise. The answer reward supplies pseudo-supervision, while the behavior reward regulates reasoning mode exploration and prevalence.

4.2 Answer Reward: Long-CoT Consensus

Answer consensus. Let $\mathcal{A}_L(q)$ contain those generated in LONG mode, whose majority voting yields a pseudo-label a_L^* . For response o_i , let a_i denote its extracted final answer, and let a_L^* denote the majority-voted answer from the corresponding LONG answer pool. We define the answer-agreement reward $R(a_i, a_L^*)$ to be 1 if the task-specific answer checker judges a_i equivalent to a_L^* , and 0 otherwise. If either answer is missing, we set $R(a_i, a_L^*) = 0$.

$\mathcal{A}_L(q)$ voting isolates evidence produced through explicit deliberation, yet remains vulnerable to limited coverage, disagreement, and shared errors. We therefore consider All-response voting to provide feedback when LONG samples are scarce, using all extractable answers $\mathcal{A}_{all}(q)$ and the agreement $R(a_i, a_{all}^*)$ with their majority-voting label a_{all}^* as a secondary supervision.

Progressive supervision. With training progress $s_t = t/T$, where T is the total number of steps, the underlying supervision curriculum is

$$r_{acc}^{prog} = (1 - s_t)R(a_i, a_{all}^*) + s_t R(a_i, a_L^*). \quad (5)$$

Early updates rely more on all-response consensus, while later updates increasingly use LONG evidence. This schedule expresses a curriculum rather than a guarantee that LONG becomes correct as training proceeds.

Confidence weighting. Without ground-truth labels erroneous majority-vote pseudo-labels can steer policy updates toward incorrect answers (Zuo et al., 2025). We therefore weight answer-agreement reward $R(a_i, a_L^*)$ to reduce reliance on pseudo-labels, allowing varying degrees of supervision based on the “confidence”. Let $N_L = |\mathcal{A}_L|$, let v_1, v_2 be the largest and second-largest answer counts, and set $v_2 = 0$ when only one distinct answer occurs. For a minimum sample threshold $m \geq 1$, define

$$c_L(q) = \begin{cases} (v_1 - v_2)/N_L, & N_L \geq m, \\ 0, & N_L < m. \end{cases} \quad (6)$$

The final weight is $w_{q,t} = s_t c_L(q)$, giving

$$r_{acc}(q, a_i, t) = (1 - w_{q,t})R(a_i, a_{all}^*) + w_{q,t}R(a_i, a_L^*). \quad (7)$$

Such a weighting makes insufficient LONG samples or a tied vote return the answer reward to all-response consensus, avoiding rejecting the whole update. When the two pseudo-labels agree, the weight does not change this reward. We use this “consensus strength” as a reliability “confidence”, not a calibrated correctness probability, and a confidently wrong consensus can still reinforce errors.

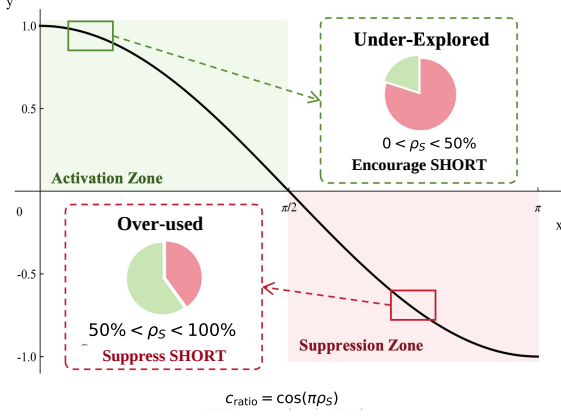


Figure 2: Cosine ratio control for SHORT activation. The horizontal coordinate is $x = \pi\rho_S$ and the vertical coordinate is c_{ratio} . Positive coefficients encourage SHORT exploration below $\rho_S = 0.5$, while negative coefficients suppress further SHORT dominance above it. Pie charts illustrate mode proportions rather than measured distributions.

4.3 Behavioral Reward: SHORT Activation and Ratio Control

From exploration to answer agreement. Answer-agreement rewards alone may favor the currently stronger LONG behavior, leaving SHORT with few training opportunities. We introduce a behavior bonus that encourages SHORT exploration early and increasingly emphasizes agreement with LONG consensus later.

Let $I_S(o_i)$ indicate a valid SHORT response. Its base behavior bonus is

$$b_i = I_S(o_i)\lambda_{\text{br}}[(1 - s_t) + s_t R(a_i, a_L^*)]. \quad (8)$$

A SHORT answer that disagrees with LONG consensus receives the decaying bonus $\lambda_{\text{br}}(1 - s_t)$, while an agreeing SHORT answer receives λ_{br} , and other responses with wrong formats receive zero. This shifts learning from making SHORT available to improving the SHORT answers. Confidence weighting modifies Eq. (7). The behavior bonus retains the time-based curriculum. This time-based schedule the curriculum: first encourages SHORT explorations and then leans to answer agreement.

Regulating the mode mixture. Unrestricted activation can make SHORT dominate sampling, reducing LONG evidence and changing the all-response consensus. We compute the proportion ρ_S of SHORT responses in the current training batch and rescale the bonus:

$$c_{\text{ratio}} = \cos(\pi\rho_S), \quad r_{\text{beh}} = b_i c_{\text{ratio}}. \quad (9)$$

When $\rho_S < 0.5$, the bonus encourages SHORT. Above 0.5, the negative coefficient opposes further SHORT dominance. Figure 2 illustrates this regulation: it does not constrain the learned mixture to remain at 50% but change the direction of the bonus.

Together, the behavioral reward adjusts the ratio of SHORT and LONG reasoning mode, encouraging attempts at SHORT while retaining LONG to provide learning signals, thereby avoiding collapse into either extreme. It provides batch-level feedback against mode imbalance, while answer quality still depends on consensus and policy optimization.

4.4 Joint Full-Trajectory Optimization

We use the combined reward in Eq. (4) to optimize complete trajectories from both modes with GRPO. Writing $u_{i,k}(\theta)$ for the token probability ratio between the current and rollout policies, the clipped policy objective is

$$J_{\text{clip}}(\theta) = \mathbb{E} \left[\frac{1}{K} \sum_{i=1}^K \frac{1}{|o_i|} \sum_{k=1}^{|o_i|} \min \left(u_{i,k} \hat{A}_i, \text{clip}(u_{i,k}, 1 - \delta, 1 + \delta) \hat{A}_i \right) \right]. \quad (10)$$

Here $|o_i|$ counts the full generated trajectory for optimization. Both mode selection and subsequent generation receive updates. Algorithm 1 summarizes the learning procedure: no target-task reference answers, external teacher, or additional verifier is used.

After adaptation, the policy generates a response conditioned on the question and the requested LONG or SHORT mode. SHORT generation uses the adapted policy directly, without requiring LONG responses or consensus computation.

5 Experiments

We evaluate whether PaceReasoner improves SHORT-mode accuracy while reducing reasoning cost, and analyze the contribution of each method component.

5.1 Experimental Setup

Benchmarks. We evaluate on GSM8K and MATH-500 for mathematical reasoning and verifiable answers (Cobbe et al., 2021; Hendrycks et al., 2021), and on CSQA and GPQA-Diamond for commonsense and scientific reasoning (Talmor et al.,

Mode	Method	CSQA		GPQA-D		MATH-500		GSM8K		Average	
		Acc.	R-tok.	Acc.	R-tok.	Acc.	R-tok.	Acc.	R-tok.	Acc.	R-tok.
Qwen3-4B-Base											
LONG	Zero-shot	50.9	413.4	21.3	461.5	50.0	510.0	60.4	328.8	45.7	428.4
	TTRL	80.9	159.1	31.8	386.3	79.4	<u>501.2</u>	<u>93.1</u>	203.0	<u>71.3</u>	312.4
	Length-Penalty	39.9	<u>158.6</u>	<u>33.3</u>	<u>283.2</u>	56.2	<u>510.6</u>	85.8	201.5	53.8	<u>288.5</u>
	CRISP	69.6	527.7	32.5	699.3	47.6	759.0	53.4	<u>186.6</u>	50.8	543.1
	PaceReasoner	<u>80.8</u>	92.0	36.9	168.0	<u>77.2</u>	450.2	93.7	173.6	72.2	221.0
SHORT	Zero-shot	25.8	482.1	11.6	306.0	<u>31.6</u>	401.0	<u>30.9</u>	286.0	25.0	368.8
	TTRL	78.1	<u>11.0</u>	22.9	19.8	19.4	<u>13.4</u>	22.9	19.8	35.8	16.0
	Length-Penalty	80.7	<u>12.4</u>	14.0	<u>18.3</u>	26.4	<u>13.9</u>	26.7	<u>15.6</u>	<u>37.0</u>	<u>15.1</u>
	CRISP	67.1	9.2	<u>31.1</u>	9.2	23.4	7.8	21.2	13.2	35.7	9.8
	PaceReasoner	<u>80.2</u>	12.0	36.9	51.4	39.8	14.5	39.1	19.3	49.0	24.3
Qwen3-8B-Base											
LONG	Zero-shot	77.0	149.8	29.8	514.0	61.8	466.0	72.1	228.3	60.2	339.5
	TTRL	84.0	<u>127.4</u>	31.3	<u>319.0</u>	79.8	487.7	<u>94.2</u>	210.5	<u>72.3</u>	<u>286.2</u>
	Length-Penalty	61.4	133.7	<u>32.8</u>	419.5	65.5	<u>463.6</u>	86.8	205.3	61.6	305.5
	CRISP	51.9	402.5	31.5	856.0	49.7	896.3	59.0	318.5	48.0	618.3
	PaceReasoner	<u>83.0</u>	112.9	36.9	241.9	<u>77.8</u>	462.9	95.0	207.9	73.2	256.4
SHORT	Zero-shot	39.1	100.8	24.2	100.7	<u>34.6</u>	206.0	<u>28.5</u>	201.3	31.6	152.2
	TTRL	<u>81.2</u>	11.0	<u>31.3</u>	11.0	27.2	<u>13.5</u>	8.1	13.2	37.0	12.2
	Length-Penalty	73.9	23.8	5.5	14.6	31.0	12.1	19.4	11.9	32.5	15.6
	CRISP	68.4	<u>17.1</u>	28.7	<u>11.2</u>	26.8	13.9	25.0	14.8	<u>37.2</u>	14.3
	PaceReasoner	82.5	11.0	36.4	11.0	37.2	14.0	46.0	<u>13.2</u>	50.5	<u>12.3</u>
Llama3.1-8B-Instruct											
LONG	Zero-shot	62.7	175.1	16.7	869.6	31.4	612.3	72.1	321.0	45.7	494.5
	TTRL	66.9	<u>133.6</u>	<u>27.9</u>	<u>308.9</u>	33.0	434.2	90.5	207.9	<u>54.6</u>	271.2
	Length-Penalty	60.4	166.8	23.2	354.5	35.9	417.1	84.4	<u>177.0</u>	51.0	278.9
	CRISP	<u>68.6</u>	127.3	26.7	357.3	25.9	291.4	76.7	156.4	49.5	233.1
	PaceReasoner	68.7	145.2	28.8	214.6	<u>33.8</u>	<u>400.2</u>	<u>89.6</u>	225.1	55.2	<u>246.3</u>
SHORT	Zero-shot	60.6	187.1	14.4	790.8	10.6	634.7	<u>11.8</u>	313.2	24.4	481.5
	TTRL	<u>66.3</u>	<u>11.0</u>	20.1	30.7	1.0	<u>13.0</u>	7.3	13.1	23.7	<u>17.0</u>
	Length-Penalty	63.5	16.3	23.5	37.6	6.4	69.7	8.5	<u>11.8</u>	25.5	33.9
	CRISP	68.6	4.0	29.5	4.9	<u>11.0</u>	6.3	10.6	5.4	<u>29.9</u>	5.2
	PaceReasoner	65.6	12.6	<u>28.8</u>	<u>30.1</u>	14.4	14.0	15.5	13.0	31.1	17.4

Table 1: Performance comparison under LONG and SHORT reasoning modes for Qwen3-4B/8B-Base and Llama3.1-8B-Instruct models. Acc.: mean@8 accuracy (%); R-tok.: reasoning tokens before the final answer, excluding answer tokens. Average is the unweighted mean across four tasks. Shading highlights PaceReasoner. Bold and underlining indicate the best and second-best results within each backbone and mode, respectively.

2019; Rein et al., 2024). Together, these benchmarks cover both open-ended and multiple-choice questions.

Implementation. We evaluate Qwen3-4B-Base, Qwen3-8B-Base, and Llama3.1-8B-Instruct (Qwen Team, 2025; Grattafiori et al., 2024). Adaptation is performed independently on each target benchmark without reference-answer rewards. We use GRPO in EasyR1 (Shao et al., 2024; Zheng et al., 2025), AdamW with learning rate 10^{-6} , 15 epochs, and 15 rollouts per question at temperature 1.0. Rollout batches contain 256 questions for CSQA/GSM8K and 64 for GPQA-Diamond/MATH-500. The configured actor update batch sizes are respectively 64 and 16 question groups, corresponding to 960 and 240 response trajectories after expansion by 15 rollouts. These are distinct from rollout batch sizes.

We use $\lambda_{br} = 2$ and 8 NVIDIA H20 GPUs.

Baselines. We compare zero-shot CoT, TTRL (Zuo et al., 2025), Length-Penalty, and CRISP (Sang et al., 2026) from the same initial checkpoint for each backbone, using the same target data for adaptation and a unified mean@8 evaluation protocol. Length-Penalty is our adaptation of reasoning-length control ideas from ThinkPrune (Hou et al., 2025b), DLER (Liu et al., 2025), and L1 (Aggarwal and Welleck, 2025b) to the label-free target-task setting. Its reward is $R_{fmt} + R_{acc}R_{length}$, where $R_{length} = 1/(L - L_{med})$ for response length L above the batch median L_{med} , and 1 otherwise.

Evaluation Metrics. For each question and mode, we sample eight responses and score each final answer as correct (1) or incorrect (0). Accu-

Supervision	CSQA		GPQA-D		MATH-500		GSM8K	
	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time
All-response	79.6	2h 24m	32.7	44m	30.2	2h 17m	26.1	4h 25m
LONG-only	78.9	2h 13m	28.2	44m	45.0	2h 22m	36.9	4h 38m
Progressive	80.2	1h 57m	36.9	40m	39.8	2h 9m	39.1	3h 51m

Table 2: Supervision ablation on Qwen3-4B-Base. Acc.: SHORT accuracy (%); time: RL training time.

SHORT exploration	LONG-answer matching	SHORT ratio control	Eval. mode	CSQA			GPQA-D			MATH-500			GSM8K		
				Acc.	R-tok.	Score	Acc.	R-tok.	Score	Acc.	R-tok.	Score	Acc.	R-tok.	Score
No	No	No	LONG	79.2	128.1	0.6	36.0	107.5	0.3	76.7	479.8	0.2	93.3	206.6	0.5
Yes	No	Yes	SHORT	79.3	<u>14.7</u>	5.4	<u>33.3</u>	31.4	1.1	40.0	14.5	2.8	<u>25.8</u>	19.3	1.3
Yes	Yes	No	SHORT	77.3	12.0	<u>6.4</u>	0.0	11.0	0.0	21.4	14.5	1.5	16.8	19.3	0.9
Yes	No	No	SHORT	<u>80.0</u>	12.0	<u>6.7</u>	22.7	<u>31.4</u>	0.7	24.4	14.5	1.7	25.2	19.3	<u>1.3</u>
Yes	Yes	Yes	SHORT	80.2	12.0	6.7	36.9	51.4	<u>0.7</u>	<u>39.8</u>	14.5	<u>2.7</u>	39.1	19.3	2.0

Table 3: Behavior-reward ablation on Qwen3-4B-Base. SHORT exploration rewards choosing SHORT; LONG-answer matching rewards SHORT answers that match LONG consensus (Eq. (8)). SHORT ratio control scales this bonus using the sampled SHORT proportion (Eq. (9)). The last row enables all three components. Score = Acc./R-tok.; accuracy is in percentage units. Among SHORT results, bold and underlining mark the best and second-best distinct reported values, respectively (higher Acc./Score, lower R-tok.). The LONG row is a reference.

Algorithm 1 PaceReasoner test-time adaptation

Require: Initial policy π_{θ_0} , questions $\mathcal{D}$, steps T

- 1: **for** $t = 1, \dots, T$ **do**
- 2: Sample a question batch from $\mathcal{D}$
- 3: Generate K mixed-mode responses per question
- 4: Parse modes and final answers
- 5: Compute all/LONG consensus and $c_L(q)$
- 6: Compute batch SHORT ratio ρ_S and $s_t = t/T$
- 7: Compute answer reward using Eq. (7)
- 8: Compute behavior reward using Eqs. (8)–(9)
- 9: Add format reward and normalize within each group
- 10: Update both modes on their complete trajectories
- 11: **end for**
- 12: **return** Adapted policy π_θ

racy is the mean of these scores across responses and questions, reported as mean@8 (%). We do not use voting or best-of-eight selection. R-tok. is the average number of generated reasoning tokens before the final answer, excluding answer tokens. We report unweighted averages across tasks. Table 1 reports accuracy and reasoning cost separately for LONG and SHORT. All main-table PaceReasoner results use the final method with confidence-weighted progressive supervision; the full-method

reference results in the component studies match the main table.

5.2 Short Reasoning Performance

Comparison with baselines. PaceReasoner achieves higher average SHORT accuracy than all baselines, exceeding TTRL by 13.2, 13.5, and 7.4 points on Qwen3-4B, Qwen3-8B, and Llama3.1-8B at 24.3, 12.3, and 17.4 reasoning tokens, respectively (Table 1). All three improve over zero-shot with fewer tokens. Qwen3-8B’s cost is similar to TTRL (12.3 versus 12.2 tokens) and below CRISP and Length-Penalty. Llama also uses fewer tokens than Length-Penalty. Qwen3-8B gains 37.9 GSM8K points over TTRL. Both average 13.2 reasoning tokens. These comparisons use average costs, not matched per-question budgets.

LONG also achieves higher average accuracy than all baselines, using fewer tokens than TTRL and Length-Penalty on all backbones and than CRISP on both Qwen models. SHORT improvements thus accompany a stronger LONG mode.

LONG–SHORT accuracy gap. The absolute LONG–SHORT accuracy gap is within 1 point in 41.7% (5/12) of model–task pairs and within 5 points in 50% (6/12). The latter cover CSQA and GPQA-D across all three models, with SHORT using 69.4%–95.5% fewer tokens. This supports SHORT for timely, sufficiently accurate answers

Reasoning mode	Model	CSQA	GPQA-D	MATH-500	GSM8K
LONG	Qwen3-4B (Thinking)	56.8	15.2	65.4	92.3
	4B-Base (PaceReasoner)	80.8	36.9	77.2	93.7
	Qwen3-8B (Thinking)	82.0	16.7	61.4	93.6
	8B-Base (PaceReasoner)	83.0	36.9	77.8	95.0
SHORT	Qwen3-4B (Non-thinking)	76.7	37.8	18.2	18.4
	4B-Base (PaceReasoner)	80.2	36.9	39.8	39.1
	Qwen3-8B (Non-thinking)	79.4	37.4	24.2	21.1
	8B-Base (PaceReasoner)	82.5	36.4	37.2	46.0

Table 4: Accuracy (%) of PaceReasoner adapted from Qwen3-4B/8B-Base and released Qwen3-4B/8B checkpoints. Thinking and non-thinking are paired with LONG and SHORT, respectively. Bold indicates the higher accuracy within each model-size and mode pair.

on these tasks, while larger mathematical gaps motivate retaining LONG.

5.3 Ablation Study

Table 2 and Table 3 present ablation results to analyze the contribution of method components.

Supervision strategy. Progressive supervision gives the highest SHORT accuracy on three tasks and the shortest training time on all four (Table 2), supporting its accuracy and training-efficiency benefits. LONG-only nevertheless leads on MATH-500 (45.0%), so supervision preferences remain task-dependent.

Behavior reward and ratio control. Table 3 ablates SHORT exploration, LONG-answer matching, and ratio control. Score measures accuracy per reasoning token (Kaiser et al., 2026). Without behavior rewards, the model relies on LONG. Removing ratio control reduces GSM8K accuracy from 39.1% to 16.8% at the same 19.3 tokens. Removing LONG-answer matching reduces GSM8K accuracy to 25.8% and GPQA-D accuracy from 36.9% to 33.3%. GPQA-D length falls from 51.4 to 31.4 tokens, increasing Score from 0.7 to 1.1 despite lower accuracy. MATH-500 differs by only 0.2 points. The components thus provide task-dependent accuracy gains.

5.4 Comparison with Post-trained Qwen3

To assess our label-free test-time adaptation with post-training method, Table 4 compares PaceReasoner with the released Qwen3-4B/8B checkpoints, using `enable_thinking=True/False` for their thinking/non-thinking modes (Qwen Team, 2025). SHORT exceeds non-thinking accuracy on six of eight model-task pairs, gaining 13.0–24.9

Eval. dataset	Training data	LONG	SHORT
GPQA-D	None (zero-shot)	21.3	11.6
	GPQA-D	36.9	36.9
	CSQA	37.4	34.7
MATH-500	None (zero-shot)	50.0	31.6
	MATH-500	77.2	39.8
	GSM8K	74.2	36.8

Table 5: Cross-dataset accuracy (%) on Qwen3-4B-Base. Each row specifies the evaluation and training datasets. None (zero-shot) uses the initial model; all trained rows use PaceReasoner with unlabeled questions from the listed training dataset.

points on mathematics. CSQA gains are smaller, at 3.1–3.5 points. GPQA-D is 0.9–1.0 points lower. LONG exceeds thinking accuracy on all eight pairs. This reference comparison involves models with different post-training histories.

5.5 Cross-Dataset Transfer

To investigate whether SHORT-mode improvements transfer beyond the adaptation dataset, we further adapt Qwen3-4B-Base on CSQA or GSM8K and evaluate on GPQA-D or MATH-500, respectively shown in Table 5. SHORT gains 23.1 and 5.2 points over zero-shot, trailing direct target-set adaptation by 2.2 and 3.0 points. This retains 63–91% of the target-adaptation gains. LONG differs from target-set adaptation by +0.5 and –3.0 points. Both modes improve over zero-shot without target-set adaptation, suggesting that the gains are not solely due to same-dataset overfitting.

6 Conclusion

We presented PaceReasoner, a label-free test-time reinforcement learning framework that improves SHORT-mode accuracy using answer supervision

from the same model’s LONG responses. Progressive, confidence-weighted supervision and behavior control coordinate answer learning and mode sampling within a shared policy. Across three models and four benchmarks, PaceReasoner improves average SHORT accuracy over TTRL by 7.4–13.5 percentage points with fewer than 25 reasoning tokens per response on average. Ablations support the key components, and cross-dataset results demonstrate transfer beyond the adaptation set. These findings support using a model’s own long-reasoning answers to strengthen its performance with short reasoning.

Limitations

Our evaluation covers three 4B/8B models and four question-answering benchmarks, without coding or interactive deployment tests. Target-set adaptation and two transfer experiments provide limited evidence of generalization. Multi-seed variability is not measured.

Self-generated consensus can reinforce shared errors and depends on the model retaining useful LONG reasoning capability. Confidence weighting is heuristic, and its incremental benefit is not isolated by the available ablations. Ratio control does not guarantee per-question LONG coverage; sensitivity to controller and reward settings remains a limitation. SHORT is a prompted mode rather than a fixed token budget.

Our cost metric counts reasoning tokens before the final answer, excluding answer tokens and adaptation computation. These measurements do not establish lower end-to-end latency, total deployment cost, or energy use.

References

- Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. 2024. [On-Policy Distillation of Language Models: Learning from Self-Generated Mistakes](#). In *The Twelfth International Conference on Learning Representations*.
- Pranjal Aggarwal and Sean Welleck. 2025a. [L1: Controlling How Long A Reasoning Model Thinks With Reinforcement Learning](#). In *Second Conference on Language Modeling*.
- Pranjal Aggarwal and Sean Welleck. 2025b. [L1: Controlling how long a reasoning model thinks with reinforcement learning](#). In *Second Conference on Language Modeling*.
- Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025. [Do NOT think that much for 2+3=? on the overthinking of long reasoning models](#). In *Forty-second International Conference on Machine Learning*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- Yuntian Deng, Yejin Choi, and Stuart Shieber. 2024. [From Explicit CoT to Implicit CoT: Learning to Internalize CoT Step by Step](#). *Preprint*, arXiv:2405.14838.
- Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2025. [Thinkless: LLM learns when to think](#). In *Advances in Neural Information Processing Systems*, volume 38.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. [Measuring mathematical problem solving with the MATH dataset](#). In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.
- Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. 2025a. [ThinkPrune: Pruning Long Chain-of-Thought of LLMs via Reinforcement Learning](#). *Preprint*, arXiv:2504.01296.
- Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. 2025b. [ThinkPrune: Pruning long chain-of-thought of LLMs via reinforcement learning](#). *arXiv preprint arXiv:2504.01296*.
- Daniel Kaiser, Arnoldo Frigessi, Ali Ramezani-Kebrya, and Benjamin Ricaud. 2026. [Decomposing reasoning efficiency in large language models](#). *arXiv preprint arXiv:2602.09805*.
- Durgesh Kalwar, Vardhan Palod, Jaya Adithya Pavuluri, and Subbarao Kambhampati. 2026. [Masked Self-Distillation: Internalizing the Chain-of-Thought in Language Models](#). *Preprint*, arXiv:2607.22629.

- Taehyeon Kim, Hyunsoo Lee, Youngsoo Jang, and Moontae Lee. 2026. [Efficiently learning to reason or not to reason: Root-token policy optimization for adaptive thinking](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 17934–17949. Association for Computational Linguistics.
- Junyan Li, Wenshuo Zhao, Yang Zhang, and Chuang Gan. 2026a. [Steering LLM thinking with budget guidance](#). In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 37437–37451, San Diego, California, United States. Association for Computational Linguistics.
- Yijiang Li, Bingyang Wang, Yijun Liang, Yunjie Tian, Di Fu, and Nuno Vasconcelos. 2026b. [On-policy self-distillation without any supervision](#). *arXiv preprint arXiv:2608.06296*.
- Junhong Lin, Xinyue Zeng, Jie Zhu, Song Wang, Julian Shun, Jun Wu, and Dawei Zhou. 2026. [Plan and budget: Effective and efficient test-time scaling on reasoning large language models](#). In *The Fourteenth International Conference on Learning Representations*.
- 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, et al. 2025. [DLER: Doing length penalty right—incentivizing more intelligence per token via reinforcement learning](#). *arXiv preprint arXiv:2510.15110*.
- Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. 2025. [Cot-valve: Length-compressible chain-of-thought tuning](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6025–6035.
- Mihir Prabhudesai, Lili Chen, Alex Ippoliti, Katerina Fragkiadaki, Hao Liu, and Deepak Pathak. 2025. [Maximizing Confidence Alone Improves Reasoning](#). *Preprint*, arXiv:2505.22660.
- Qwen Team. 2025. [Qwen3 technical report](#). *Preprint*, arXiv:2505.09388.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. [GPQA: A graduate-level google-proof Q&A benchmark](#). In *First Conference on Language Modeling*.
- Hejian Sang, Yuanda Xu, Zhengze Zhou, Ran He, Zhipeng Wang, and Jiachen Sun. 2026. [CRISP: Compressed reasoning via iterative self-policy distillation](#). *arXiv preprint arXiv:2603.05433*.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. 2024. [DeepSeekMath: Pushing the limits of mathematical reasoning in open language models](#). *arXiv preprint arXiv:2402.03300*.
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. [CommonsenseQA: A question answering challenge targeting commonsense knowledge](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.
- Aozhe Wang, Zhengxi Lu, Jianze Wang, Shangke Lv, Ying Liu, Weiming Lu, Jun Xiao, Yueting Zhuang, Hua Yang, Qianglong Chen, and Yongliang Shen. 2026. [TTPO: Test-time policy optimization](#). *arXiv preprint arXiv:2608.27448*.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. [Self-Consistency Improves Chain of Thought Reasoning in Language Models](#). In *The Eleventh International Conference on Learning Representations*.
- Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. 2025. [AdaptThink: Reasoning models can learn when to think](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3716–3730, Suzhou, China. Association for Computational Linguistics.
- Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. 2026a. [Self-Distilled Reasoner: On-Policy Self-Distillation for Large Language Models](#). *Preprint*, arXiv:2601.18734.
- Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. 2026b. [Learning to Reason without External Rewards](#). In *The Fourteenth International Conference on Learning Representations*.
- Yaowei Zheng, Juntong Lu, Shenzhi Wang, Zhangchi Feng, Dongdong Kuang, Yuwen Xiong, and Richong Zhang. 2025. [EasyR1: An efficient, scalable, multi-modality RL training framework](#). <https://github.com/hiyouga/EasyR1>.
- Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, et al. 2025. [TTRL: Test-time reinforcement learning](#). In *Advances in Neural Information Processing Systems*, volume 38.

A Additional Experimental Details

A.1 Benchmarks and Data

CSQA evaluates commonsense question answering with five answer options. GPQA-Diamond contains 198 graduate-level science questions. MATH-500 contains 500 problems from MATH, while GSM8K evaluates arithmetic word problems (Talmor et al., 2019; Rein et al., 2024; Hendrycks et al., 2021; Cobbe et al., 2021). The supplied question pools contain 1,221 CSQA, 198 GPQA-Diamond, 500 MATH-500, and 1,319 GSM8K items. For each task, adaptation and evaluation files contain the same question sequence. We use these questions as unlabeled target data for independent adaptation and evaluate their final answers. MATH-500 does not require a particular visible reasoning length for an answer to be scored correct. Multiple-choice and mathematical answers are parsed and normalized for voting and evaluation.

Evaluation generation settings. Zero-shot, TTRL, and PaceReasoner use forced LONG and SHORT instructions. For CRISP, the LONG group reports its native short-reasoning setting; the SHORT group reports native no-think for Qwen3-8B-Base and answer-only generation for the other backbones. These results use the unified LONG/SHORT labels in Table 1; actual reasoning lengths are measured for each method.

A.2 Training Configuration

Table 6 distinguishes a rollout batch of questions from an actor update batch. With 15 responses per question, the training framework expands the configured actor batch by a factor of 15 before distributing it across workers. Thus, a configured actor batch of 64 denotes 960 response trajectories globally, not 64 trajectories per GPU. Main-table evaluation uses the unified mean@8 protocol described in Section 5.1.

A.3 Consensus Construction

For each question, final-answer extraction forms an all-response answer pool and a LONG-only pool. We vote over normalized final answers, rather than entire reasoning traces. If a pool is empty, its pseudo-label is missing and its agreement reward is zero. A tie in the two leading LONG answer counts yields zero confidence in Eq. (6); this affects the answer-supervision weight but does not remove the time-based behavior bonus. The additional confidence weight therefore reduces reliance on weak

Algorithm 2 Majority Voting for Pseudo Ground Truth (Reasoning / All Rollouts)

Input: rollout set $\mathcal{O}(q) = \{o_i\}_{i=1}^N$ for each query q
Output: pseudo ground truths $gt_{\text{think}}(q)$ and $gt_{\text{all}}(q)$
Initialize dictionaries $\mathcal{A}_{\text{think}} \leftarrow \{\}$, $\mathcal{A}_{\text{all}} \leftarrow \{\}$
for each response $o_i \in \mathcal{O}(q)$ **do**
 $o_i \leftarrow \text{STRIPSPACESAROUNDTAGS}(o_i)$
 $m_i \leftarrow \text{DETECTMODE}(o_i)$
 $a_i \leftarrow \text{EXTRACTANSWER}(o_i)$
 if $a_i \neq \emptyset$ **then**
 Append a_i to $\mathcal{A}_{\text{all}}[q]$
 if $m_i = \text{think}$ **then**
 Append a_i to $\mathcal{A}_{\text{think}}[q]$
 end if
 end if
end for
for each query q **do**
 if $|\mathcal{A}_{\text{think}}[q]| > 0$ **then**
 $gt_{\text{think}}(q) \leftarrow \arg \max_{a \in \mathcal{A}_{\text{think}}[q]} \text{COUNT}(a, \mathcal{A}_{\text{think}}[q])$
 else
 $gt_{\text{think}}(q) \leftarrow \emptyset$
 end if
 if $|\mathcal{A}_{\text{all}}[q]| > 0$ **then**
 $gt_{\text{all}}(q) \leftarrow \arg \max_{a \in \mathcal{A}_{\text{all}}[q]} \text{COUNT}(a, \mathcal{A}_{\text{all}}[q])$
 else
 $gt_{\text{all}}(q) \leftarrow \emptyset$
 end if
end for

LONG consensus without establishing correctness of the fallback consensus.

Algorithm 2 retains the original construction of the LONG-only (think) and all-response answer pools. Confidence weighting is applied after this voting step.

A.4 Mode Instructions

The following template states the LONG/SHORT generation contract used in the revised method description. At forced-mode evaluation, the corresponding opening marker conditions generation. A direct answer is allowed as the empty-reasoning special case of SHORT; there is no separate SHORT token cap.

Parameter	CSQA / GSM8K	GPQA-D / MATH-500
Rollout batch (questions)	256	64
Actor update batch (question groups, configured)	64	16
Responses per question	15	15
Rollout batch (response trajectories)	3840	960
Actor update batch (response trajectories)	960	240
Training epochs	15	15
Learning rate	10^{-6}	10^{-6}
Optimizer	AdamW	AdamW
Weight decay	0.01	0.01
Learning-rate warmup ratio	0	0
Rollout temperature	1.0	1.0
Maximum prompt / response length	2048 / 2048	2048 / 2048
Reference KL loss coefficient	0.01	0.01
Behavior bonus λ_{br}	2	2
GPUs / tensor parallel size	8 / 2	8 / 2
Actor update microbatch (responses per device)	1	1
Gradient checkpointing	Enabled	Enabled
Rollout GPU memory utilization	0.6	0.6

Table 6: Training settings. Batch quantities are global unless explicitly marked per device. Maximum response length is an overall generation limit, not a separate SHORT budget.

LONG/SHORT mode specification

Choose a solution mode for the question.
LONG: develop the necessary explicit analysis, intermediate derivations, and checks. Use: `<reason>reasoning</reason><answer>final answer</answer>`
SHORT: solve concisely and directly, retaining only essential steps when needed. Use: `<not>brief reasoning</not><answer>final answer</answer>`
A direct answer may use an empty `<not></not>` block.
Place only the final answer in `<answer>...</answer>`.

```
'<not></not><answer> </answer>'.
ONLY include the final answer
between '<answer> </answer>' tags
and do not include any other texts.
- If you decide to think, then you MUST
enclose thinking process within
<reason> </reason> tags, and the
final answer MUST BE enclosed within
<answer> </answer> tags. Output as
'<reason> </reason><answer>
</answer>'. Do not contain any
<answer> </answer> tags within
<reason> </reason> tokens, and all
the texts MUST BE included in tags
'<reason> </reason>' and '<answer>
</answer>'.
# Requirements
ALL the output texts MUST BE enclosed
within tags.
```

A.5 Recorded Training Prompt

The recorded training prompt below uses the `<not></not>` marker for the direct-answer case of SHORT.

Recorded training prompt

You are required to first decide whether to think or not based on the given question. The reasoning process MUST BE enclosed within `<reason>` `</reason>` tags, and the final answer MUST BE enclosed within `<answer>` `</answer>` tags.

Output Format

- If you decide NOT to think, you should ONLY output `<not></not>` and DIRECTLY output the final answer, and DO NOT contain any thinking process, the final answer MUST BE enclosed within `<answer>` `</answer>` tags. Output as

B Additional Component Results

The tables in this section report Qwen3-4B-Base component studies, with full-method reference results matching Table 1. Gray entries identify runs evaluated using LONG after failing to activate the intended short behavior, and should not be read as SHORT measurements.

B.1 Controller Shape and Bonus Strength

Table 7 compares the controller functions, and Table 8 retains the reported shifted-cosine settings. These choices differ in their sign, scale, and behavior near the endpoints, so the comparison does not isolate smoothness or establish a unique optimal controller. The unshifted cosine gives the highest SHORT accuracy among the controller settings on

all four tasks, although its higher GPQA accuracy comes with longer reasoning and a lower accuracy-per-token score. In the bonus-strength comparison (Table 9), a value of 1 fails to activate SHORT on the two mathematical tasks; a value of 2 gives the highest SHORT accuracy on three tasks, while 3 is slightly better on GSM8K (39.3% versus 39.1%) with more reasoning tokens. On GPQA, the default bonus of 2 improves accuracy at the cost of longer reasoning, rather than a higher accuracy-per-token score.

B.2 Training Dynamics

Figure 3(a–b) shows that SHORT sampling approaches zero without activation and approaches one without ratio control. With both components, the rollout pool retains a mixture of modes, supplying SHORT training examples alongside LONG answers for supervision. These trajectories complement the behavior-reward ablation in Table 3.

On GPQA, all-response voting suffers an early accuracy drop and LONG-only voting declines later, whereas progressive supervision improves more steadily (Figure 3c). These trajectories complement the supervision comparison in Table 2; they do not separately isolate the additional effect of confidence weighting.

C Recorded Qualitative Examples

We retain the original recorded outputs for CSQA, GPQA, MATH-500, and GSM8K below. Their generated text is unchanged; these selected examples illustrate behavior and are not additional measurements from the updated main-table configurations. Mode markers inside a recorded response are preserved as part of that response. GT denotes the ground-truth reference answer. Table 10 includes the original questions and SHORT-mode responses, including zero-shot outputs. The remaining tables retain the LONG-mode examples.

Ratio Controller	CSQA			GPQA			MATH-500			GSM8K		
	acc	R-tok.	score	acc	R-tok.	score	acc	R-tok.	score	acc	R-tok.	score
$f(x)$												
1	77.3	12.0	6.4	0.0	11.0	0.0	21.4	14.5	1.5	16.8	19.3	0.9
$1 - x$	78.5	11.6	6.8	20.7	31.5	0.7	32.5	14.4	2.3	36.0	19.3	1.9
$\frac{1}{2} - x$	79.8	12.0	6.7	31.2	31.9	1.0	27.8	14.5	1.9	26.0	19.1	1.4
$\frac{1}{x}$	76.7	12.0	6.4	22.2	31.5	0.7	31.2	14.4	2.2	37.2	19.3	1.9
$\cos(\pi x)$	80.2	12.0	6.7	36.9	51.4	0.7	39.8	14.5	2.7	39.1	19.3	2.0

Table 7: Controller-function sensitivity on Qwen3-4B-Base. The reported score is accuracy divided by reasoning tokens; accuracy and length remain the primary measurements.

Ratio Controller	CSQA			GPQA			MATH-500			GSM8K		
	acc	R-tok.	score	acc	R-tok.	score	acc	R-tok.	score	acc	R-tok.	score
$f(x)$												
$\cos(\pi(x - \frac{\pi}{4}))$	74.1	11.5	6.4	28.8	31.6	0.9	25.2	13.8	1.8	90.3	198.3	0.5
$\cos(\pi(x + \frac{\pi}{4}))$	77.2	12.0	6.4	27.8	31.4	0.9	32.3	13.5	2.4	89.3	202.4	0.4
$\cos(\pi x)$	80.2	12.0	6.7	36.9	51.4	0.7	39.8	14.5	2.7	39.1	19.3	2.0

Table 8: Shifted-controller sensitivity. Functions and values are retained as reported. Gray entries are LONG evaluations, not successful SHORT results.

Behavior Bonus	CSQA			GPQA			MATH-500			GSM8K		
	acc	R-tok.	score	acc	R-tok.	score	acc	R-tok.	score	acc	R-tok.	score
λ_{br}												
1	79.9	13.0	6.1	32.3	31.3	1.0	78.0	440.7	0.2	92.2	194.8	0.5
2	80.2	12.0	6.7	36.9	51.4	0.7	39.8	14.5	2.7	39.1	19.3	2.0
3	80.0	13.0	6.2	30.3	35.0	0.9	39.6	14.8	2.7	39.3	21.9	1.8

Table 9: Behavior-bonus sensitivity. Gray entries are LONG evaluations. The score is accuracy divided by reasoning tokens.

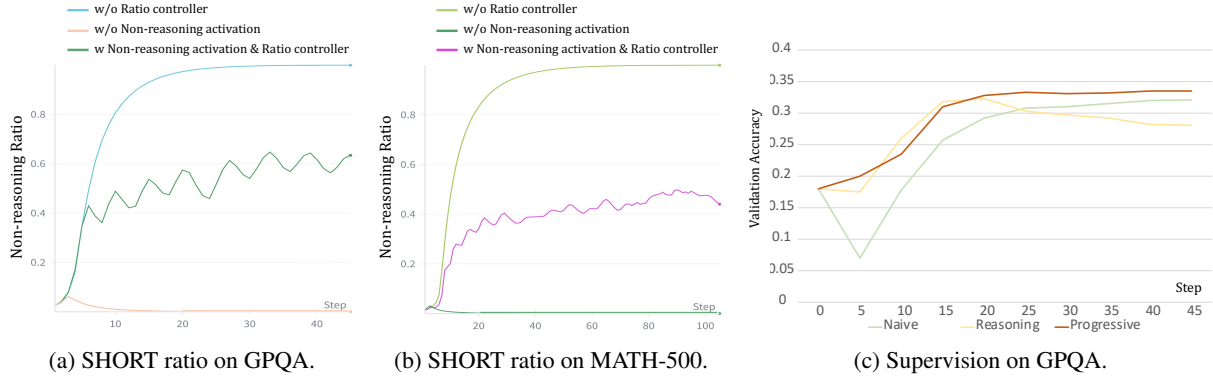


Figure 3: Training dynamics on Qwen3-4B-Base. (a–b) SHORT sampling with and without activation or ratio control; “Non-reasoning” denotes the direct-answer SHORT setting. (c) Validation accuracy under all-response (Naive), LONG-only (Reasoning), and progressive supervision.

Table 10: Recorded SHORT-mode responses of Qwen3-4B-Base. The questions, reference answers, and generated outputs are preserved from the original examples.

	CSQA	GPQA	MATH-500	GSM8K
Question	Question: The man closed his eyes as the music played, what effect did the music have? (A)coma (B)enjoyable (C)soothing (D)universal (E)good or bad	Question: Astronomers are observing a planet with a radius of 1 Earth radius in a circular orbit, ... What is the maximum orbital period of a second planet, with a radius of 2.5 times that of Earth, ... A.37.5 B.33.5 C.12.5 D.7.5	Question: Compute $\frac{1}{\cos^2 10^\circ} + \frac{1}{\sin^2 20^\circ} + \frac{1}{\sin^2 40^\circ}$	Question: Two trains leave San Rafael at the same time. They begin traveling westward, both traveling for 80 miles. The next day, they travel northwards, covering 150 miles. What’s the distance covered by each train in the two days?
GT	C	B	12	230
Zero-shot’s outputs	The man closed his eyes as the music played, what effect did the music have? <not></not><answer> (C)soothing</answer>	To solve this problem, we need to determine the maximum orbital period of the second planet that can both transit and occult the first planet. Here’s the step-by-step reasoning: ... <answer> D</answer>	To compute the expression $\frac{1}{\cos^2 10^\circ} + \frac{1}{\sin^2 20^\circ} + \frac{1}{\sin^2 40^\circ}$, we can use trigonometric identities and properties. First, recall the identity: $\frac{1}{\cos^2 \theta} = \sec^2 \theta$, $\frac{1}{\sin^2 \theta} = \csc^2 \theta$ (no answer)	The total distance covered by each train in the two days is calculated by adding the distance traveled on the first day (80 miles) and the distance traveled on the second day (150 miles). This gives us a total distance of 230 miles.
TTRL’s outputs	<not></not><answer> C</answer>	<not></not><answer> C</answer>	<not></not><answer> 16</answer>	<not></not><answer> 460 miles</answer>
Ours	<not></not><answer> C</answer>	<not></not><answer> B.33.5</answer>	<not></not><answer> 12</answer>	<not></not><answer> 230</answer>

CSQA	
Question	Question: A revolving door is convenient for two direction travel, but it also serves as a security measure at a what? (A)bank (B)library (C)department store (D)mall (E)new york
GT	A
Zero-shot's outputs	To determine whether to think or not, I will analyze the question and the options provided. Question: A revolving door is convenient for two-direction travel, but it also serves as a security measure at a what? Options: (A) bank (B) library (C) department store (D) mall (E) New York Reasoning: A revolving door is commonly used in places where a high volume of people move in and out, and it helps to prevent the spread of germs and maintain a sense of security. This is particularly important in financial institutions like banks, where security is a top priority. Given this information, I can conclude that the correct answer is: <answer>(A) bank</answer>
TTRL's outputs	<reason>The question asks which location a revolving door serves as a security measure. A revolving door is commonly used as a security measure at banks to control access and prevent unauthorized entry. Libraries, department stores, and malls may use revolving doors for convenience, but banks are the most likely to use them for security purposes.</reason><answer>A</answer>
PaceReasoner's outputs	<reason>A revolving door is typically used as a security measure at banks, as it helps prevent tailgating (where one person follows another into a secure area) and ensures that only one person can pass through at a time. While revolving doors can also be found in other places like department stores and malls, the primary security function is most commonly associated with banks.</reason><answer>A</answer>

Table 11: **Illustration of generated responses under LONG mode** on CSQA, including zero-shot, TTRL and PaceReasoner. The base model is Qwen3-4B-Base.

GPQA	
Question	Question: You are analyzing a small peptidic compound that has been chemically synthesized. The ¹ H NMR spectrum of the crude compound appears consistent with the expected molecule, with the exception of two peaks that both correspond to the same alpha-proton. These two peaks have similar chemical shifts and roughly equal integrals (together they integrate to 1H - the expected integral); based on the coupling pattern, you can rule out spin-spin coupling as an explanation for the duplicate peaks. LC-MS analysis of the crude compound at elevated temperature shows two clearly defined peaks of equal intensities. Both peaks have the same mass spectrum, which is consistent with the expected molecule. What is the most likely explanation for these observations? A. The compound is contaminated with a precursor B. The crude compound exists as a mixture of diastereoisomers C. 'Double coupling' has occurred during an amide-bond forming reaction D. The crude compound exists as a mixture of enantiomers. You are required to first decide whether to think or not based on the given question.
GT	B
Zero-shot's outputs	You are a helpful assistant. <reason>The question describes a small peptidic compound with a ¹ H NMR spectrum showing two peaks for the same alpha-proton, which are consistent in chemical shift and integral. LC-MS analysis at elevated temperature shows two peaks of equal intensities with the same mass spectrum. The absence of spin-spin coupling and the consistency with the expected molecule suggest that the issue is not due to contamination or diastereoisomers. The most plausible explanation is that the compound exists as a mixture of enantiomers, as enantiomers have identical NMR spectra and mass spectra but different optical rotations. The observation of two peaks in the NMR spectrum could be due to the presence of both enantiomers in the sample.</reason><answer>D</answer>
TTRL's outputs	<reason>The question describes a scenario where a synthetic peptidic compound shows two peaks in its ¹ H NMR spectrum that correspond to the same alpha-proton, with similar chemical shifts and integrals. The LC-MS analysis at elevated temperature shows two peaks of equal intensity with the same mass spectrum. The key points are: 1. NMR Analysis : Two peaks for the same proton, ruling out spin-spin coupling. 2. LC-MS Analysis : Two peaks of equal intensity with the same mass spectrum, indicating identical molecular species. The most likely explanation for these observations is that the compound exists in a form where these two peaks arise from different molecular conformations or isomers. Since the peaks have similar chemical shifts and integrals, they are likely not due to contamination or precursor molecules. Double coupling is not a plausible explanation given the context. Enantiomers would typically show distinct peaks in NMR due to their chirality, but here the peaks are identical. Diastereoisomers, however, can have similar NMR spectra due to their structural similarities, especially if they are in a conformational equilibrium.</reason><answer>B. The crude compound exists as a mixture of diastereoisomers </answer>
PaceReasoner's outputs	<reason>The question describes a situation where a synthetic peptidic compound shows two peaks in its ¹ H NMR spectrum that appear to be the same alpha-proton but do not couple to each other, and LC-MS analysis shows two peaks of equal intensity with identical mass spectra. The options provided are: A. Contamination with a precursor - This would likely show a different peak or pattern in the NMR or MS. B. Mixture of diastereoisomers - Diastereoisomers would have different physical properties and would show distinct peaks in NMR and MS. C. Double coupling during an amide-bond forming reaction - This would typically result in more complex splitting patterns due to the coupling constants involved. D. Mixture of enantiomers - Enantiomers would have identical physical properties and would show the same peaks in NMR and MS.</reason><answer>B. The crude compound exists as a mixture of diastereoisomers </answer>

Table 12: **Illustration of generated responses under LONG mode** on GPQA, including zero-shot, TTRL and PaceReasoner. The base model is Qwen3-4B-Base.

MATH-500	
Question	Question: What is the least positive integer multiple of 30 that can be written with only the digits 0 and 2?
GT	2220
Zero-shot's outputs	To find the least positive integer multiple of 30 that can be written with only the digits 0 and 2, we need to consider the properties of multiples of 30. A number is a multiple of 30 if and only if it is divisible by both 2 and 15. This means the number must end in 0 (to be divisible by 2) and the sum of its digits must be divisible by 3 (to be divisible by 15). Given that the number can only contain the digits 0 and 2, let's start by checking the smallest possible numbers: 1. Single-digit numbers : None of the single-digit numbers (0, 2) are multiples of 30. 2. Two-digit numbers : The only two-digit number that ends in 0 is 20, but 20 is not divisible by 15 (since $2 + 0 = 2$ is not divisible by 3). 3. Three-digit numbers : We need to find a number ending in 0 where the sum of the digits is divisible by 3. Let's check: - 200: ($2 + 0 + 0 = 2$) (not divisible by 3) - 220: ($2 + 2 + 0 = 4$) (not divisible by 3) - 202: ($2 + 0 + 2 = 4$) (not divisible by 3) - 222: ($2 + 2 + 2 = 6$) (divisible by 3) Since 222 is divisible by 3 and ends in 0, it is a multiple of 30. Therefore, the least positive integer multiple of 30 that can be written with only the digits 0 and 2 is 222. <reason>The number must end in 0 to be divisible by 2 and the sum of its digits must be divisible by 3 to be divisible by 15. Checking the smallest possible numbers, we find that 222 meets both criteria.</reason><answer>222</answer>
TTRL's outputs	<reason>To find the least positive integer multiple of 30 that can be written with only the digits 0 and 2, we need to consider the properties of multiples of 30. A number is a multiple of 30 if and only if it is a multiple of both 3 and 10. 1. Multiple of 10 : A number is a multiple of 10 if its last digit is 0. Therefore, the number we are looking for must end in 0. 2. Multiple of 3 : A number is a multiple of 3 if the sum of its digits is a multiple of 3. Since the only digits we can use are 0 and 2, we need to find the smallest number ending in 0 where the sum of the digits is a multiple of 3. Let's start constructing such a number: - The smallest number ending in 0 is 20 ($2 + 0 = 2$, not a multiple of 3). - The next candidate is 220 ($2 + 2 + 0 = 4$, not a multiple of 3). - The next candidate is 200 ($2 + 0 + 0 = 2$, not a multiple of 3). - The next candidate is 2200 ($2 + 2 + 0 + 0 = 4$, not a multiple of 3). - The next candidate is 2000 ($2 + 0 + 0 + 0 = 2$, not a multiple of 3). - The next candidate is 2220 ($2 + 2 + 2 + 0 = 6$, which is a multiple of 3). Thus, 2220 is the smallest number that meets both criteria: it ends in 0 and the sum of its digits is a multiple of 3. Therefore, the least positive integer multiple of 30 that can be written with only the digits 0 and 2 is 2220.</reason><answer>2220</answer>
PaceReasoner's outputs	<reason>To find the least positive integer multiple of 30 that can be written with only the digits 0 and 2, we need to consider the properties of multiples of 30. A number is a multiple of 30 if and only if it is a multiple of both 3 and 10. 1. Multiple of 10 : A number is a multiple of 10 if its last digit is 0. Therefore, the number we are looking for must end in 0. 2. Multiple of 3 : A number is a multiple of 3 if the sum of its digits is a multiple of 3. Since the only digits we can use are 0 and 2, the sum of the digits must be a multiple of 3. Given that the number must end in 0, the possible forms are 2220, 220, 20, etc. We need to check the smallest ones first: - 20 : Sum of digits = $2 + 0 = 2$ (not a multiple of 3) - 220 : Sum of digits = $2 + 2 + 0 = 4$ (not a multiple of 3) - 2220 : Sum of digits = $2 + 2 + 2 + 0 = 6$ (a multiple of 3) Since 2220 is the smallest number that meets both criteria (ends in 0 and the sum of its digits is a multiple of 3), it is the least positive integer multiple of 30 that can be written with only the digits 0 and 2.</reason><answer>2220</answer>

Table 13: **Illustration of generated responses under LONG mode** on MATH-500, including zero-shot, TTRL and PaceReasoner. The base model is Qwen3-4B-Base.

GSM8K	
Question	Question: I have 10 liters of orange drink that are two-thirds water and I wish to add it to 15 liters of pineapple drink that is three-fifths water. But as I pour it, I spill one liter of the orange drink. How much water is in the remaining 24 liters?
GT	15
Zero-shot's outputs	You are a helpful assistant. <reason>First, let's determine the amount of water in the original 10 liters of orange drink. Since it is two-thirds water, the amount of water in the orange drink is $(2/3) * 10 = 6.67$ liters. Next, we need to account for the spill. Since one liter of the orange drink is spilled, we subtract that from the total amount of water in the orange drink: $6.67 - 1 = 5.67$ liters. Now, let's calculate the amount of water in the 15 liters of pineapple drink. Since it is three-fifths water, the amount of water in the pineapple drink is $(3/5) * 15 = 9$ liters. Finally, we add the remaining water from the orange drink and the water from the pineapple drink to find the total amount of water in the remaining 24 liters: $5.67 + 9 = 14.67$ liters. Therefore, there are 14.67 liters of water in the remaining 24 liters.</reason><answer>14.67</answer>
TTRL's outputs	<reason>First, let's determine the amount of water in the original 10 liters of orange drink. Since it is two-thirds water, the amount of water in the orange drink is: Water in orange drink = $\frac{2}{3} \times 10 = \frac{20}{3} \approx 6.67$ liters. Next, let's determine the amount of water in the 15 liters of pineapple drink. Since it is three-fifths water, the amount of water in the pineapple drink is: Water in pineapple drink = $\frac{3}{5} \times 15 = 9$ liters. Since you spill 1 liter of the orange drink, the amount of orange drink left is: Remaining orange drink = $10 - 1 = 9$ liters. Now, let's calculate the amount of water in the remaining 9 liters of orange drink: Water in remaining orange drink = $\frac{2}{3} \times 9 = 6$ liters. Finally, the total amount of water in the combined mixture is the sum of the water from the remaining orange drink and the pineapple drink: Total water = $6 + 9 = 15$ liters. So, the total amount of water in the remaining 24 liters is 15 liters.</reason><answer>15</answer>
PaceReasoner's outputs	<reason>First, let's determine the amount of water in the original 10 liters of orange drink. Since the orange drink is two-thirds water, the amount of water in it is: Water in orange drink = $\frac{2}{3} \times 10 = \frac{20}{3} \approx 6.67$ liters. Next, let's determine the amount of water in the 15 liters of pineapple drink. Since the pineapple drink is three-fifths water, the amount of water in it is: Water in pineapple drink = $\frac{3}{5} \times 15 = 9$ liters. Since you spill 1 liter of the orange drink, the amount of orange drink left is: Remaining orange drink = $10 - 1 = 9$ liters. Now, let's calculate the amount of water in the remaining 9 liters of orange drink: Water in remaining orange drink = $\frac{2}{3} \times 9 = 6$ liters. The total amount of water in the combined mixture is the sum of the water in the remaining orange drink and the water in the pineapple drink: Total water = $6 + 9 = 15$ liters. The total volume of the mixture after spilling is: Total volume = $9 + 15 = 24$ liters. Therefore, the amount of water in the remaining 24 liters is: Water in the remaining mixture = 15liters.</reason><answer>15</answer>

Table 14: **Illustration of generated responses under LONG mode** on GSM8K, including zero-shot, TTRL and PaceReasoner. The base model is Qwen3-4B-Base.