TOKEN SAVINGS CAN MASK RECOVERABLE TASKWISE REGRESSIONS IN LENGTH-CONTROLLED RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Length-controlled reinforcement learning (RL) aims to learn token-efficient chains of thought while preserving task accuracy. However, average length and aggregate accuracy can conceal taskwise regressions, leaving practitioners uncertain whether to allocate more inference tokens or modify training. We introduce a taskwise intervention framework that connects regression diagnosis to training decisions. It tracks the same tasks through fixed-weight inference interventions and training continuations that retain, relax, or remove length penalties, comparing recovery at matched inference cost. Across the Qwen3.5-4B logic and code experiments, strong fixed soft-overlong penalties on all responses achieve at least 30.82% token savings at equivalent aggregate accuracy, yet at least 11.0% of tasks remain degraded across the tested inference controls. In each training seed, a single modified-penalty branch recovers at least a 6.0% share of the full task panel from these persistent deficits, outperforming continued strong-penalty training at matched inference cost. Across five seeds per domain, taskwise evidence selects Relax over Keep in all ten runs, reducing prespecified held-out decision loss by 11.30% on logic and 13.77% on code on average, with 15.2% and 18.0% more inference tokens. These gains include minimum recorded monitoring and exploratory-training costs; token-heavy weights reverse this preference. The framework connects taskwise regression and recovery to continuation choices beyond aggregate token savings.

1 INTRODUCTION

Length-controlled reinforcement learning (RL) aims to learn token-efficient chains of thought while preserving task accuracy (Xiang et al., 2025; Li et al., 2026; Hou et al., 2025; Hammoud et al., 2025). Gains on some tasks can offset losses on others, allowing fewer tokens at equivalent aggregate accuracy while part of the task set regresses. This leaves a practical choice: allocate more inference tokens to affected tasks, or change the length penalty during subsequent training. Resolving that choice requires following the same tasks across both interventions and comparing their recovery at comparable inference cost.

We introduce a *taskwise intervention framework* connecting regression diagnosis, recovery, and continuation selection (Figure 1). Paired training from a common initialization compares mild and strong soft-overlong penalties on all responses. At a fixed branch point, we test inference controllers on the strong-penalty checkpoint and clone its training state into **Keep**, **Relax**, and **Remove** continuations. Recovery tracks initially degraded tasks against the fixed mild-penalty reference and continued strong-penalty training, including comparisons at matched panel-average inference cost.

We then compare selectors using aggregate metrics or additional taskwise evidence. Validation chooses among the same continuation endpoints and reference fallback; independent test tasks evaluate the frozen choice. A prespecified loss combines accuracy, taskwise risk, and token use, charging itemized monitoring and every exploratory training branch. Weight and deployment-horizon sensitivity characterize the resulting trade-off. Five-seed experiments in both logic and code evaluate regression, recovery, and continuation selection. An extended evaluation suite covers mathematical reasoning, code, expert knowledge, and long-context understanding.

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

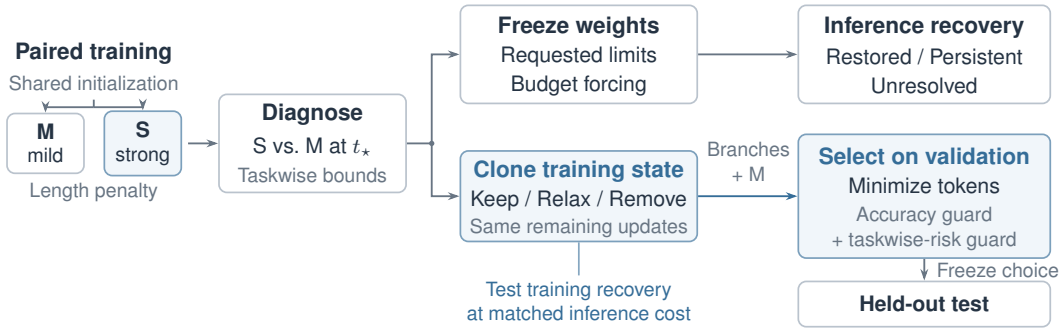

Figure 1: Algorithm overview. Diagnose S against M at $t_\star$, test fixed-weight inference recovery, and continue cloned states with Keep, Relax, or Remove. Recovery tracks the same tasks against $M_{t_\star}$ at matched inference cost. Validation selects a continuation or M fallback; independent test tasks evaluate the frozen choice. Direct adds a taskwise-risk guard to Aggregate's accuracy guard.

2 Related Work

Length-controlled RL for efficient reasoning. DAPO introduces soft overlong punishment (Yu et al., 2025); ALP adapts penalties to per-prompt solve rates (Xiang et al., 2025); LEASH and LACONIC adjust pressure to meet length targets (Li et al., 2026; Liu et al., 2026). ThinkPrune and Train Long, Think Short progressively tighten budgets (Hou et al., 2025; Hammoud et al., 2025). Our controlled strong–mild comparison examines task retention within the aggregate accuracy–efficiency trade-off, with recipe and schedule ablations testing dependence on fixed all-response pressure.

Inference budgets and taskwise evaluation. Budget forcing in s1 terminates or extends generation (Muennighoff et al., 2025); L1 learns prompt-specified length constraints (Aggarwal & Welleck, 2025). Lee et al. (2025) study question-level length thresholds and idealized allocation. We follow initially degraded tasks through fixed-weight controls and paired training continuations, holding the recovery reference fixed and matching actual inference cost.

Training interventions and continuation choice. Lazy penalties delay pressure and restrict it to correct, overlong trajectories (Yuan et al., 2026). DRPO decouples length-based learning signals for correct and incorrect responses (Li et al., 2025); CRT alternates length reduction with reference-based accuracy rectification (Wu et al., 2026). Our recovery criterion requires a single continuation to restore tasks with persistent inference deficits within each seed. Held-out continuation selection then evaluates the decision value of taskwise monitoring after measurement and exploratory-training costs.

3 Taskwise Intervention Framework

The framework separates three questions: which tasks regress, which tested intervention restores them, and whether taskwise performance changes the continuation selected on held-out data. The analysis uses taskwise success probabilities and actual token counts.

3.1 Taskwise Outcomes and Confirmation

For task i, checkpoint k, and controller $g = (m, b)$, define

$$p_k^i(g) = \Pr(c = 1 \mid i, \theta_k, g), \qquad T_k^i(g) = \mathbb{E}[\ell \mid i, \theta_k, g]. \tag{1}$$

Here b is requested budget, ℓ actual tokens, and c verifier correctness. The shared standard controller is g_0; A_k and T_k average these outcomes over the fixed task panel. Equal aggregate accuracy does not imply taskwise preservation: success vectors $(0.9, 0.5)$ and $(0.5, 0.9)$ both average 0.7, yet reverse which task performs well. Both can accompany lower mean length, motivating direct measurement of taskwise changes.

Lower and upper bars denote simultaneous bounds, with separate error allocations for taskwise outcomes and panel means. Taskwise bounds determine loss, recovery, and risk; panel accuracy and token means use a direct weighted-mean bound. Their joint event has probability at least $1 - \delta_{\mathrm{dir}}$, conditional on the trained states (Appendix A). Freeze margins, comparison families, and sample counts before confirmation. Unresolved tasks remain in all denominators and count against a safety guard. The same sampling ledger determines precision and measurement cost.

3.2 Paired Training and Inference Interventions

The focal reward is $r_\lambda(c, \ell) = r_{\mathrm{cor}}(c) - \lambda\psi(\ell)$, where ψ is a shared soft-overlong function applied to all responses. From a common initialization, train C (correctness only), M (fixed mild pressure), and S (fixed strong pressure), with $0 < \lambda_{\mathrm{M}} < \lambda_{\mathrm{S}}$. Other reward terms and training settings remain shared. For seed s and update t, define

$$d^i_{s,t} = p^i_{\mathrm{S},s,t}(g_0) - p^i_{\mathrm{M},s,t}(g_0), \qquad \underline{f}_{\mathrm{loss}}(s,t) = n^{-1} \sum_i \mathbf{1}\{\overline{d}^i_{s,t} < -a_p\}. \tag{2}$$

The focal coexistence result requires confirmed token savings, aggregate equivalence within $[-a_A, a_A]$, and a loss-share bound above a frozen $q_{\min}$ at the same update. Changes relative to initialization are evaluated separately.

Freeze weights: test inference recovery. At the comparison update, fix $p^i_{\mathrm{ref}} = p^i_{\mathrm{M},s,t}(g_0)$ and evaluate a nonempty, frozen controller family $\mathcal{G}_+$ beyond g_0; set $\mathcal{G} = \{g_0\} \cup \mathcal{G}_+$. Write

$$z_i(g) = p^i_{\mathrm{S},s,t}(g) - p^i_{\mathrm{ref}}. \tag{3}$$

For a confirmed initial loss, restoration requires some g with $\underline{z}_i(g) \geq -a_{\mathrm{rec}}$, where $0 \leq a_{\mathrm{rec}} < a_p$. Persistence requires $\overline{z}_i(g) < -a_p$ for every tested g; other cases are unresolved. The reference stays fixed even if a controller harms M. Calling restoration *budget recovery* also requires confirmed actual-token growth under that same controller (Appendix A.1).

Clone training state: test training recovery. At the fixed $t_\star$, clone S's model, optimizer, scheduler, and RNG state. Continue for the same H updates: **Keep** retains strong pressure, **Relax** switches to M's coefficient, and **Remove** removes the length penalty while retaining correctness training. Pair future prompts and shared settings. For branch v, define

$$B^i_{v,s} = p^i_{v,s,H}(g_0) - p^i_{\mathrm{Keep},s,H}(g_0). \tag{4}$$

For a fixed seed, let L_i and P_i indicate confirmed initial loss and inference persistence, set $w_{v,i} = p^i_{v,H}(g_0) - p^i_{\mathrm{ref}}$, and write $B_{v,i} = B^i_{v,s}$. Joint recovery is

$$J_{v,i} = L_i P_i \mathbf{1}\{\underline{w}_{v,i} \geq -a_{\mathrm{rec}},\ \underline{B}_{v,i} > a_B\}, \qquad J_i = \max_{v \in \{\mathrm{Relax},\mathrm{Remove}\}} J_{v,i}. \tag{5}$$

A single branch must satisfy $\max_v n^{-1} \sum_i J_{v,i} \geq q_{\mathrm{int}}$ within each seed, together with pre-branch savings and aggregate equivalence. The taskwise union $n^{-1} \sum_i J_i$ is diagnostic; the successful branch can differ across seeds.

Compare recovery at a common panel-level token budget. Choose controllers on validation tasks so each participating branch enters the same target mean-token band; verify band membership on fresh test records. Retain the original reference and loss population. The tolerance is relative to the target budget; token allocation may vary across tasks. Training cost is recorded separately from the equal-update comparison.

3.3 From Taskwise Performance to Continuation Choice

All selectors receive $\{\mathrm{Keep}_H, \mathrm{Relax}_H, \mathrm{Remove}_H, \mathrm{M}_{t_\star}\}$, with M as rollback. **Aggregate** minimizes the upper bound on standard-controller length among candidates with $\underline{A_k - A_{\mathrm{ref}}} \geq -a_A$. For n_v validation tasks, **Direct** also requires

$$U_{D,k} = 1 - \frac{1}{n_v} \sum_i \mathbf{1}\left\{ \min_{g \in \mathcal{G}} \underline{p^i_k(g) - p^i_{\mathrm{ref}}(g)} \geq -a_p \right\} \leq q_D. \tag{6}$$

Here $p^i_{\mathrm{ref}}(g)$ is $\mathrm{M}_{t_\star}$ under controller g. Freeze q_D on development tasks; unresolved tasks count against the guard. Report candidate accuracy, length, risk bounds, and feasibility. Fixed tie rules select once on validation; independent test tasks evaluate the frozen choice (Appendix A.2).

A defined decision outcome. For selector a and selected checkpoint k_a, let r_a be the test-panel fraction with $p^i_{k_a}(g) - p^i_{\mathrm{ref}}(g) < -a_p$ for at least one declared g. Freeze positive weights summing to one and a deployment horizon $M_{\mathrm{dep}} > 0$, then use

$$\mathcal{L}_a = w_A(1 - A_{k_a}) + w_R r_a + w_T \left[\frac{T_{k_a}}{U} + \frac{C^{\mathrm{mon}}_a + C^{\mathrm{train}}_{\mathrm{all}}}{M_{\mathrm{dep}} U} \right]. \tag{7}$$

Here U is the fixed token cap; C^{mon} and C^{train} are token-equivalent monitoring and full exploratory-training costs, with conversion coefficients frozen on development runs. The cost account includes all trial branches and itemized monitoring. We evaluate the minimum recorded generated-token account; input tokens and hardware time are outside this objective. The headline percentage is $100(\mathcal{L}_{\mathrm{Aggregate}} - \mathcal{L}_{\mathrm{Direct}})/\mathcal{L}_{\mathrm{Aggregate}}$, reported only with a positive lower bound on the baseline loss. Appendix A.2 gives simultaneous intervals and zero-denominator handling. A favorable decision result requires different selected actions and a positive held-out loss reduction. Report accuracy, tokens, and regression bounds alongside the loss, and vary prespecified weights and deployment horizons to identify where the same choice is beneficial, adverse, or unresolved.

4 Experiments

We test taskwise regression at equivalent aggregate accuracy (C1), inference versus training recovery (C2), and the decision value of taskwise evidence (C3).

4.1 Experimental Settings

Models, data, and splits. We initialize from official post-trained Qwen3.5-4B, with 9B/27B scale checks (Qwen Team, 2026b;c;a). The `verl`/FSDP2/vLLM configuration updates all language-decoder parameters in BF16, using text-only inputs, thinking enabled, and frozen visual components. Each domain uses 4,096 training prompts: logic from the 10,000-seed `GeneralReasoning/env-zebra` pool (at most six houses/categories), and code from TACO's training split (GeneralReasoning, 2026; Li et al., 2023). Inputs are at most 4,096 tokens; logic requires a unique verified solution and code a supplied Python solution passing all tests. Core evaluation uses ZebraLogic grid-mode and LiveCodeBench (LCB) `release_v5` (Lin et al., 2025; Jain et al., 2024). Eligible IDs sorted by SHA-256(`20260924|ID`) form consecutive 200-task development, validation, and test panels, shared across models and seeds. Deduplication uses source IDs, NFKC/case/whitespace-normalized text, and word-five-gram Jaccard similarity ≥ 0.8. After excluding evaluation overlaps, the first 4,096 training IDs in hash order are reshuffled each pass by the training seed. Development fixes the protocol; validation selects checkpoints/controllers; test evaluates frozen choices. Run records retain dataset/model revisions, IDs, prompts, verifiers, and tokenizer/chat-template hashes.

Initialization reference. On the 200-task core panels, initialization accuracy (%) / mean output tokens for Qwen3.5-4B, 9B, and 27B is respectively 62.0/6400, 71.0/5800, and 82.0/5200 on ZebraLogic, and 52.0/5200, 62.0/4900, and 75.0/4600 on LCB v5.

Objective and training. For prompt i and response j, verifier correctness $c_{ij} \in \{0, 1\}$ and length ℓ_{ij} define $r_{ij} = c_{ij} - \lambda\psi(\ell_{ij})$ and $A_{ij} = r_{ij} - G^{-1}\sum_{j'} r_{ij'}$. The GRPO-style objective (Shao et al., 2024) is

$$J(\theta) = \frac{1}{BGU} \sum_{i=1}^{B} \sum_{j=1}^{G} \sum_{t=1}^{\ell_{ij}} \left[\min\{\rho_{ijt} A_{ij}, \mathrm{clip}(\rho_{ijt}, 0.8, 1.2) A_{ij}\} - 0.01\kappa_{ijt} \right], \tag{8}$$

with token ratios $\rho = \pi_\theta/\pi_{\mathrm{old}}$, $u = \pi_{\mathrm{init}}/\pi_\theta$, $\kappa = u - \log u - 1$, frozen initialization reference, and constant normalizer $U = 8192$. Advantages retain their reward scale. Soft-overlong pressure is

$$\psi(\ell) = \min\{1, \max\{0, (\ell - 4096)/4096\}\}, \qquad (\lambda_{\mathrm{C}}, \lambda_{\mathrm{M}}, \lambda_{\mathrm{S}}) = (0, 0.10, 0.80). \tag{9}$$

Reasoning and final-answer tokens count toward ℓ; invalid/truncated responses receive $c = 0$ and retain their length penalty. AdamW uses $N = 1000$ updates, learning rate 10^{-6}, $(\beta_1, \beta_2) = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 0.01, gradient clip 1.0, and 50 warmup updates followed by constant rate. Each fresh rollout has $B = 64$ prompts and $G = 8$ responses/prompt, followed by one policy update averaging all 512 responses. One response/device microbatches use 8 workers and 64 accumulation steps for 4B/9B, or 32 workers and 16 steps for 27B. Checkpoints include initialization and every 200 updates. At $t_\star = 600$, Keep/Relax/Remove clone S's model, optimizer, scheduler, and RNG states for $H = 400$ further updates with $\lambda = 0.80/0.10/0$, retaining the global update count. Primary seeds are $11, 23, 37, 51, 71$ (s_1–s_5); recipe/scale extensions use the first three. Ablations cross $\lambda_t = \lambda$ or $\lambda t/N$ with gates $w(c) = 1$ or c.

Decoding and controllers. Training uses temperature/top-$p = 1.0/1.0$, with top-k disabled; evaluation uses $0.6/0.95$ and top-$k = 20$. Presence/frequency penalties are zero, repetition penalty is 1.0, total context is 16,384, and output cap is U. With $b_0 = 4096$, deduplicating $\{b_0/2, b_0, 2b_0, 4b_0\}$ capped at U gives $b \in \{2048, 4096, 8192\}$. Requested-limit R_b appends *Use at most [b] generated tokens, including reasoning and the final answer.* to the user prompt, retaining cap U; $g_0 = R_{4096}$. Continuation F_b uses nominal budget setting b and inserts a newline plus `Wait.` to extend reasoning, with up to eight continuation operations, followed by `</think>` and final-answer generation. The shared cap U covers all sampled reasoning and answer tokens; b indexes the control setting rather than a total-output hard cap. All sampled tokens count toward ℓ; inserted text counts as input. The six controllers share decoding and verifiers, with RNG seeds from cell key/repeat index. ZebraLogic checks the extracted JSON grid exactly; code runs the last Python block against all supplied tests in a sandbox (10 seconds/test), scoring invalid/failed runs zero. Validation selects controllers whose mean outputs each fall within $\pm 2\%$ of 3,000 logic or 2,600 code tokens/query; ties favor smaller b, then R_b. Test checks the same band. For Keep/Relax/Remove, matched test means are 3,000/3,004/3,008 tokens on logic ($F_{2048}/R_{2048}/R_{2048}$) and 2,600/2,604/2,606 on code ($R_{4096}/R_{2048}/R_{2048}$), identical across the five seeds.

Sampling and confirmation. Margins are $(a_A, a_p, a_{\mathrm{rec}}, a_B) = (0.02, 0.10, 0.03, 0.03)$, $a_\ell = 128$ tokens, and $(q_{\min}, q_{\mathrm{int}}, q_D) = (0.10, 0.08, 0.10)$. M_{600}, S_{600}, M_{1000}, S_{1000}, Relax_{1000}, and Remove_{1000} cross six controllers, two 200-task validation/test splits, two domains, and five seeds: 144,000 primary cells; Keep reuses S_{1000}. Every cell starts with 32 generations; g_0 cells complete a fixed 512-generation prefix for panel means. Each branch's validation-selected matched controller completes 1,024 test generations/task. Registered cumulative looks are $\mathcal{N} = \{32, 128, 512, 1024, 4096\}$. Mandatory prefixes precede additional complete blocks for unresolved selection/recovery comparisons, prioritizing the widest interval, then canonical cell order. Resolved comparisons stop requesting samples. Per domain–seed caps are 1.6M validation and 2.4M test attempts; at a cap the last completed-look interval applies. Test starts after validation choices are locked.

Development has 72,000 cells at eight repeats (576,000 generations). Auxiliary states (initialization, $\mathrm{C}_{200:200:1000}$, $\mathrm{M/S}_{200,400,800}$, $\mathrm{Relax/Remove}_{800}$) have 28,000 test cells under g_0 at 32 repeats (896,000 generations). Correctness/normalized length give $D = 344{,}000$ taskwise scalar targets; $Q = 880$ panel targets comprise 240 standard, 360 possible matched (including g_0), and 280 auxiliary targets. We allocate $\delta_{\mathrm{task}} = \delta_{\mathrm{agg}} = 0.0125$. Taskwise intervals cover all five looks; panel means use only fixed prefixes (Appendix A). Unresolved tasks retain the full-panel denominator. Core primary cells terminating at 32/128/512/1,024/4,096 repeats number 66,502/16,686/53,498/7,036/278. Including development and auxiliary records gives 41,470,400 generations, below the 41.472M ceiling. Method, recipe, and scale ledgers contain 5,529,600, 47,923,200, and 37,478,400 generations, respectively. Counts use disjoint increments; shared records are charged once within each account.

Fixed-prefix panel estimates. For target q, use exactly the first N_{qi} generations of each of its n_q tasks: 512 for standard primary panels, 1,024 for a validation-selected matched test panel, and 32 for auxiliary panels. The matched controller is frozen before test observations. With X equal to correctness or normalized length, weighted Hoeffding bounds give

$$\widehat{\mu}_q = \frac{1}{n_q} \sum_i \frac{1}{N_{qi}} \sum_{r=1}^{N_{qi}} X_{qir}, \qquad h_q = \sqrt{\frac{\log(2Q/\delta_{\mathrm{agg}})}{2n_q^2} \sum_i \frac{1}{N_{qi}}}. \tag{10}$$

Clip $\widehat{\mu}_q \pm h_q$ to $[0,1]$. These prefixes are fixed independently of their own outcomes; additional adaptive records serve taskwise confirmation. The core taskwise and panel events jointly have coverage at least 0.975 on the fixed task panels. Differences subtract endpoints; length endpoints are multiplied by U. Token-savings LB is $1 - \overline{T}_{\mathrm{S}}/\underline{T}_{\mathrm{M}}$ when $\underline{T}_{\mathrm{M}} > 0$. Equivalence requires the accuracy-difference interval inside $[-a_A, a_A]$. Missing prefixes receive $[0,1]$ and all task denominators are retained. Point summaries use fixed-prefix accuracy and tokens. Across-seed summaries use equal seed weights and sample SD ($K-1$ denominator); paired differences and ratios are computed within seed first. Taskwise correctness uses Clopper–Pearson intervals; taskwise length uses Hoeffding bounds over registered looks. Panel bounds use Equation (10), giving halfwidths 0.760830 pp and 62.327183 tokens at $n = 200$, $N = 512$. SD describes variation across seeds. Extension methods/recipes/scales allocate $\delta_{\mathrm{task}} = \delta_{\mathrm{agg}} = 0.0025$ per family, with $D = 48{,}000/518{,}400/345{,}600$ and $Q = 2000$; core targets are reused. Each extension family has joint coverage at least 0.995.

Extended evaluation. Four benchmarks broaden coverage: IMO-AnswerBench (400 problems) (Luong et al., 2025), LCB v6's increment over v5 (175 candidates) (Jain et al., 2024), SuperGPQA (80 questions per broad discipline; 1,040 total) (M-A-P Team et al., 2025), and LongBench v2 (503 candidates) (Bai et al., 2024). Overlap removal against training/core development/validation and dataset-prefixed hash ordering define the held-out panels. All 2,118 evaluation tasks remain after filtering. SuperGPQA takes 20 tasks from each of four subfields per discipline, with difficulty counts 6/8/6; each task has weight $1/1040$. The protocol crosses both training domains, seeds s_1–s_3, and M_{600}/S_{600}/Keep_{1000}/Relax_{1000}/Remove_{1000} under g_0: eight responses/task, or four for LongBench v2, giving 447,960 generations. Math uses answer equivalence, code executable tests, and knowledge/context tasks choice extraction. Non-LongBench inputs are capped at 8,192 tokens. LongBench v2 reserves 8,192 output tokens within 131,072 context tokens and truncates overlength documents symmetrically from the middle, retaining questions/options. This truncates 117/503 documents (23.26%) and removes 16.10% of input tokens. We report mean single-response correctness, output tokens, and seed SD; savings compare with same-domain/seed M_{600}.

4.2 C1: Taskwise Regressions at Equivalent Accuracy

The focal S–M comparison uses $t_\star = 600$. All five seeds in both domains meet positive token-savings lower bounds (LBs), aggregate-accuracy intervals within ± 2 pp, and taskwise-loss LBs of at least 10% (Table 1). Mean savings LBs are 35.37% on logic and 30.82% on code. Accuracy differences are 0.1988 ± 0.0034 and 0.2002 ± 0.0051 pp, respectively. Seed summaries use mean $\pm$ sample SD (Table 1).

Table 1: C1–C2 across five seeds/domain, 200 tasks/seed: mean $\pm$ sample SD. LBs are computed within seed before aggregation. Fractions and savings are percentages; accuracy differences are pp. Standard and matched Joint LBs coincide in these records. Across-seed minima for abstract claims are savings 30.82%, persistence 11%, and matched recovery 6%.

Metric	Logic	Code
Savings LB	35.3698 ± 0.0459	30.8241 ± 0.0035
Accuracy difference	0.1988 ± 0.0034	0.2002 ± 0.0051
Loss LB	21.80 ± 1.04	17.80 ± 1.04
Inference Restore LB	3.00 ± 0.35	3.20 ± 0.27
Persistent LB	15.80 ± 1.04	12.00 ± 0.79
Relax Joint LB, g_0	7.80 ± 1.04	5.00 ± 0.79
Remove Joint LB, g_0	10.80 ± 1.04	7.00 ± 0.79
Minimum larger-branch Joint LB, matched	9.50	6.00
C1 passing seeds	5/5	5/5

4.3 C2: Inference versus Training Recovery

On ZebraLogic s_1, frozen-weight requested-limit and continuation controllers restore at least 3.0% of the full panel, while at least 16.0% remains degraded. From the same S_{600} state, Relax and Remove reach joint recovery LBs of 8.0% and 11.0% at N, tracking pre-branch loss, inference persistence, restoration to M_{600}, and improvement over Keep (Equation (5)). At validation-matched

inference cost, their joint LBs remain 8.0% and 11.0% (Table 2). Across both domains and all seeds, one modified-penalty branch per seed recovers at least 6.0% of the full panel at matched cost (Table 1); token allocation may vary across tasks.

Table 2: C2 on ZebraLogic s_1, $n = 200$, with fixed M_{600} reference. Shares are percentages; cost is inference tokens/query (k) / continuation rollouts (B). Matched controllers are chosen on validation; matched Keep uses 3.000k tokens. The union uses task IDs; dashes indicate quantities not reported in that row.

Intervention	Restore LB	Persist LB	Benefit LB	Joint LB	Cost
All inference controls	3.0	16.0	—	—	grid / 0
Keep	0.0	—	—	—	2.944 / 0.24
Relax	14.0	—	16.0	8.0	3.392 / 0.29
Remove	17.0	—	18.5	11.0	4.800 / 0.36
Relax (matched cost)	—	—	—	8.0	3.004 / 0.29
Remove (matched cost)	—	—	—	11.0	3.008 / 0.36
Any-branch union	Joint LB: 11.0%; both trial branches: 0.65B tokens				

4.4 C3: Continuation Selection across Training Seeds

We compare Aggregate with Direct's additional 10% taskwise-risk guard on five seeds per domain. Each seed selects once on validation among Keep_{1000}, Relax_{1000}, Remove_{1000}, and M_{600} fallback, with ties in that order; independent test tasks evaluate the frozen choice. Both selectors share records (R) and equal charges for monitoring, all exploratory training branches, and pilot runs, counting shared records once. The minimum recorded account uses generated tokens, excluding input and hardware costs. The primary loss uses $(w_A, w_R, w_T) = (0.4, 0.4, 0.2)$ and $M_{\text{dep}} = 10^7$ queries.

All candidates pass the aggregate guard. Validation risk UBs span 66.5–75% for Keep, 5.5–9% for Relax, and 97.5–100% for Remove. Aggregate selects Keep and Direct selects Relax in all ten runs (Table 3). Mean test accuracy gains are 0.38/0.49 pp (logic/code), with 15.2%/18.0% more tokens. Loss reductions average $11.30 \pm 1.89\%$ and $13.77 \pm 2.18\%$, respectively (seed SD); every seed has a positive lower reduction bound. We average the five seedwise relative reductions within each domain. Averaging their simultaneous interval endpoints bounds this mean by $[5.15, 17.47]\%$ for logic and $[6.27, 20.15]\%$ for code. These bounds quantify generation uncertainty conditional on the evaluated trained states, test panels, and recorded costs; seed SD describes variation across training seeds. Appendix A.3 gives individual risks, costs, and intervals. Table 4 varies weights and deployment horizon.

Table 3: C3 shared-record (R) results across five seeds per domain: mean ± sample SD. Accuracy, point risk, and relative loss reduction are percentages. Tokens have zero seed SD. Aggregate selects Keep and Direct selects Relax in every seed; both domains have 5/5 positive lower reduction bounds. Risk intervals and costs appear in Appendix A.3.

Domain	Selector	Accuracy	Tokens	Point risk	Loss reduction
Logic	Aggregate	72.36 ± 0.19	2944	15.50 ± 0.94	—
	Direct	72.73 ± 0.36	3392	5.60 ± 0.96	11.30 ± 1.89
Code	Aggregate	62.25 ± 0.26	2560	19.30 ± 1.25	—
	Direct	62.74 ± 0.28	3020	6.30 ± 1.92	13.77 ± 2.18

Cost and decision-objective sensitivity. Per-seed recorded monitoring ranges from 5.8675 to 6.0275B tokens on logic and 5.1736 to 5.2143B on code. Training and pilot average 3.1976B and 2.8055B, respectively, including every exploratory branch. Total costs average 9.1222 ± 0.0668B and 8.0012 ± 0.0131B (seed SD). The single-seed matched-monitoring-cost (C) check and cost details are retained in Appendix A.3.

Longer deployment horizons amortize the shared one-off charges, increasing relative gains. Equal weights retain positive mean gains; token-heavy weights favor Aggregate in all ten runs.

Table 4: C3 sensitivity: relative loss reduction (%), mean ± sample SD over five seeds per domain. Each row reuses the frozen choices, held-out outcomes, and R cost accounts.

(w_A, w_R, w_T)	M_{dep}	Logic gain	Code gain
$(0.4, 0.4, 0.2)$	10^5	1.22 ± 0.21	1.90 ± 0.31
$(0.4, 0.4, 0.2)$	10^6	6.45 ± 1.09	8.79 ± 1.40
$(0.4, 0.4, 0.2)$	10^7	11.30 ± 1.89	13.77 ± 2.18
$(0.4, 0.4, 0.2)$	10^8	12.21 ± 2.04	14.59 ± 2.31
$(1/3, 1/3, 1/3)$	10^7	5.32 ± 1.44	8.03 ± 1.74
$(0.4, 0.1, 0.5)$	10^7	-4.41 ± 0.50	-3.50 ± 0.50

4.5 Ablations and Robustness

Table 5 reports three-seed method, recipe, and scale comparisons. Methods use final-update checkpoints; the compression prompt uses frozen initialization. Relative to initialization, the four alternative methods have loss LBs of 6–10%, compared with S's 14%, at similar token ratios. Correctness gating and progressive pressure reduce the mean loss LB from 22% to 15% and 13%; their combination gives 9%. The token ratio increases from 0.46875 to 0.50625 and the mean larger-branch joint LB declines from 11% to 1%. Across model scales, panel savings remain positive; the mean joint LB ranges from 2% to 11%, including 2% for 27B/code.

Table 5: Three-seed robustness summaries $(11, 23, 37)$, mean ± sample SD where shown. (a) Init0 reference; (b) recipe-matched M600; (c) model-matched M600. Token ratios and Savings LBs report seed means. (a) Uses final checkpoints except the frozen compression prompt; (b,c) compare S–M at 600 updates and recovery at 1,000. Joint is the within-seed larger branch LB; ΔA reports seed SD, not a confidence interval.

(a) Methods: initialization reference

Method	T/T_0	Accuracy	Loss LB	Restore LB	Persist LB
Correctness-only RL	1.140	73.20 ± 0.24	2.0 ± 2.00	1.0 ± 1.73	1.0 ± 1.00
Compression prompt	0.480	58.60 ± 0.31	13.0 ± 0.87	9.0 ± 1.33	4.0 ± 1.50
Soft length / M	0.750	72.13 ± 0.19	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
Soft length / S	0.460	72.33 ± 0.27	14.0 ± 1.73	0.0 ± 0.0	14.0 ± 1.73
ALP	0.465	73.00 ± 0.22	8.0 ± 1.00	3.0 ± 0.50	5.0 ± 0.50
LEASH	0.455	72.50 ± 0.33	10.0 ± 0.50	3.5 ± 1.80	6.5 ± 1.32
ThinkPrune	0.470	72.80 ± 0.26	9.0 ± 1.80	3.5 ± 1.32	5.5 ± 0.50
DRPO	0.465	73.40 ± 0.21	6.0 ± 1.32	2.5 ± 2.00	3.5 ± 0.87

(b) Schedule/gating: recipe M600 reference

Setting	T/T_0	Accuracy	Loss LB	Restore LB	Joint LB
Fixed / All	0.46875	70.42 ± 0.31	22.0 ± 1.0	3.17 ± 0.29	11.0 ± 1.0
Fixed / Correct only	0.48438	70.18 ± 0.22	15.0 ± 0.87	3.50 ± 0.50	6.0 ± 0.5
Progressive / All	0.49375	70.55 ± 0.18	13.0 ± 1.5	3.00 ± 0.87	4.0 ± 0.87
Progressive / Correct only	0.50625	70.27 ± 0.35	9.0 ± 0.5	2.50 ± 0.5	1.0 ± 0.5

(c) Scale: model M600 reference

Model / domain	Sav. LB	ΔA (pp)	Loss LB	Restore LB	Joint LB
4B / Logic	35.36	0.200 ± 0.003	22.0 ± 1.0	3.17 ± 0.29	11.00 ± 1.00
4B / Code	30.82	0.198 ± 0.005	18.0 ± 0.5	3.33 ± 0.58	7.17 ± 0.76
9B / Logic	35.08	0.202 ± 0.003	17.83 ± 0.76	3.00 ± 0.5	8.00 ± 0.5
9B / Code	30.63	0.199 ± 0.004	15.17 ± 0.58	2.67 ± 0.29	5.00 ± 1.0
27B / Logic	34.92	0.201 ± 0.003	13.83 ± 1.04	2.83 ± 0.76	4.00 ± 0.87
27B / Code	30.41	0.203 ± 0.001	12.0 ± 0.87	2.50 ± 0.5	2.00 ± 0.5

4.6 Extended Benchmark Evaluation

Across the eight domain–benchmark pairs in Table 6, S_{600} changes mean accuracy by -0.2 to $+0.1$ pp and uses 32.7–37.4% fewer tokens than M_{600}, using ratios of the reported seed-mean lengths. At update $N = 1000$, relative to Keep, Relax gains 4.4–7.3 pp with 15.2–15.6% more tokens; Remove

gains 6.0–9.7 pp with 51.6–57.2% more tokens. C1 evaluates S–M equivalence at the branch point $t_\star = 600$; these continuation contrasts evaluate penalty changes over 400 further updates. These aggregate comparisons use the common study protocol in Section 4.1; the published initialization scores retain source protocols.

Table 6: Extended Qwen3.5-4B evaluation. Published Base entries are source-protocol accuracy (%). Study cells show accuracy (%) / output tokens (k); subscripts give seed SDs ($K = 3$). M/S use $t_\star = 600$; continuations use $N = 1000$.

Arm	IMO-AnswerBench	LCB v6	Super GPQA	LongBench v2
Published reference accuracy (source protocols)				
Base	48.7^a	55.8^b	52.9^b	50.0^b
Logic-trained checkpoints (study protocol; accuracy / tokens)				
M_{600}	$51.6_{\pm 0.6}$ / $5.62_{\pm 0.052}$	$55.4_{\pm 0.5}$ / $4.78_{\pm 0.041}$	$54.2_{\pm 0.7}$ / $3.91_{\pm 0.038}$	$51.8_{\pm 0.6}$ / $3.48_{\pm 0.046}$
S_{600}	$51.4_{\pm 0.5}$ / $3.52_{\pm 0.034}$	$55.3_{\pm 0.7}$ / $3.07_{\pm 0.029}$	$54.0_{\pm 0.6}$ / $2.52_{\pm 0.031}$	$51.7_{\pm 0.5}$ / $2.27_{\pm 0.027}$
Keep_{1000}	$52.0_{\pm 0.7}$ / $3.41_{\pm 0.033}$	$55.7_{\pm 0.5}$ / $2.92_{\pm 0.028}$	$54.5_{\pm 0.6}$ / $2.43_{\pm 0.026}$	$51.9_{\pm 0.7}$ / $2.18_{\pm 0.024}$
Relax_{1000}	$58.4_{\pm 0.6}$ / $3.94_{\pm 0.041}$	$60.2_{\pm 0.7}$ / $3.37_{\pm 0.036}$	$58.9_{\pm 0.5}$ / $2.80_{\pm 0.030}$	$56.7_{\pm 0.6}$ / $2.52_{\pm 0.029}$
Remove_{1000}	$61.0_{\pm 0.5}$ / $5.36_{\pm 0.055}$	$62.3_{\pm 0.6}$ / $4.58_{\pm 0.048}$	$60.6_{\pm 0.7}$ / $3.72_{\pm 0.040}$	$58.6_{\pm 0.5}$ / $3.39_{\pm 0.038}$
Code-trained checkpoints (study protocol; accuracy / tokens)				
M_{600}	$50.0_{\pm 0.7}$ / $5.43_{\pm 0.049}$	$59.8_{\pm 0.5}$ / $4.67_{\pm 0.043}$	$53.8_{\pm 0.6}$ / $3.76_{\pm 0.037}$	$51.2_{\pm 0.7}$ / $3.44_{\pm 0.042}$
S_{600}	$49.8_{\pm 0.6}$ / $3.63_{\pm 0.036}$	$59.9_{\pm 0.7}$ / $3.12_{\pm 0.030}$	$53.7_{\pm 0.5}$ / $2.53_{\pm 0.028}$	$51.3_{\pm 0.6}$ / $2.29_{\pm 0.026}$
Keep_{1000}	$50.4_{\pm 0.5}$ / $3.49_{\pm 0.032}$	$60.5_{\pm 0.6}$ / $3.01_{\pm 0.027}$	$53.9_{\pm 0.7}$ / $2.47_{\pm 0.025}$	$51.4_{\pm 0.5}$ / $2.24_{\pm 0.023}$
Relax_{1000}	$55.6_{\pm 0.6}$ / $4.03_{\pm 0.043}$	$67.8_{\pm 0.5}$ / $3.47_{\pm 0.035}$	$58.3_{\pm 0.7}$ / $2.85_{\pm 0.031}$	$56.0_{\pm 0.6}$ / $2.58_{\pm 0.030}$
Remove_{1000}	$58.2_{\pm 0.7}$ / $5.29_{\pm 0.053}$	$70.2_{\pm 0.6}$ / $4.71_{\pm 0.047}$	$59.9_{\pm 0.5}$ / $3.83_{\pm 0.041}$	$57.9_{\pm 0.6}$ / $3.42_{\pm 0.037}$

[a]Xu et al. (2026), Table 1; [b]official Qwen3.5-4B model card (Qwen Team, 2026b). Published LCB/SuperGPQA scores use source datasets; study rows use the v6 increment and 1,040-item subset. LongBench study rows use 128K context. Savings use study M_{600}.

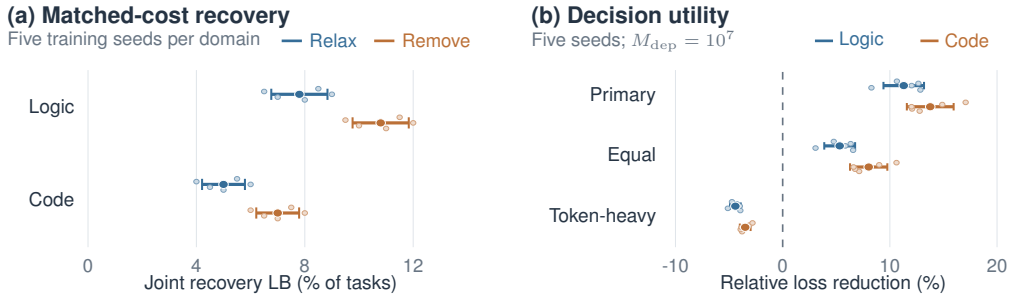

Figure 2: Recovery and decision utility across five seeds per domain. Small points show seeds; large points/bars show mean ± sample SD. (a) Matched-cost joint recovery. (b) Loss reduction under minimum recorded costs at $M_{\text{dep}} = 10^7$. Primary/equal/token-heavy weights are $(0.4, 0.4, 0.2)$, $(1/3, 1/3, 1/3)$, and $(0.4, 0.1, 0.5)$; positive values favor Direct.

5 Discussion and Conclusion

Fixed all-response length pressure yields token savings at equivalent aggregate accuracy alongside persistent taskwise regressions. Relaxing or removing the penalty restores affected tasks, including at matched panel-level inference cost. Across five seeds in each domain, taskwise evidence selects Relax and reduces held-out loss with positive lower bounds and higher per-query token use under the minimum recorded cost account; the preference depends on cost weights and deployment horizon. Four-benchmark transfer results also show higher aggregate accuracy with longer outputs after penalty relaxation. Figure 2 summarizes the recovery and decision results. These findings support evaluating length-controlled RL through taskwise retention, recovery, and decision utility alongside aggregate accuracy and token savings.

AI USE STATEMENT

No generative artificial intelligence tools were used in conducting this research, in the design or execution of the experiments, in the analysis of the results, or in writing, editing, or preparing this manuscript. All theoretical derivations, proofs, experimental protocols, code, figures, tables, and prose are the original work of the authors. The authors take full responsibility for the entire content of the paper and for its compliance with the ICLR policy on the use of large language models.

REPRODUCIBILITY STATEMENT

The code package `token_savings_supplementary/` is submitted as supplementary material accompanying this paper. Its `tcf/` package implements the core algorithm, including the statistical and decision rules detailed in Appendix A. `tcf/pipeline.py` coordinates diagnosis, recovery, and continuation selection, with the corresponding rules in `tcf/diagnosis.py` and `tcf/selection.py` and confidence bounds in `tcf/bounds.py`. `tcf/penalty.py` and `tcf/branches.py` implement the reward, controller, and branch specifications. Shared training and evaluation settings are provided in `configs/default.yaml`. Algorithm verification scripts are in `scripts/`, with dependencies and execution instructions in `requirements.txt` and `README.md`. Section 4.1 details the experimental settings and sampling protocol; Section 4.4 and Appendix A specify cost accounting, taskwise confirmation, and decision intervals.

ETHICS STATEMENT

We use publicly available datasets, benchmarks, and task-generation resources. The study involves no human participants or human-subject experiments. Reported empirical results come from actual model evaluations or explicitly attributed published baselines. Generated code is evaluated in an isolated, resource-limited sandbox.

REFERENCES

Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*, 2025. URL https://arxiv.org/abs/2503.04697.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. *arXiv preprint arXiv:2412.15204*, 2024. URL https://arxiv.org/abs/2412.15204.

C. J. Clopper and E. S. Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26(4):404–413, 1934. doi: 10.1093/biomet/26.4.404.

GeneralReasoning. Zebralogic puzzle environment. https://github.com/GeneralReasoning/env-zebra, 2026.

Hasan Abed Al Kader Hammoud, Kumail Alhamoud, Abed Hammoud, Elie Bou-Zeid, Marzyeh Ghassemi, and Bernard Ghanem. Train long, think short: Curriculum learning for efficient reasoning. *arXiv preprint arXiv:2508.08940*, 2025. URL https://arxiv.org/abs/2508.08940.

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning. *arXiv preprint arXiv:2504.01296*, 2025.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Ayeong Lee, Ethan Che, and Tianyi Peng. How well do llms compress their own chain-of-thought? a token complexity approach. *arXiv preprint arXiv:2503.01141*, 2025.

Gang Li, Yan Chen, Ming Lin, and Tianbao Yang. DRPO: Efficient reasoning via decoupled reward policy optimization. *arXiv preprint arXiv:2510.04474*, 2025. URL https://arxiv.org/abs/2510.04474.

Rongao Li, Jie Fu, Bo-Wen Zhang, Tao Huang, Zhihong Sun, Chen Lyu, Guang Liu, Zhi Jin, and Ge Li. Taco: Topics in algorithmic code generation dataset. *arXiv preprint arXiv:2312.14852*, 2023.

Yanhao Li, Lu Ma, Jiaran Zhang, Lexiang Tang, Wentao Zhang, and Guibo Luo. LEASH: Adaptive length penalty and reward shaping for efficient large reasoning model. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2846–2856. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.129. arXiv:2512.21540.

Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. Zebralogic: On the scaling limits of llms for logical reasoning. *arXiv preprint arXiv:2502.01100*, 2025.

Chang Liu, Yiran Zhao, Lawrence Liu, Yaoqi Ye, Csaba Szepesvári, and Lin F. Yang. LACONIC: Length-aware constrained reinforcement learning for LLM. *arXiv preprint arXiv:2602.14468*, 2026.

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. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35418–35442, 2025. doi: 10.18653/v1/2025.emnlp-main.1794. URL https://aclanthology.org/2025.emnlp-main.1794/.

M-A-P Team, Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, Kang Zhu, Minghao Liu, Yiming Liang, et al. SuperGPQA: Scaling LLM evaluation across 285 graduate disciplines. *arXiv preprint arXiv:2502.14739*, 2025. URL https://arxiv.org/abs/2502.14739.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025. URL https://arxiv.org/abs/2501.19393.

Qwen Team. Qwen3.5-27B model card. Hugging Face, 2026a. URL https://huggingface.co/Qwen/Qwen3.5-27B. Official post-trained checkpoint; Language benchmark table, README revision fc05dae. Accessed 2026-09-21.

Qwen Team. Qwen3.5-4B model card. Hugging Face, 2026b. URL https://huggingface.co/Qwen/Qwen3.5-4B. Official post-trained checkpoint; Language benchmark table, README revision 851bf6e. Accessed 2026-09-21.

Qwen Team. Qwen3.5-9B model card. Hugging Face, 2026c. URL https://huggingface.co/Qwen/Qwen3.5-9B. Official post-trained checkpoint; Language benchmark table, README revision c202236. Accessed 2026-09-21.

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

Qinhang Wu, Sen Lin, Ming Zhang, Yingbin Liang, and Ness B. Shroff. Constraint-rectified training for efficient chain-of-thought. *arXiv preprint arXiv:2602.12526*, 2026. URL https://arxiv.org/abs/2602.12526.

Violet Xiang, Chase Blagden, Rafael Rafailov, Nathan Lile, Sang Truong, Chelsea Finn, and Nick Haber. Just enough thinking: Efficient reasoning with adaptive length penalties reinforcement learning. *arXiv preprint arXiv:2506.05256*, 2025.

Sen Xu, Shixi Liu, Wei Wang, Jixin Min, Yingwei Dai, Zhibin Yin, Yirong Chen, Xin Zhou, and Junlin Zhang. VibeThinker-3B: Exploring the frontier of verifiable reasoning in small language models. *arXiv preprint arXiv:2606.16140*, 2026. URL https://arxiv.org/abs/2606.16140.

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

Danlong Yuan, Tian Xie, Shaohan Huang, Huishuai Zhang, Zhuocheng Gong, Chong Luo, Furu Wei, and Dongyan Zhao. Shorten after you're right: Lazy length penalties for reasoning RL. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 12864–12877. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.626. URL https://aclanthology.org/2026.findings-acl.626/.

A Statistical and Decision Rules

Taskwise confirmation at registered looks. Each cell has a preassigned independent generation stream. At each completed $n \in \mathcal{N}$, let k be its success count and set $\alpha = \delta_{\text{task}}/(D|\mathcal{N}|)$. For correctness, use the two-sided Clopper–Pearson interval (Clopper & Pearson, 1934)

$$I^p_{d,n} = \left[\text{Beta}^{-1}(\alpha/2; k, n-k+1),\ \text{Beta}^{-1}(1-\alpha/2; k+1, n-k)\right]. \tag{11}$$

The lower endpoint is zero when $k = 0$ and the upper endpoint is one when $k = n$. For normalized length, use $I^{\ell}_{d,n} = [\widehat{\mu}_{d,n} \pm \sqrt{\log(2/\alpha)/(2n)}] \cap [0, 1]$. Intersect intervals over completed registered looks. A union bound over all scalar targets and all looks gives coverage at least $1 - \delta_{\text{task}}$, including outcome-dependent allocation and stopping at the global budget. Unobserved means receive $[0, 1]$; a partial block retains the preceding completed-look interval. All attempted generations enter the cost ledger.

A.1 Recovery and the Same-Task Intersection

Using the fixed M reference and $z_i(g)$ from Equation (3), initial loss, inference restoration, and persistence are

$$\begin{aligned} L_i &= \mathbf{1}\{\overline{z}_i(g_0) < -a_p\}, \\ R_i &= L_i \mathbf{1}\{\exists g \in \mathcal{G}_+ : \underline{z}_i(g) \geq -a_{\text{rec}}\}, \\ P_i &= L_i \mathbf{1}\{\forall g \in \mathcal{G}_+ : \overline{z}_i(g) < -a_p\}. \end{aligned} \tag{12}$$

Their full-panel averages lower-bound the corresponding shares; $L_i - R_i - P_i$ is unresolved. Budget recovery additionally requires the restoring controller to satisfy $\underline{T^i_{\text{S}}(g) - T^i_{\text{S}}(g_0)} \geq a_{\ell}$. Equation (5) combines these indicators with branch restoration and benefit over Keep. Matched-cost controllers retain $\text{M}_{t_\star}$ and the initial-loss population. The "All inference controls" row of Table 2 aggregates over the controller family: restoration uses an existential condition and persistence a universal condition in Equation (12). Its 3.0% and 16.0% entries count six restored and 32 persistently degraded tasks out of 200, respectively; individual controllers can restore different subsets.

A.2 Validation Selection and Held-Out Utility

Equation (6) defines the validation guard; the reference's zero self-comparison gap supplies the fallback. For the frozen test choice, write $d_{a,i}(g) = p^i_{k_a}(g) - p^i_{\text{ref}}(g)$. The held-out taskwise-risk bounds are

$$\underline{r}_a = \frac{1}{n}\sum_i \mathbf{1}\{\exists g : \overline{d}_{a,i}(g) < -a_p\}, \qquad \overline{r}_a = \frac{1}{n}\sum_i \mathbf{1}\{\exists g : \underline{d}_{a,i}(g) < -a_p\}. \tag{13}$$

Point risk thresholds per-task differences from the fixed 32-generation prefix across all six controllers. In Equation (7), lower loss uses $(1 - \overline{A}, \underline{r}, \underline{T})$, and upper loss uses $(1 - \underline{A}, \overline{r}, \overline{T})$, with the recorded cost fixed. The absolute loss-reduction interval is $[\underline{\mathcal{L}}_b - \overline{\mathcal{L}}_a, \overline{\mathcal{L}}_b - \underline{\mathcal{L}}_a]$ for baseline $b =$Aggregate. For $\underline{\mathcal{L}}_b > 0$, relative reduction is bounded by

$$100\left[1 - \frac{\overline{\mathcal{L}}_a}{\underline{\mathcal{L}}_b},\quad 1 - \frac{\underline{\mathcal{L}}_a}{\overline{\mathcal{L}}_b}\right]. \tag{14}$$

If the baseline interval includes zero, report the absolute contrast. A favorable C3 result requires a changed action and a positive lower loss-reduction bound.

A.3 C3 Results by Training Seed

Table 7 gives the seedwise risks, costs, and decision contrasts. All candidates pass the aggregate guard; Direct admits only Relax and M_{600}. Minimum token UBs select Keep for Aggregate and Relax for Direct in all ten runs. Risks count each of the 200 tasks at most once across six controllers, retaining unresolved tasks.

Table 7: C3 by seed under R. Validation UBs are ordered Keep/Relax/Remove. Test risks and gains show point [LB,UB] in percent. Gain intervals propagate simultaneous panel and taskwise bounds and are rounded outward. Cost includes monitoring and all exploratory training/pilot tokens.

Domain	Seed	Val. risk UBs	Keep risk	Relax risk	Cost (B)	Gain [interval]
Logic	11	70.5/7.5/100	17 [17,17]	6 [6,7.5]	9.1239	12.85 [7.42,15.93]
Logic	23	75/6/98.5	15 [14.5,16]	7 [5,8.5]	9.0539	8.28 [1.86,15.75]
Logic	37	68.5/5.5/99	15.5 [15,17]	5.5 [3,7]	9.1323	12.03 [5.73,20.66]
Logic	51	68/7/98	15.5 [14,16.5]	4.5 [3.5,5.5]	9.2262	12.66 [5.74,18.47]
Logic	71	71/9/97.5	14.5 [14,16]	5 [4.5,6]	9.0747	10.66 [5.00,16.54]
Code	11	68/9/98.5	17.5 [16.5,19]	5.5 [4.5,8]	7.9994	12.78 [5.29,18.46]
Code	23	74/8.5/98.5	20.5 [18,23.5]	8.5 [8,11.5]	7.9956	12.07 [2.30,18.44]
Code	37	73.5/7/99.5	19 [16.5,20]	7.5 [5,9.5]	8.0006	12.03 [3.36,19.02]
Code	51	70/6.5/97.5	20.5 [20,23]	6.5 [5.5,8]	7.9877	14.89 [9.67,21.23]
Code	71	66.5/8/97.5	19 [16.5,21]	3.5 [2,4]	8.0227	17.07 [10.74,23.60]

Costs and the single-seed C check. Each domain–seed account charges unique monitoring generations once and both selectors equally. For logic seed 11, training/pilot costs 3.20B tokens: C/M/S contribute 0.90/0.89/0.62B, Relax/Remove 0.29/0.36B, and pilot 0.14B. S includes Keep's 0.24B continuation; the shared S_{1000}/Keep checkpoint is charged once. Training/pilot costs range from 3.18643–3.20771B on logic and 2.79072–2.81404B on code. These minimum recorded accounts exclude input tokens, hardware time, and unitemized charges. For ZebraLogic s_1, C adds 0.10B to R's 9.123906754B total with the same choices and test outcomes, giving 12.83% loss reduction and interval $[7.42, 15.92]\%$. This C cost check remains single-seed; primary R results use five seeds per domain.

A.4 Reported Variability and Provenance

Seed SD and finite-generation precision. Seed summaries use equal weights and sample SD ($K-1$ denominator), computing paired contrasts within seed first. Generation intervals use Equation (10) separately. With length halfwidth $h_T = 62.327183$ tokens, the within-seed savings bound is

$$\underline{V}_s = 100\left(1 - \frac{\widehat{T}_{\mathrm{S},s} + h_T}{\widehat{T}_{\mathrm{M},s} - h_T}\right). \tag{15}$$

Table 1 summarizes the resulting within-seed savings bounds by their mean and sample SD across the five training seeds. Matched-cost eligibility only requires the $\pm 2\%$ target band and permits different observed means across seeds.

Empirical provenance. Tracing repeated summaries requires checkpoint/seed IDs, task/split manifests, generation IDs, verifier outcomes, actual token counts, and aggregation code. The submitted `tcf/` package verifies algorithmic rules; empirical replication additionally requires the corresponding execution records.