DUET: Optimize Token-Budget Allocation for Reinforcement Learning with Verifiable Rewards

Haoyu Hu[1,*] **Xuandong Zhao**[2] **Xuhai "Orson" Xu**[3,†] **Nori Jacoby**[1,†]
[1]Cornell University [2]University of California, Berkeley [3]Columbia University
[*]hh824@cornell.edu [†]Joint senior authors

Abstract

Reinforcement learning with verifiable rewards (RLVR) generates hundreds of thousands of tokens per training step, with rollout generation dominating the computational cost. The overall token budget can be controlled along two main dimensions: (i) deciding which prompts to allocate rollouts to, and (ii) deciding how long each rollout should be. Prior work has generally controlled only one of these dimensions at a time. We show that jointly tuning both decisions under a shared compute budget improves both reasoning quality and wall-clock training time. We instantiate this view as **DU**al-controlled tok**E**n alloca**T**ion (DUET), a computationally efficient layer over GRPO that uses a lightweight pre-rollout surrogate of prompt informativeness to set how many rollouts each prompt receives, and a marker-gated abort rule with importance reweighting to set when to stop them. On Qwen3-1.7B trained on MATH, DUET outperforms full-budget GRPO and the other three budget-aware baseline methods. DUET's advantage further generalizes to other benchmarks across math and coding, and is on par with the best baseline on the scientific Q&A domain, while also achieving a $1.62\times$ wall-clock speedup. More notably, using only 50% of the token budget, DUET still outperforms all baseline methods at their full budget, achieving an even higher $2.51\times$ speedup over full-budget GRPO. We verify the high performance of DUET on other backbone LLMs, including Qwen3-4B to 14B and Llama-3.2-3B-Instruct. Notably, the gap between DUET and the strongest baseline *widens* as the budget tightens, contrary to the usual pattern in which efficient methods trade off quality as compute decreases. More broadly, these results suggest that DUET budget-aware control strategies are valuable not only for accelerating training, but also for improving the quality of the learning signal. The code and data setup are publicly available at https://github.com/Haoyu-Hu/DUET

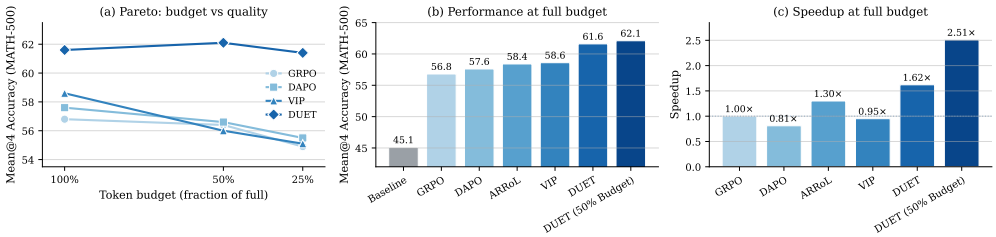


Figure 1: DUET dominates the budget–accuracy frontier on Qwen3-1.7B-base trained on MATH. (a) Accuracy versus rollout budget for four methods at three budget points; DUET dominates at every point. (b) Accuracy at full budget across baselines; DUET at half budget exceeds full-budget GRPO. (c) Wall-clock speedup at full budget, normalized to GRPO; DUET at half budget runs $2.51\times$ faster than full-budget GRPO.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

1 Introduction

Reasoning-centric large language models (LLMs) such as DeepSeek-R1 [11], Light-R1 [39], and Qwen3 [42] have advanced state-of-the-art performance on mathematical and code-reasoning benchmarks [13, 6, 4], and the post-training engine behind these results is reinforcement learning with verifiable rewards (RLVR) [31]. The recent rise of RLVR has been closely associated with GRPO [31], whose recipe is straightforward: at each training step, draw several candidate solutions per prompt, score them with a fast verifier or reward model, and update the policy using advantages computed relative to the rewards within that group. The recipe is also expensive: a single training step requires generating hundreds of thousands of tokens. Meanwhile, the verifier or reward model themselves are significantly cheaper [41, 47]. Thus, as reasoning chains grow longer and models grow larger, the cost of an RLVR run is increasingly dominated by *the total number of tokens* per training step.

A critical question follows: *how do we reduce the token cost without losing model performance?* Recent work (Appendix A.1) optimizes one of two natural degrees of freedom: either deciding which prompts should receive rollouts [44, 47, 26, 18] or deciding how long each rollout should continue [41]. Imposing a fixed length cap on every rollout is strictly suboptimal: aggressive truncation rewards the policy for committing to its first guess and silently eliminates the long chains of thought that underlie frontier reasoning performance [40, 45, 44]. In this work we coordinate the prompt-level decision of how many rollouts to draw with the within-rollout decision of when to stop them, all under a single shared compute budget. This simple coordination delivers substantial gains over either dimension alone.

Classical statistics offers a clean solution to precisely this two-decision budget-allocation problem. Stratified Monte-Carlo integration [25, 27] and budgeted best-arm identification [7, 3] both treat "how many samples" and "when to stop sampling" as two faces of the same problem, governed by a single shared budget-pressure signal. We bring this perspective to RLVR with **DU**al-controlled tok**E**n alloca**T**ion (**DUET**). At each training step, DUET estimates each prompt's informativeness from recent within-prompt gradient variance and allocates more rollouts to prompts that provide a stronger learning signal. During generation, DUET evaluates each rollout against two adaptive thresholds derived from the distribution of previously generated rollout lengths: rollouts that run too long without producing an answer are terminated as likely dead-end chains, while successful rollouts are stopped shortly after an answer appears. A single budget-pressure signal couples these two controls and enforces the compute budget at every step: when tokens are scarce, both rollout allocation and length gates tighten; when tokens are plentiful, both relax. DUET requires no auxiliary models trained online, adds less than one percent overhead to a standard GRPO step, and operates within the curated multi-pass training regime that dominates published RLVR practice [44, 46, 39, 22, 38].

Our proposal yields substantial performance gains (Figure 1). On Qwen3-1.7B-base [42] trained on MATH [13], DUET at half the rollout budget matches or beats every full-budget baseline (GRPO, DAPO, ARRoL, VIP) on all five reasoning benchmarks (MATH-500, GSM8K, AIME-2024, HumanEval, GPQA-Diamond) at $2.51\times$ wall-clock speedup over full-budget GRPO. At full budget, DUET surpasses these three budget-aware baselines (DAPO, ARRoL, VIP) on the same benchmarks at $1.62\times$ speedup. These gains are largely maintained from 4B to 14B scale and also transfer to a Llama-3.2-3B-Instruct [10]. We further verified the robustness and efficiency of DUET via comprehensive ablation studies including asynchronous RL methods and different rollout allocation heuristics. These findings show that these gains are not merely a flat speedup.

DUET learns which prompts merit compute without supervision. The per-prompt rollout count begins uniformly at 4 and expands to a range as wide as $[1, 32]$ as the controller's online variance estimate accumulates, with accuracy increasing fastest during this fan-out phase. *DUET removes two forms of token waste under a single shared budget.* A marker-gated abort terminates rollouts that exceed the policy's natural stopping length without producing a parsable answer, and trims successful rollouts shortly after the answer appears; both decisions are coupled by the same budget-pressure signal that enforces the budget. *DUET's advantage grows as the budget tightens.* Existing efficient-RLVR methods tend to trade quality for compute savings, so their gains shrink at lower budgets; DUET exhibits the opposite pattern, with its largest margin over the next-best baseline at quarter budget.

2 Problem Setup

At each step t, an RLVR trainer draws a batch $\mathcal{B}_t$ of M prompts, generates n_q rollouts per prompt q from the current policy π_θ, scores them with a verifier, and updates the parameters θ to maximise

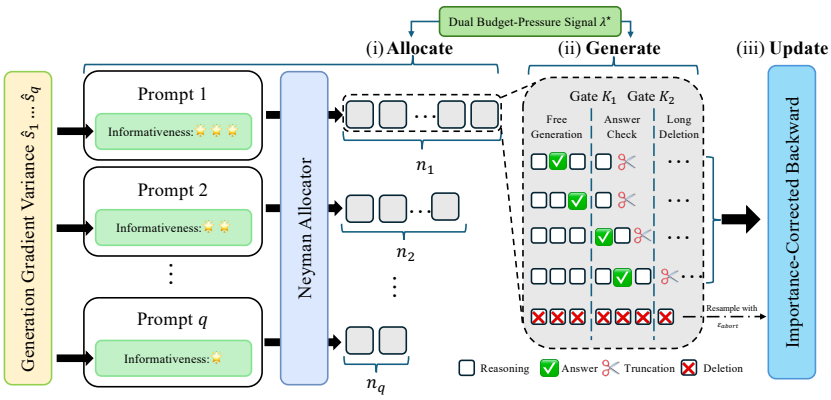


Figure 2: One DUET training step. *Allocate*: each prompt receives an informativeness score $\hat{s}_q$ from previous training contributions, and the cost-weighted Neyman rule sets the per-prompt rollout count n_q. *Generate*: a marker-gated rule terminates a rollout shortly after an answer marker fires and aborts a marker-less rollout past a quantile threshold. *Update*: kept rollouts feed an importance-corrected, gradient-masked backward step. The dual variable $\lambda^\star$ is the budget-pressure signal that closes the shared token budget across all three phases.

$J(\theta) = \mathbb{E}_{q\sim\mu,\, y\sim\pi_\theta(\cdot|q)}[R(q,y)]$. The dominant computational cost of this step is the rollout generation: with expected rollout length $\bar{L}_q$, the per-step token count is $\sum_q n_q \bar{L}_q$, which we cap at a budget B. Under independent rollouts within a prompt, the trace variance of the standard policy-gradient estimator has the textbook form

$$V(n) \;=\; \sum_q \sigma_q^2 / n_q, \tag{1}$$

where σ_q^2 is the within-prompt variance of the per-rollout gradient contribution $Z_{q,i} := A_{q,i}\,\nabla_\theta \log \pi_\theta(y_{q,i} \mid q)$, where $A_{q,i}$ is the advantage of rollout i for prompt q.

The control problem is then simple: pick a per-prompt rollout count n_q and a per-rollout stopping rule under the token budget. The two choices must keep $V(n)$ small and the gradient unbiased for the standard policy gradient. We restrict to the multi-pass curated-corpus regime ($E \geq 2$ epochs over a fixed prompt pool), which covers most published RLVR practice [44, 46, 39, 15, 22, 38]. Statements below are written for an action-independent-baseline surrogate $g^{\mathrm{AIB}}(\theta_t)$; the residual coupling under the GRPO group-normalised advantages used in the implementation is reported empirically and analysed in Appendix A.2.6.

3 DUal-controlled tokEn allocaTion (DUET)

DUET is a thin three-phase layer over the standard GRPO loop. *Allocate* (Figure 2(i)) decides how many rollouts each prompt receives. *Generate* (Figure 2(ii)) decides when each rollout stops. *Update* (Figure 2(iii)) reweights the kept rollouts so the gradient stays unbiased. A single budget-pressure signal $\lambda^\star$ ties the three phases together and constrains the token budget at every step. The controller's overhead (a bisection over λ, a streaming marker check, and an importance reweighting on kept rollouts) adds less than one percent to per-step wall-clock at the model scales we run. The full pseudocode is Algorithm 1 in Appendix A.10.

3.1 Allocate: Cost-Weighted Neyman with an Online Surrogate

This step concerns how many rollouts each prompt should receive under the per-step token budget B. Stratified Monte-Carlo integration [27] provides exactly the recipe with variance minimization: sample more from prompts whose rollouts disagree more. The classical Neyman optimum is $n_q \propto \sigma_q$ when every rollout costs the same [25]. Rollouts in RLVR have prompt-dependent expected length $\bar{L}_q$, so the answer acquires a square-root cost factor.

Theorem 1 (Cost-weighted Neyman allocation; proof in Appendix A.2.3)**.** *For independent rollouts, among positive allocations n satisfying $\sum_q n_q \bar{L}_q = B$, the variance $V(n) = \sum_q \sigma_q^2/n_q$ is uniquely*

minimised by

$$n_q^\star \;=\; \frac{\sigma_q}{\sqrt{\lambda^\star\, \bar{L}_q}}, \tag{2}$$

where $\lambda^\star > 0$ is the unique Lagrange multiplier closing the budget, and the minimum is $V^\star = \lambda^\star B$.

Theorem 1 is the basis for the following steps. The numerator σ_q tells the controller to spend more rollouts where the within-prompt rollouts disagree the most, since those are the prompts whose gradient signal is updating the policy. Its denominator $\sqrt{\bar{L}_q}$ favours cheap prompts, since a long rollout costs more tokens. The Lagrange multiplier $\lambda^\star$ is the shared budget-pressure signal: when tokens are scarce $\lambda^\star$ is large and every $n_q^\star$ shrinks; when tokens are plentiful $\lambda^\star$ is small and every $n_q^\star$ grows. Once the abort policy of Section 3.2 fixes the realized lengths $\bar{L}_q(\theta)$, a unique $\lambda^\star(\theta)$ constrains the budget under the abort policy (Proposition 1 in Appendix A.2.4). The same scalar reappears in the abort gate of phase (ii), which is the mechanism that ties the two phases under one shared budget.

The variance signal σ_q is unobservable at allocation time. We use a per-prompt running estimate: at the end of every step, every prompt q with $n_q \geq 2$ kept-not-aborted rollouts contributes

$$\hat{\sigma}_{q,t}^{\text{obs}} \;:=\; \mathrm{std}_i\Big(A_{q,i} \sum_\ell \log \pi_{\theta_t}(y_{q,i,\ell} \mid q, y_{q,i,<\ell})\Big), \tag{3}$$

which updates a running mean $\hat{\sigma}_q^{\text{obs}}$. The plug-in surrogate is $\hat{s}_q := \max\{s_{\text{floor}}, \hat{\sigma}_q^{\text{obs}}\}$ with a small cold-start floor for prompts not yet observed; the cost factor $\hat{L}_q$ is the per-prompt running mean over kept-rollout lengths. Substituting $\hat{s}_q$ into Theorem 1 keeps the allocation rule unchanged. The resulting plug-in cost is bounded above by a single divergence between surrogate and truth that decays as $1/B$ (Theorem 2 in Appendix A.2.5). The empirical difference between $\hat{s}_q$ and the within-step variance estimate is therefore the natural diagnostic of allocation quality. The realised allocation is

$$n_q \;=\; \max\left(n_{\min},\; \mathrm{round}\left(\frac{\hat{s}_q}{\sqrt{\lambda^\star\, \hat{L}_q}}\right)\right), \tag{4}$$

where $\lambda^\star$ solves $\sum_{k \in \mathcal{B}_t} n_k(\lambda)\, \hat{L}_k = B$ via a few bisection iterations warm-started from $\lambda_{t-1}^\star$, and $n_{\min}$ enforces a per-prompt minimum.

3.2 Generate: Marker-Gated Abort with ε-Keep

This step concerns controlling when each rollout should stop. A fixed-length cap is the simplest answer and the worst one: it penalizes a coherent partial chain-of-thought identically to a wrong answer, and silently kills the long traces that drive frontier reasoning [40, 45, 44]. Instead, we use an answer-specific stopping criterion. The marker $m(y^{1:t}) \in \{0, 1\}$ detects a parsable answer in the prefix, which reuses the machine veriable format defined by RLVR. For example, it could be a closed numerical box for math, a closing code fence for code, or an answer span for short-form QA.

Two thresholds set the schedule. A polling threshold K_1 is the earliest token at which the marker check turns on; before K_1 the marker is rarely needed and inspection wastes cycles. An abort threshold K_2 is a high quantile (e.g. 80 %) of the policy's natural-stopping length with our assumption that a rollout reaching K_2 without a marker is unlikely to land an answer. Both thresholds adapt online from a sliding window of recent kept rollouts. With $\tau_q^{\text{marker}} := \inf\{t \geq K_1 : m(y_{q,i,1:t}) = 1\}$ and a small grace window G for a clean tail,

$$\tau_q^{\text{abort}} \;:=\; \min\big\{\tau_q^{\text{marker}} + G,\; K_2 + G\big\}. \tag{5}$$

A rollout that emits a marker by τ_q^{abort} is kept and contributes its gradient at propensity 1; this is the answer-fired path between K_1 and K_2 in phase (ii). A marker-less rollout cannot simply be dropped, since dropping silently removes the slowest, least-confident prompts from the gradient and biases learning toward the prompts the policy already solves. We apply a minimal importance correction instead: with probability $\varepsilon_{\text{abort}} \in (0, 1)$ the marker-less rollout is continued to natural EOS (end of sentence, subject to a large safeguard length limit) and its contribution is reweighted

by $1/\varepsilon_{\text{abort}}$, and otherwise the rollout is aborted and masked out. Let $I^{\text{abort}}_{q,i} \in \{0,1\}$ be the abort-and-mask indicator and $p^{\text{abort}}_{q,i} \in \{1, \varepsilon_{\text{abort}}\}$ the kept-rollout propensity. The per-rollout estimator $\hat{h}_{q,i} := (1 - I^{\text{abort}}_{q,i})\, Z_{q,i}/p^{\text{abort}}_{q,i}$ is exactly unbiased for the natural-rollout contribution under an action-independent baseline. Its trace second moment lifts by at most $(1 - p^{\text{marker}}_q)(1 - \varepsilon_{\text{abort}})/\varepsilon_{\text{abort}}$ times the marker-less stratum's second moment $\sigma^2_{\text{m-less},q}$ (Theorem 3 in Appendix A.2.6). The surcharge is the price the controller pays to keep the slow prompts in the gradient. Under realisable optima it goes to zero asymptotically (Appendix A.2.7); under KL regularisation it stays small but non-zero, and in our runs it never exceeds $1.5\times$ the natural-rollout second moment (Figure 6(c)).

3.3 Update: Importance-Corrected, Gradient-Masked Backward

The last step runs the gradient update on the kept rollouts. Writing $w_{q,i} := (1 - I^{\text{abort}}_{q,i})/(s^{\text{pre}}_q\, p^{\text{abort}}_{q,i})$ with $s^{\text{pre}}_q := \text{clip}(n_q/\bar{n}_t, \varepsilon_{\text{pre}}, 1)$ a per-prompt stratification factor and $m_{q,i,\ell}$ the post-abort response mask, where $\bar{n}_t = M^{-1}\sum_q n_q$ is the batch-mean rollout count at step t, and ε_{pre} is a small lower clip floor,

$$\mathcal{L}_t(\theta) = -\frac{1}{N_t}\sum_{q,i,\ell} w_{q,i} m_{q,i,\ell} \min\{r_{q,i,\ell}(\theta)A_{q,i}, \text{clip}(r_{q,i,\ell}(\theta), 1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}})\, A_{q,i}\}\,. \tag{6}$$

where

$$r_{q,i,\ell}(\theta) = \frac{\pi_\theta(y_{q,i,\ell} \mid q, y_{q,i,<\ell})}{\pi_{\theta_{\text{old}}}(y_{q,i,\ell} \mid q, y_{q,i,<\ell})}, N_t = \sum_{q,i,\ell} m_{q,i,\ell}, \epsilon_{\text{clip}} = 0.2 \tag{7}$$

The response mask removes tokens after an abort. Group-normalised advantages and token-mean aggregation match the standard GRPO loss aggregator; DAPO retains its own $0.20/0.28$ clipping setting. The rollout count from (4) and the abort rule from (5) remain coupled through the same dual $\lambda^\star$ that closes the budget.

4 Results

4.1 Experimental Setup

Models and Datasets. Training uses Qwen3-1.7B-base, Qwen3-4B-base [42], and Llama-3.2-3B-Instruct [10] as a cross-family check, all with full-parameter fine-tuning on the Hendrycks MATH [13] 7.5k corpus. Evaluation covers MATH-500 [13], GSM8K [6], and AIME-2024 for math; HumanEval [4] for code generation; and GPQA-Diamond [29] as a cross-domain reasoning probe. Full configurations appear in Appendix A.3. Due to the limitation of computing resources, we only test smaller models in this study.

Baselines and Protocol. We compare DUET against the non-budgeted reference (GRPO [31]) and three efficient-RLVR baselines: DAPO [44] (dynamic-sampling), ARRoL [41] (within-rollout-pruning), and VIP [26] (budget-adaptive), each at full, half and quarter rollout budget. We additionally evaluate APRIL [48], an asynchronous partial-rollout method that reduces idle time from long-tail responses, and GRESO [47], a reward-history prompt-skipping method, in Appendix A.5. We used 8 as the full rollout budget per prompt, which is the most common choice for RL training [47, 41], and set the hard limit of the budget by capping the number of rollouts for methods without budget settings (GRPO, DAPO, ARRoL). All cells run on the same software and hardware. Specifically, we use a server with 8×H100 80GB GPUs and verl 0.4.1 [32] with vLLM 0.9.2 [19]. We set top-p to 0.95, the max prompt length to 512 tokens, and the max generation length to $3{,}072$ tokens. We train the model for 232 steps ($\sim$4 epochs) on the MATH dataset, with each step using 128 prompts. In the main content, we evaluate every 30 training steps, generate 4 rollouts per prompt at temperature 0.9, and report the best mean@4 across checkpoints. Multi-seed trajectories, held-out checks, and scale-up runs are reported as robustness ablations in Appendix A.5.

Validating the rollout-budget knob. GRPO, DAPO, and VIP all expose a per-prompt rollout count as their primary budget control. Halving this knob halves the per-step generated-token cost on our setup (Figure 9 in Appendix A.7), so "rollout budget b" in this paper is a faithful proxy for "token budget bB". ARRoL is the exception: its κ knob is a soft post-generation drop probability, not a hard

Table 1: Accuracy (mean@4, %) and GRPO-normalized speedup on Qwen3-1.7B-base, Qwen3-4B-base, and Llama-3.2-3B-Instruct trained on MATH. **Bold** = best per column within each model section; underline = second-best (ties at the rounded value are all underlined). Multi-seed and scale-up robustness checks are in Appendix A.5, and full budgeted results are in Table 16.

Method	MATH-500	GSM8K	AIME-24	HumanEval	GPQA-D	Speedup
Qwen3-1.7B-base						
Baseline	45.1	60.4	5.8	34.5	22.2	–
+GRPO	56.8	78.0	6.7	47.6	28.8	1×
+DAPO	57.6	77.4	7.5	46.8	**29.1**	0.81×
+ARRoL	58.4	79.6	8.3	47.7	26.7	1.30×
+VIP	58.6	78.1	5.8	48.9	27.8	0.95×
+DUET (50% Budget)	**62.1**	84.0	10.8	51.7	**29.1**	**2.51**×
+DUET	61.6	**84.7**	**12.5**	**52.7**	29.0	1.62×
Qwen3-4B-base						
Baseline	54.3	58.5	10.0	62.8	28.8	–
+GRPO	74.5	90.1	13.3	73.6	**38.5**	1×
+DAPO	73.2	89.9	**18.3**	75.2	37.4	0.63×
+ARRoL	73.9	90.3	11.7	73.6	34.5	1.19×
+VIP	74.4	90.1	16.7	75.2	36.8	0.75×
+DUET (50% Budget)	76.7	92.3	13.3	**77.0**	38.1	**2.38**×
+DUET	**78.2**	**93.9**	14.2	**77.0**	38.3	1.43×
Llama-3.2-3B-Instruct						
Baseline	37.1	60.3	3.3	50.5	20.5	–
+GRPO	44.4	75.7	7.5	52.8	**25.4**	1×
+DAPO	44.1	75.5	7.5	52.7	24.5	0.82×
+ARRoL	45.3	74.8	6.7	52.8	25.3	1.13×
+VIP	43.6	75.2	**9.2**	**53.0**	24.9	0.95×
+DUET (50% Budget)	44.5	72.8	6.7	52.9	23.2	**2.04**×
+DUET	**45.6**	**75.9**	7.5	52.1	25.3	1.26×

token cap, so we compare ARRoL only at its default operating point and audit its budget semantics in Appendix A.7.

4.2 Main Results

DUET matches full-budget GRPO quality at half the rollouts and half the wall-clock, and the headline transfers across model scale and model family. Table 1 reports mean@4 accuracy on the five evaluation benchmarks at two Qwen3 scales and on the Llama-3.2-3B-Instruct cross-family check. At half rollout budget, DUET matches or exceeds full-budget GRPO on four of five benchmarks at both Qwen3 scales while running $2.51\times$ (1.7B) and $2.38\times$ (4B) faster, so a practitioner reaches the same model in less than half the wall-clock. At full budget DUET leads four of five benchmarks at 1.7B and three of five at 4B, with a $1.62\times$ and $1.43\times$ speedup over same-rollout-count GRPO. The single column where vanilla GRPO leads on Qwen3 is GPQA-Diamond, an out-of-distribution probe for math-trained policies, and the gap is at most 0.5%. On Llama-3.2-3B-Instruct the RLHF prior compresses headroom across all methods, but full-budget DUET still leads on MATH-500 and GSM8K, and half-budget DUET runs $2.04\times$ faster than GRPO with at most a 1% gap on every benchmark. HumanEval lifts by at most 2.5% across every method on this family, a property of the strong RLHF prior rather than the training rule.

4.3 Wall-Clock Efficiency

DUET shifts the entire speed-quality Pareto outward, not just one operating point. Figure 3(a) shows that DUET at full budget runs faster than full-budget GRPO from the first validation checkpoint onward, and DUET at half budget sits below every baseline throughout training, stabilising near a $2.5\times$ ratio once the online length-quantile estimator settles. Figure 3(b) sweeps the rollout budget at three points and reports speedup against full-budget GRPO: DUET dominates at every point, reaches $3.4\times$ at quarter budget, and pulls away from a much shallower GRPO/DAPO/VIP frontier. The separation is structural rather than incremental. ARRoL's pruner saves only the small slice of generated tokens past its mid-decode inspection point. DAPO's post-rollout filter pays the full generation cost on the rollouts it eventually drops. DUET's marker-gated abort acts before generation reaches the model's natural maximum length, and the savings compound across the heterogeneous allocation that the Neyman optimum prescribes. The Pareto curve does not just slide down. The *slope* changes, so the practitioner's case for DUET strengthens at every tighter operating point.

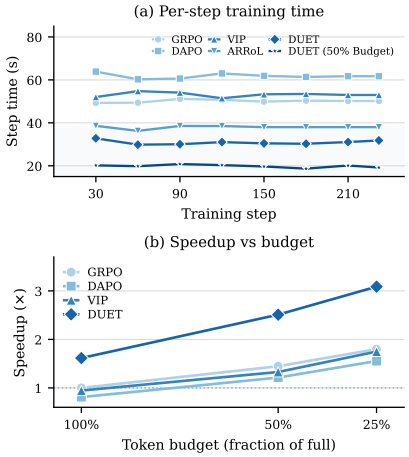


Figure 3: Wall-clock efficiency on Qwen3-1.7B / MATH. **(a)** Per-step training time. **(b)** Speedup over full-budget GRPO across rollout budgets.

4.4 Ablations

We first isolate two questions on DUET ablations with Qwen3-1.7B: do the allocator and the abort each pull their weight, and how sensitive is the performance to hyperparameters?

Table 2: Ablation of the cost-weighted Neyman allocator and the marker-gated abort on Qwen3-1.7B / MATH at 50% rollout budget, ε_{abort}=0.05.

Configuration	MATH-500	GSM8K	AIME-24	HumanEval	GPQA-D	Speedup
Baseline	45.1	60.4	5.8	34.5	22.2	–
GRPO	56.4	75.8	6.7	46.9	27.4	1×
+ allocator only	<u>61.7</u>	**84.1**	**12.5**	<u>51.0</u>	28.3	1.30×
+ abort only	61.3	82.1	**12.5**	49.4	<u>28.6</u>	**1.93**×
+ both (DUET)	**62.1**	<u>84.0</u>	<u>10.8</u>	**51.7**	**29.1**	<u>1.73×</u>

The allocator and the abort specialise; combining them is Pareto-better than either alone. Table 2 disables one ingredient at a time. The allocator alone (uniform abort off) delivers $+5.3\%$ on MATH-500 over the rollout-matched anchor at $1.30\times$ speedup, and is the per-benchmark winner on GSM8K and AIME-24; the abort alone (uniform n_q, abort on) delivers $+4.9\%$ at $1.93\times$ speedup. The two ingredients are not interchangeable: the allocator concentrates compute on prompts whose rollouts move the gradient, while the abort cuts the marker-less tail of every rollout. Combining them into the full controller produces $+5.7\%$ on MATH-500 (a Pareto improvement on quality) at $1.73\times$ speedup, and the highest HumanEval and GPQA-Diamond scores of the four cells. The allocator is the quality lever, the abort is the wall-clock lever, and the joint controller takes both at once.

The broader evidence points in the same direction. Three-seed final and best-checkpoint trajectories prove that DUET converge not only faster but also higher. The performance maintains on two held-out benchmarks, Minerva-Math [21] and Olympiad-Bench [12]. The Qwen3-8B/14B scale-up runs retain DUET's advantage on six of seven directional benchmarks. As DUET is presented as a synchronous method, we tested asynchronous method (i.e., APRIL [48]) in Table 8. We also compared other heuristics methods include GRESO [47] to test whether DUET's advantage extends beyond its joint allocator–abort controller. The heuristic comparison in Table 12 indicates that replacing gradient variance with verifier accuracy does not consistently improve the quality–speed trade-off. These

Table 3: Sensitivity to the abort floor $\varepsilon_{\text{abort}}$ on Qwen3-1.7B / MATH at 50% rollout budget. All cells share the same controller, surrogate, and seed; only $\varepsilon_{\text{abort}}$ varies.

$\varepsilon_{\text{abort}}$	MATH-500	GSM8K	AIME-24	HumanEval	GPQA-D	Speedup
0.01	61.4	83.4	<u>12.5</u>	<u>51.1</u>	27.1	**3.13**×
0.05 (default)	**62.1**	**84.0**	10.8	**51.7**	<u>29.1</u>	<u>2.51</u>×
0.10	<u>62.0</u>	83.1	**13.3**	50.6	27.5	2.05×
0.20	61.9	<u>83.9</u>	<u>12.5</u>	<u>51.1</u>	**30.5**	1.94×

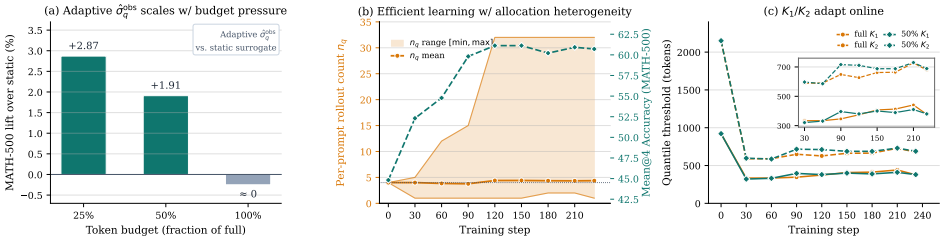


Figure 4: DUET's emergent properties on Qwen3-1.7B / MATH. **(a)** Adaptive-surrogate gain over the cold-start static surrogate scales monotonically with budget pressure. **(b)** Allocation heterogeneity emerges online (with DUET b=50%, ε_{abort}=0.01): the per-prompt rollout count n_q starts uniform at 4 (shaded band collapses to a point) and fans out to a range as wide as $[1, 32]$ as the online surrogate accumulates observations; MATH-500 mean@4 (right axis, dashed teal) lifts in lockstep with the fan-out, with the steepest accuracy gain coinciding with the widening of the allocation distribution. **(c)** The abort thresholds K_1 and K_2 adapt online without manual scheduling: both collapse from cold-start defaults to the policy's realised p_{30} / p_{80} within 30 training steps, then track the natural-stopping length distribution as the policy's output tightens. Same trajectory at full budget (amber) and 50% budget (teal).

checks preserve the same quality–speed conclusion. The detailed trajectories, scale-up rows, and heuristic controls appear in Appendix A.5 (Tables 6, 7, 13–14, and 12).

The performance is set by the framework, not by hyperparameter tuning. Table 3 sweeps $\varepsilon_{\text{abort}} \in \{0.01, 0.05, 0.10, 0.20\}$ across two orders of magnitude. MATH-500 accuracy moves only 0.7% ($61.4 \rightarrow 62.1$), well inside the single-seed sampling band, and the rest of the benchmarks stay within roughly 1.4% end to end. Wall-clock cost moves monotonically: $\varepsilon = 0.01$ runs at $3.13\times$ vs full-budget GRPO and $\varepsilon = 0.20$ at $1.94\times$, with the default $\varepsilon = 0.05$ at the knee of the curve ($2.51\times$). $\varepsilon_{\text{abort}}$ therefore trades wall-clock for a small SNIPS-overhead margin; the quality of the trained model is determined by the cost-weighted Neyman allocation and the marker-gated abort, not by the value of $\varepsilon_{\text{abort}}$. We also investigate the influence of δ_{poll} (marker polling interval), W (detector prefix window), s_{floor} (minimum surrogate score), and the marker patterns (answer detector). Across these implementation choices, the advantage of DUET in its quality and speed, and the privilege of its allocator–abort design remain unchanged. This indicates that the default configuration is representative rather than a narrowly tuned point (Tables 9–11 in Appendix A.5). Overall, the result is not driven by a narrow hyperparameter choice.

4.5 Emergent Online Properties of the Controller

A method designed for efficiency need not display anything beyond a flat speedup. DUET, in our runs, surfaces three online behaviours that frame the controller as more than a budget knob (Figure 4).

(1) *The adaptive surrogate's value scales with budget pressure.* Replacing the cold-start static surrogate with the online running estimate $\hat{\sigma}_q^{\text{obs}}$ lifts MATH-500 by $+2.87\%$ at quarter budget, $+1.91\%$ at half budget, and ties at full budget (Figure 4(a)); the same monotone pattern holds on GSM8K (Appendix A.5).

(2) *Heterogeneous allocation is discovered, not engineered.* Starting from $n_q = 4$ uniform across the batch, the per-prompt count fans out to a range as wide as $[1, 32]$ within one epoch (Figure 4(b)); the fan-out coincides with the steepest segment of the validation-accuracy curve, so the controller's

internal difficulty estimate drives the model's external improvement.

(3) *The abort thresholds adapt to the policy without manual scheduling.* The early-skip K_1 and abort K_2 start from cold-start defaults near 921 and 2150 tokens and collapse to the policy's realised p_{30} and p_{80} within 30 training steps, tracking the natural-stopping length distribution as it tightens (Figure 4(c)). Two kinds of token waste are eliminated as a consequence: rollouts that drift past K_2 without producing a parsable answer are aborted, and rollouts that have already produced one are trimmed shortly after the marker fires. The same convergence shape holds at full and half budget, so the controller is not budget-tuned.

A closer look at what the controller spends its compute on (Figure 5) exposes two additional findings.

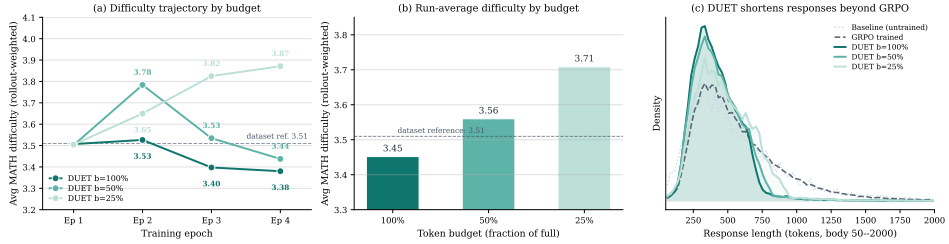


Figure 5: DUET's allocation internals on Qwen3-1.7B / MATH. **(a)** Per-epoch trajectory of rollout-weighted average difficulty at three budgets, against the dataset reference 3.51 (dashed). Full-budget DUET drifts *below* the reference (3.38); half-budget DUET concentrates sharply at epoch 2 (3.78) then relaxes back below the reference (3.44); only at quarter budget is the concentration sustained, climbing monotonically to 3.87. **(b)** Run-average difficulty across budgets. Tighter budget shifts the run average up. **(c)** Response-length density across DUET budgets, trained GRPO, and the untrained baseline. The baseline $\rightarrow$ GRPO shift is small (median $505 \rightarrow 511$, mean $735 \rightarrow 665$); GRPO $\rightarrow$ DUET shift is large (median $511 \rightarrow 330$).

(1) *Difficulty concentration is jointly shaped by budget pressure and training stage.* Across the run, the rollout-weighted average difficulty rises monotonically as the budget tightens: 3.45 at 100%, 3.56 at 50%, 3.71 at 25%, against a dataset reference of 3.51 (Figure 5(b)). The per-epoch trajectories tell the deeper story (Figure 5(a)). At full budget the controller has enough rollouts that it never needs to concentrate: average difficulty drifts from 3.51 at epoch 1 down to 3.38 at epoch 4, ending below the dataset reference. At half budget the controller concentrates sharply during the steepest learning segment (3.78 at epoch 2, the highest point in the run) and then relaxes back to 3.44 by epoch 4, also dipping below the dataset reference once the policy has absorbed the early hard-prompt signal. Only at quarter budget does the pressure force sustained concentration: difficulty climbs monotonically to 3.87 by epoch 4. The picture is of a controller that spends extra rollouts on hard prompts when the budget forces it to, and surrenders that concentration the moment the budget allows.

(2) *DUET also shifts the response-length distribution toward shorter rollouts, and the shift is method-driven rather than a generic RLVR effect* (Figure 5(c)). Trained GRPO (full 232-step run) sits almost on top of the untrained baseline (median $505 \rightarrow 511$ tokens, mean $735 \rightarrow 665$); the same RLVR loop with DUET on top collapses the median to roughly 330 tokens and removes most of the right tail beyond 750. The cost-weighted Neyman correction in Theorem 1 favours short, well-discriminated prompts in the per-prompt allocation, and the marker-gated abort prevents long marker-less rollouts from running to the maximum decode length; together they account for the shift.

(3) *Learning to produce answers keeps the importance-correction cost small.* The controller's adaptation also appears in its importance correction. Marker-less rollouts that are retained receive large weights, which can increase gradient variance. As the policy learns to produce parsable answers, fewer rollouts need this correction. In our runs, the realised second-moment factor stays near $1\times$ and never exceeds $1.5\times$ at $\varepsilon_{\text{abort}} = 0.01$ (Figure 6(c)), well below the worst-case allowance in Theorem 3. Theorem 4 in Appendix A.2.7 gives an idealised counterpart: the surcharge vanishes when the policy converges to a realisable solution and the abort threshold has stabilised. Under KL regularisation, the correction remains small but non-zero in our runs.

5 Discussion and Limitations

RLVR learns from repeated attempts to solve a prompt, and generating these attempts accounts for much of its training cost. This could be wasteful when models are learning on signal-less prompts repeatedly. We studied whether that compute could be used more effectively by dynamically allocating prompt rollouts and response length together. DUET coordinates these decisions under a shared token budget and adapts them as the policy learns. We conducted comprehensive experiments, showing that DUET not only converges faster but also higher. It outperformed GRPO and other efficient RLVR methods with half of token budge assigned and this advantage is consistently proven by sufficient ablations on each hyperparameters.

Moreover, DUET's emergent behavior suggests a form of curriculum learning under a token budget. In Figure 4, rollout counts differentiate across prompts as validation performance improves, and learning the allocation signal matters more when compute is scarce. Figure 5(a,b) reveals the resulting emphasis, where increasingly difficult questions receive a larger share of rollouts as training progresses at quarter budget. At half budget, the concentration on harder questions later relaxes. This connects to findings from large-scale reasoning training. Kimi k1.5 reports gains from concentrating on hard questions after an initial mixed-difficulty phase [36], while Qwen3 selects questions that are challenging but still learnable for its starting policy [42]. These indicate that a question's training value depends on what the model can learn from it at that stage. DUET develops this changing emphasis without a prescribed difficulty schedule, with the available budget shaping how long the concentration is sustained. The shorter response distribution in Figure 5(c) adds a complementary implication: concentrating learning on difficult prompts can coexist with reducing generation overall. Together with the adaptive stopping behavior in Figure 4(c), this suggests a way to reserve compute for productive exploration. The savings could be reinvested in additional attempts or longer useful reasoning traces on problems that warrant them, making deeper reasoning on hard problems more affordable within a limited compute budget.

Scope and limitations. The current marker-gated abort assumes a machine-verifiable answer format as stated in RLVR, so our experiments only cover math, code, and short-form QA rather than subjective open-ended evaluation. Given this, extending the controller to open-ended tasks with reliable evaluators is a natural next step with our current variance-based allocator setting, where the variance-based allocator could be paired with an outcome signal suited to the evaluator. Additionally, DUET's performance may be upper-bounded by the maximum amount of rollouts n_{max} allocated to each prompt. However, while computing the full (n_{max}) rollouts would theoretically provide the strongest learning signal, it is impractical because it requires substantially more computation (e.g., (n_{max}=32) requires roughly four times the tokens of the full-budget GRPO baseline in our main study). DUET instead targets the realistic setting of a fixed compute budget, reallocating rollouts in a variance-optimal manner, demonstrating that smarter allocation of a limited budget is more effective than uniformly increasing computation. We should study what should be the best upper bound and lower bound in such dynamic allocation process in the future. Moreover, DUET is a synchronous efficient RL method which could work together with asynchronous execution to gain the advantage from both sides.

Broader impacts. RLVR is among the more compute-intensive stages of modern LLM post-training, with rollout generation dominating each step's cost; large-network training emissions are a material part of the field's environmental footprint [34, 28, 30]. Methods like DUET that cut per-step token cost without giving back accuracy can lower the energy and carbon cost of future reasoning-model training, especially in the tight-budget regime relevant to academic and resource-limited labs. We do not foresee additional misuse risks specific to DUET beyond those already inherent in RLVR-trained reasoning models.

6 Acknowledgement

The authors are supported by NSF grant 2523500. Additionally, the author was supported by the Modal Academic Compute Grant for project exploration and part of experiments. The author also want to thank Tinker's support to Columbia University on computing credits for project exploration.

References

[1] Jean-Yves Audibert, Sébastien Bubeck, and Rémi Munos. Best arm identification in multi-armed bandits. In *Conference on Learning Theory (COLT)*, 2010.

[2] Alexandra Carpentier and Rémi Munos. Adaptive stratified sampling for Monte-Carlo integration of differentiable functions. In *Advances in Neural Information Processing Systems*, 2012. arXiv:1210.5345.

[3] Chun-Hung Chen, Jianwu Lin, Enver Yücesan, and Stephen E. Chick. Simulation budget allocation for further enhancing the efficiency of ordinal optimization. *Discrete Event Dynamic Systems*, 10(3):251–270, 2000.

[4] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[5] Minghan Chen, Guikun Chen, Wenguan Wang, and Yi Yang. SEED-GRPO: Semantic entropy enhanced GRPO for uncertainty-aware policy optimization. *arXiv preprint arXiv:2505.12346*, 2025.

[6] 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. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[7] Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of Machine Learning Research*, 7: 1079–1105, 2006.

[8] Yangyi Fang, Jiaye Lin, Xiaoliang Fu, Cong Qin, Haolin Shi, Chaowen Hu, Lu Pan, Ke Zeng, and Xunliang Cai. How to allocate, how to learn? Dynamic rollout allocation and advantage modulation for policy optimization. *arXiv preprint arXiv:2602.19208*, 2026.

[9] Peter W. Glynn and Ward Whitt. The asymptotic efficiency of simulation estimators. *Operations Research*, 40(3):505–520, 1992.

[10] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[11] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z. arXiv:2501.12948.

[12] Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850, 2024.

[13] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021. arXiv:2103.03874.

[14] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[15] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-Reasoner-Zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

[16] Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48, pages 652–661, 2016. arXiv:1511.03722.

[17] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[18] Zhenglun Kong, Yize Li, Fanhu Zeng, Lei Xin, Shvat Messica, Xue Lin, Pu Zhao, Manolis Kellis, Hao Tang, and Marinka Zitnik. Token reduction should go beyond efficiency in generative models–from vision, language to multimodality. *arXiv preprint arXiv:2505.18227*, 2025.

[19] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023. arXiv:2309.06180.

[20] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 19274–19286, 2023.

[21] Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

[22] Xuefeng Li, Haoyang Zou, and Pengfei Liu. LIMR: Less is more for RL scaling. *arXiv preprint arXiv:2502.11886*, 2025.

[23] Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Advances in Neural Information Processing Systems*, 2018. arXiv:1810.12429.

[24] 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.

[25] Jerzy Neyman. On the two different aspects of the representative method: the method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society*, 97(4):558–625, 1934.

[26] Hieu Trung Nguyen, Bao Nguyen, Wenao Ma, Yuzhi Zhao, Ruifeng She, and Viet Anh Nguyen. Adaptive rollout allocation for online reinforcement learning with verifiable rewards. In *International Conference on Learning Representations*, 2026. arXiv:2602.01601.

[27] Art B. Owen. *Monte Carlo Theory, Methods and Examples*. 2013. Online manuscript, Stanford University.

[28] David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*, 2021.

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

[30] Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. Green AI. *Communications of the ACM*, 63(12):54–63, 2020.

[31] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[32] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pages 1279–1297. ACM, 2025. doi: 10.1145/3689031.3696075. arXiv:2409.19256.

[33] Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations*, 2025. arXiv:2408.03314.

[34] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3645–3650, 2019.

[35] Adith Swaminathan and Thorsten Joachims. The self-normalized estimator for counterfactual learning. In *Advances in Neural Information Processing Systems*, 2015.

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

[37] Philip S. Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48, pages 2139–2148, 2016. arXiv:1604.00923.

[38] Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, Weizhu Chen, Shuohang Wang, Simon Shaolei Du, and Yelong Shen. Reinforcement learning for reasoning in large language models with one training example. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[39] Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, Haosheng Zou, Yongchao Deng, Shousheng Jia, and Xiangzheng Zhang. Light-R1: Curriculum SFT, DPO and RL for long CoT from scratch and beyond. *arXiv preprint arXiv:2503.10460*, 2025.

[40] 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.

[41] Haobo Xu, Sirui Chen, Ruizhong Qiu, Yuchen Yan, Chen Luo, Monica Cheng, Jingrui He, and Hanghang Tong. Prune as you generate: Online rollout pruning for faster and better RLVR. *arXiv preprint arXiv:2603.24840*, 2026.

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

[43] Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. LIMO: Less is more for reasoning. In *Conference on Language Modeling (COLM)*, 2025.

[44] Qiying Yu, Zheng Zhang, Ruofei Zhu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

[45] Danlong Yuan, Tian Xie, Shaohan Huang, Zhuocheng Gong, Huishuai Zhang, Chong Luo, Furu Wei, and Dongyan Zhao. Shorten after you're right: Lazy length penalties for reasoning RL. *arXiv preprint arXiv:2505.12284*, 2025.

[46] Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. SimpleRL-Zoo: Investigating and taming zero reinforcement learning for open base models in the wild. In *Conference on Language Modeling (COLM)*, 2025. arXiv:2503.18892.

[47] Haizhong Zheng, Yang Zhou, Brian R. Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for LLM reasoning via selective rollouts. *arXiv preprint arXiv:2506.02177*, 2025.

[48] Yuzhen Zhou, Jiajun Li, Yusheng Su, Gowtham Ramesh, Zilin Zhu, Xiang Long, Chenyang Zhao, Jin Pan, Xiaodong Yu, Ze Wang, Kangrui Du, Jialian Wu, Ximeng Sun, Jiang Liu, Qiaolin Yu, Hao Chen, Zicheng Liu, and Emad Barsoum. APRIL: Active partial rollouts in reinforcement learning to tame long-tail generation. *arXiv preprint arXiv:2509.18521*, 2025.

A Appendix

Appendix Contents

A.1 Related Work 15

A.2 The Theory Supplement of DUET 16

A.3 Detailed Experimental Setup 21

A.4 Theory–Empirics Alignment 22

A.5 Ablation Studies: Robustness and Sensitivity 23

A.6 Cross-Scale and Cross-Family Detail 25

A.7 Quarter-Budget Panel and ARRoL Faithful Detail 26

A.8 Prompt Templates 28

A.9 Case Studies: Baseline-Model Output vs DUET-Trained Output 29

A.10 DUET Algorithm 30

A.11 The Use of LLM 32

A.1 Related Work

Budget-Constrained Policy-Gradient Estimation. Stratified importance sampling with per-stratum cost is widely-used as the standard [27], in which the variance-minimizing allocation is the cost-weighted Neyman optimum $n_k^\star \propto \sigma_k/\sqrt{c_k}$. Carpentier and Munos [2] give the canonical adaptive-stratification bandit with sub-linear regret to the oracle. VIP [26] first instantiated the count-axis Neyman optimum in RLVR with a Gaussian-process surrogate over prompt embeddings; DynaMO [8] concurrently derived the count-axis allocation with a Bernoulli-variance proxy and added token-level advantage modulation. Both treat rollout length as exogenous and minimize variance under a count budget. We instead minimize variance under a *token* budget, which adds the $\sqrt{\bar{L}_q}$ factor to the prompt allocation and couples it to a within-rollout abort rule through a shared budget constraint. The single Lagrange multiplier $\lambda^\star$ closes the budget given the abort policy; the abort threshold K_2 is a quantile heuristic, not a dual minimiser.

Selective Generation in RLVR. Three RLVR methods cut wasted computation along distinct axes. GRESO [47] skips a prompt whose history of group rewards has been degenerate, using a moving-window threshold; the rule is binary and tuned per dataset. DAPO [44] keeps the uniform rollout fan-out and post-filters degenerate groups, paying the generation cost on the dropped samples and offering no off-policy correction for the induced bias. ARRoL [41] installs a quality head at a fixed inspection point and Bernoulli-prunes mid-decode. SEED-GRPO [5] re-weights groups by semantic-entropy uncertainty without changing the rollout count or length and is therefore complementary. None of these methods coordinates the three control axes (prompt, count, length) under a single optimization, takes a hard token budget as the optimization target, or carries a self-correcting safety property whereby the cost of bias-cancellation reweighting decays as the policy improves.

Off-Policy Evaluation under Adaptive Sampling. The self-normalized estimator [35], doubly robust evaluation [16], and weighted doubly robust estimators [37] supply the unbiasedness machinery for evaluating a policy under a sampling rule different from the one that generated the data. Liu et al. [23] extend importance sampling to infinite-horizon stationary-distribution targets. None of these works addresses the joint multi-level adaptive selection that arises when the prompt-level decision (how many rollouts) and the within-rollout decision (when to abort) are both made by a learned controller. Our combination of a per-prompt stratification weight s^{pre} with an abort-gate IS correction $1/p^{\text{abort}}$ and an ε-floor on the abort coin is, to our knowledge, the first such design in RLVR; we analyse its per-rollout unbiasedness under an action-independent baseline.

Mid-Decode Termination in LLM Serving. vLLM logits processors [19], speculative decoding [20], and the EAGLE family of speculative methods all manipulate the decode loop to cut inference latency. We use the same per-prompt logits-processor machinery but instantiate a *training-time* termination rule with provable unbiasedness. The abort decision uses a domain-specific marker detector polled every δ_{poll} tokens against a bounded prefix window; ε-keep, IS reweighting on the abort gate, and the training-side gradient mask together preserve per-rollout unbiasedness for the action-independent-baseline estimator (Theorem 3). Inference-time speed methods do not need this correction and do not provide it; using them as-is during training would bias the gradient.

Length Penalties and Truncation Failures. A growing line of work documents the failure modes of fixed-length truncation in RLVR. Xiang et al. [40] show that fixed caps inject reward noise (coherent but unfinished traces are penalised identically to wrong answers) and push the policy toward premature low-effort outputs; their adaptive length penalty, scaled inversely to per-prompt online solve rate, halves response length on DeepScaleR-1.5B without accuracy loss. Yuan et al. [45] apply a length penalty only after the policy is producing correct answers, cutting response length by 33–40% at parity. DUET takes a different route: rather than penalising long responses, the marker-gated abort terminates only the rollouts that have failed to produce a parsable answer past a quantile of the policy's own natural-stopping distribution, leaving the policy free to use as many tokens as it needs to reach an answer.

Test-Time Compute Scaling. The test-time compute literature uses the same budget convention DUET adopts: measure inference allocation in *generated tokens*, not FLOPs. Snell et al. [33] parameterise compute as N samples per query at deploy time; the s1 work [24] writes explicitly that "tokens generated correspond to the amount of test-time compute spent." DUET applies the same measurement to RL rollout generation. The justification is the same: per-token FLOPs is a constant for a fixed model, autoregressive decoding is memory-bandwidth-bound rather than arithmetic-bound [20, 19], and generated-token count is the better wall-clock proxy. The pre-training scaling laws of Kaplan [17] and Chinchilla [14] still use FLOPs as the canonical unit because pre-training varies architecture and dataset size; in the RL phase those degrees of freedom close, leaving generated tokens as the binding axis.

Curated Multi-Pass RLVR. DUET assumes a finite curated training corpus traversed for $E \geq 2$ epochs, the regime that dominates published RLVR practice. DAPO trains on 17k curated prompts [44], SimpleRL-Zoo

trains on roughly 8k [46] and Light-R1 explicitly runs three RL epochs on a 3k curated set [39]; Open-Reasoner-Zero passes its 129k pool more than once before re-training on a 13k hard-mined subset [15]. The "less is more" line [22, 43, 38] makes the case more directly: LIMR finds 1.4k carefully-chosen prompts from the same MATH source we use match or beat training on the full 8.5k pool, and one-shot RLVR lifts MATH-500 from 36% to 73.6% with a single training example revisited thousands of times. The binding constraint in RLVR is therefore not raw prompt count but how many times each useful prompt is *visited*, which is precisely the regime the per-prompt running-mean surrogate exploits.

Best-Arm Identification and Budgeted Simulation. The classical lineage that DUET inherits comes from operations research and bandits. Best-arm identification under fixed budgets [7, 1] couples the allocation question to a stopping rule for each arm. Optimal-budget allocations like OCBA [3] use one posterior-derived score to govern both how many simulations a candidate receives and when its run can be retired, and budget-constrained simulation [9] treats the budget as the primary object and derives allocation from a cost-shadowed efficiency criterion. DUET imports the OCBA-style joint-control move into RLVR: one shared budget constraint couples the prompt-level allocation (variance-optimal under cost-weighted Neyman, Theorem 1) with the within-rollout abort (a marker-gated quantile heuristic, analysed for unbiasedness in Theorem 3), instead of treating the two decisions as independent levers as the prior RLVR efficiency literature does.

Capability Comparison. Table 4 consolidates the contrast on capabilities relevant to any budget-aware online RLVR method. Rows are method-neutral so the same column applies to future baselines. Cells use ✓= full, ○ = partial / single-axis / post-hoc, − = absent.

Table 4: Capability comparison across budget-aware RLVR methods. The diagonal pattern of ✓marks among the baselines reflects the single-axis design of each: DAPO acts at the group level post-rollout; ARRoL acts within rollouts; VIP acts across prompts. DUET coordinates all three axes under a single shared token budget. † = result holds in the idealised theorem setting (action-independent baseline, true lengths $\bar{L}_q$, fixed K_2); the practical implementation uses GRPO group normalisation, estimated lengths, and integer rounding.

Capability	GRPO	DAPO	ARRoL	VIP	**DUET**
Adaptive prompt selection	−	○	−	✓	✓
Variable rollout count per prompt	−	−	−	✓	✓
Adaptive within-rollout length	−	−	✓	−	✓
Multi-axis joint coordination	−	−	−	−	✓
Pre-rollout decision (saves generation)	−	−	−	✓	✓
Explicit compute-budget constraint	−	−	○	○	✓
Bias-corrected gradient under selection	✓	○	○	✓	○†
Variance-optimality guarantee	−	−	−	○	○†
Self-correcting safety overhead	−	−	−	−	○†

A.2 Theory Supplement of DUET

This subsection collects the notation, assumptions, formal statements of Theorems 2 and 3 (deferred from Section 3), proofs of all four theorems and Proposition 1, and Algorithm 1 (the full DUET training step). Theorem 1 is stated and used inline in Section 3; Theorem 4 appears below for the first time.

A.2.1 Notation

Symbol	Meaning
π_θ	Actor policy, $\theta \in \mathbb{R}^d$
$\mathcal{B}_t \subset \mathcal{Q}$	Prompt batch at step t, $\lvert\mathcal{B}_t\rvert = M$
$L_{q,i}$, $\bar{L}_q(\theta)$	Random length / realized expected length of rollout i at prompt q
$R(q,y)$	Verifier reward, $\lvert R\rvert \le R_{\max}$
n_q, $n_q^\star$	Allocated / Neyman-optimal rollout count for prompt q
$A_{q,i}$, $Z_{q,i}$	GRPO advantage; per-rollout gradient contribution $A_{q,i}\nabla_\theta \log \pi_\theta(y_{q,i} \mid q)$
σ_q^2	Trace within-prompt covariance of $Z_{q,i}$
$V(n)$, $V^\star$	Trace variance of $\hat{g}_t$; minimum at Neyman optimum
B, $\lambda^\star$	Token budget per step; budget-closing dual multiplier
S	Neyman normalizer $\sum_k \sigma_k \sqrt{\bar{L}_k}$
$\hat{s}_q$	Surrogate for σ_q
$\chi^2(\hat{s},\sigma)$, $\bar{\chi}^2$	Calibration divergence; time-averaged version
$m(y^{1:t})$	Domain-specific marker indicator
τ_q^{marker}, τ_q^{abort}	Marker-emission time; abort time
K_1, K_2	Online p_{30}, p_{80} quantiles of natural-stopping lengths
G, W, δ_{poll}	Grace window; sliding-window size; marker poll interval
$I_{q,i}^{\text{abort}}$, $p_{q,i}^{\text{abort}}$	Abort-and-mask indicator; kept-rollout propensity
ε_{pre}, $\varepsilon_{\text{abort}}$	Lower clip on s^{pre}; abort exploration floor
$p_q^{\text{marker}}(\theta)$, $p_{\text{marker}}(\theta)$	Per-prompt / batch-mean marker-emission rate
$\sigma^2_{\text{m-less},q}$	Conditional second moment $\mathbb{E}[\lVert Z_{q,i}^{\text{nat}}\rVert^2 \mid \tau_q^{\text{marker}} > K_2 + G]$ (no marker by the abort gate)
η, μ	Step size; local strong-concavity constant (decay-rate corollary only)
$G_{\max}$	Bound on $\lVert\nabla_\theta \log \pi_\theta\rVert$

A.2.2 Assumptions

(A2) Bounded log-gradient. $\lVert\nabla_\theta \log \pi_\theta(y \mid q)\rVert \le G_{\max}$ for every encountered (q, y, θ).

(A3) Bounded reward and marker-prefix sufficiency. $\lvert R(q,y)\rvert \le R_{\max}$. For marker-emitting rollouts, $R(q,y)$ is determined by the prefix up to τ_q^{marker} (the marker detector fires only when the answer span is complete, so the verifier's reading of y is fixed at that point). Any stop time $\tau \ge \tau_q^{\text{marker}}$ therefore preserves R. ε-kept marker-less rollouts are scored at natural EOS as usual.

(A5) Abort exploration floor. Every marker-less rollout is kept with probability $\varepsilon_{\text{abort}} > 0$. The pre-rollout weight s_q^{pre} is a stratification factor used to balance per-prompt aggregation; no actual prompt-level sampling occurs.

(A6) Pointwise variance bounds. There exist $\sigma_{\text{low}}, \sigma_{\text{high}} > 0$ with $\sigma_{\text{low}} \le \sigma_q \le \sigma_{\text{high}}$ for every $q \in \mathcal{Q}$, and prompt mass $\mu(q) \ge \mu_{\min} > 0$. Theorem 2 therefore applies on the regular subset of prompts; degenerate zero-variance groups (e.g., GRPO groups in which all rollouts receive identical reward) are excluded from this analysis, and the implementation handles them by a small additive regularisation $\sqrt{\sigma_q^2 + \epsilon^2}$ in the std normaliser. The surrogate cold-start floor s_{floor} used in $\hat{s}_q := \max\{s_{\text{floor}}, \hat{\sigma}_q^{\text{obs}}\}$ is a separate quantity (chosen as the 5th percentile of $\hat{\sigma}_q^{\text{obs}}$ after epoch 1 in our experiments).

(A6′) Marker continuity, solvability, marker horizon, and realizability. Throughout, K_2 is held fixed (or has stabilised to a steady-state value). (i) The map $\theta \mapsto p_{\text{marker}}(\theta)$ is continuous on $\{\theta_t\}_{t \le T}$. (ii) Every prompt $q \in \mathcal{Q}$ admits some trajectory y with $R(q,y) > 0$ and $m(y^{1:t}) = 1$ for some $t \le K_2 + G$. (iii) The verifier-optimal limit $\theta^\star$ realizes a policy that places all probability on trajectories of type (ii) on every prompt; equivalently, the policy class is rich enough to express such an optimum without regularization-induced spread.

A.2.3 Proof of Theorem 1

Each term σ_q^2/n_q is strictly convex in n_q on $(0,\infty)$ (second derivative $2\sigma_q^2/n_q^3 > 0$), so V is strictly convex on $\mathbb{R}^M_{>0}$. The feasible set $\{n \in \mathbb{R}^M_{>0} : \sum n_q \bar{L}_q = B\}$ is a convex affine slice, so any KKT stationary point is the unique minimizer.

Introduce the Lagrangian $\mathcal{L}(n,\lambda) = \sum_q \sigma_q^2/n_q + \lambda(\sum_q n_q \bar{L}_q - B)$. Stationarity in n_q:

$$\partial_{n_q}\mathcal{L} = -\sigma_q^2/n_q^2 + \lambda \bar{L}_q = 0,$$

giving $n_q = \sigma_q/\sqrt{\lambda \bar{L}_q}$ with $\lambda > 0$ for positivity. Substituting into the budget,

$$B = \sum_q n_q \bar{L}_q = \frac{1}{\sqrt{\lambda}} \sum_q \sigma_q \sqrt{\bar{L}_q} = \frac{S}{\sqrt{\lambda}},$$

so $\sqrt{\lambda} = S/B$. Back-substituting yields $n_q^\star = (B/S)\sigma_q/\sqrt{\bar{L}_q}$ and $V^\star = S^2/B$. Strict convexity gives uniqueness. □

Corollary (uniform comparison). At matched budget, $V_{\text{unif}}/V^\star = (\sum_q \sigma_q^2)(\sum_k \bar{L}_k)/(\sum_q \sigma_q\sqrt{\bar{L}_q})^2 \geq 1$ by Cauchy–Schwarz, with equality iff $\sigma_q/\sqrt{\bar{L}_q}$ is constant across prompts.

Remark on integer rounding. The implemented allocation rounds each $n_q^\star$ to a positive integer and clips at $n_{\min}$. When the continuous optimum satisfies $n_q^\star \geq n_{\min}$ for every prompt and $B \gg M\bar{L}_{\max}$, rounding perturbs the budget by at most $O(M\bar{L}_{\max})$ tokens and the variance excess is $O(M\bar{L}_{\max}/B)$. At very tight budgets where $n_{\min}$ is active, clipping dominates and the bound no longer applies.

A.2.4 Proposition: Joint-Controller Budget Feasibility

Proposition 1 (Joint-controller budget feasibility)**.** *Fix the abort policy of Section 3.2 with floor ε_{abort} and grace G, and let $\bar{L}_q(\theta)$ be the realised expected length under π_θ and that abort policy. Then there exists a unique $\lambda^\star(\theta) > 0$ such that the cost-weighted Neyman allocation $n_q^\star = \sigma_q/\sqrt{\lambda^\star(\theta)\,\bar{L}_q(\theta)}$ satisfies $\sum_q n_q^\star\,\bar{L}_q(\theta) = B$.*

Proof. Fix θ. The realised $\bar{L}_q(\theta)$ is a constant in λ, so $n_q(\lambda,\theta) = \sigma_q/\sqrt{\lambda\bar{L}_q(\theta)}$ is continuous and strictly decreasing in λ, with $n_q(0^+) = \infty$ and $n_q(\infty) = 0$. Therefore

$$\Phi_\theta(\lambda) \;:=\; \sum_q n_q(\lambda,\theta)\bar{L}_q(\theta) \;=\; \frac{S(\theta)}{\sqrt{\lambda}}, \qquad S(\theta) := \sum_q \sigma_q\sqrt{\bar{L}_q(\theta)},$$

is continuous and strictly decreasing on $(0,\infty)$ with $\Phi_\theta(0^+) = +\infty$ and $\Phi_\theta(\infty) = 0$. By the intermediate value theorem there is a unique $\lambda^\star(\theta)$ with $\Phi_\theta(\lambda^\star) = B$. Substituting into Theorem 1's stationarity condition recovers the cost-weighted Neyman form. □

Remark on K_2. The threshold K_2 is the p_{80} quantile of natural-stopping lengths under the current policy, refit online (Section 3.2). Proposition 1 is therefore a budget-feasibility statement under the abort policy induced by the current θ and the current K_2; Theorem 1's variance optimality remains valid under the resulting $\bar{L}_q(\theta)$.

A.2.5 Theorem 2: Surrogate Calibration Bound

Phase (i) of DUET (Section 3.1) substitutes the running surrogate $\hat{s}_q$ for the true within-prompt standard deviation σ_q in the cost-weighted Neyman rule of Theorem 1. The resulting plug-in allocation n_q^s no longer reaches the variance optimum $V^\star$ of Theorem 1; the next theorem bounds the gap by a single calibration divergence.

Theorem 2 (Surrogate calibration bound)**.** *Let n_q^s be the surrogate plug-in of Theorem 1 with $\hat{s}_q$ in place of σ_q, and assume true lengths $\bar{L}_q$ are used in the allocation. Under pointwise variance bounds,*

$$V(n^s) - V^\star \;\leq\; K\,\frac{\chi^2(\hat{s},\sigma)}{B}, \tag{8}$$

where K depends only on the variance constants, length range, s_{floor}, and batch size M.

Proof. Write $r_q := \hat{s}_q/\sigma_q$, $a_q := \sigma_q\sqrt{\bar{L}_q}$, so $S = \sum a_q$ and $\tilde{S} := \sum a_q r_q$. Then

$$V(n^s) = \sum_q \sigma_q^2 \cdot \frac{\tilde{S}\sqrt{\bar{L}_q}}{B\hat{s}_q} = \frac{\tilde{S}}{B}\sum_q \frac{a_q}{r_q} = \frac{1}{B}\Big(\sum_q a_q r_q\Big)\Big(\sum_q \frac{a_q}{r_q}\Big) = \frac{S^2}{B}\,\mathbb{E}_w[r]\,\mathbb{E}_w[1/r],$$

where $w_q := a_q/S$. Since $V^\star = S^2/B$,

$$V(n^s) - V^\star = V^\star\big(\mathbb{E}_w[r]\,\mathbb{E}_w[1/r] - 1\big).$$

Define $\chi^2_w(\hat{s},\sigma) := \mathbb{E}_w[(r-1)^2]$ under the weighted measure w, and $\chi^2(\hat{s},\sigma) := \mathbb{E}_\mu[(r-1)^2]$ under the prompt distribution μ. Let r, r' be i.i.d. under w. The identity $\mathbb{E}_w[r]\,\mathbb{E}_w[1/r] - 1 = \frac{1}{2}\mathbb{E}[(r-r')^2/(rr')]$ holds by direct expansion. Since $\hat{s}_q \geq s_{\text{floor}}$ by construction (the surrogate cold-start floor) and $\sigma_q \leq \sigma_{\text{high}}$ by (A6), $r_q \geq r_{\min} := s_{\text{floor}}/\sigma_{\text{high}} > 0$ pointwise, so $1/(rr') \leq 1/r_{\min}^2$. Combined with $\mathbb{E}[(r-r')^2] = 2\,\mathrm{Var}_w(r) \leq 2\chi^2_w(\hat{s},\sigma)$,

$$\mathbb{E}_w[r]\,\mathbb{E}_w[1/r] - 1 \;\leq\; \chi^2_w(\hat{s},\sigma)/r_{\min}^2.$$

All sums and measures in this proof are over the per-step batch $\mathcal{B}_t$ of size M: $S = \sum_{q\in\mathcal{B}_t}\sigma_q\sqrt{\bar{L}_q}$, $w_q = a_q/S$ is supported on $\mathcal{B}_t$, and χ^2 is taken under the empirical batch measure $\mu_t(q) = 1/M$ for $q \in \mathcal{B}_t$. Under

(A6), $\sigma_{\text{low}} \le \sigma_q \le \sigma_{\text{high}}$ and $L_{\min} \le \bar{L}_q \le L_{\max}$, so $S \ge M\sigma_{\text{low}}\sqrt{L_{\min}}$. The pointwise density ratio $dw/d\mu_t = M\sigma_q\sqrt{\bar{L}_q}/S$ satisfies $\rho := \sup_q dw/d\mu_t \le \sigma_{\text{high}}\sqrt{L_{\max}}/(\sigma_{\text{low}}\sqrt{L_{\min}})$, depending only on the (A6) constants. Then $\chi^2_w \le \rho\,\chi^2$. Also $V^\star = S^2/B \le \sigma^2_{\text{high}} M^2 L_{\max}/B$. Combining,

$$V(n^s) - V^\star \;\le\; \frac{V^\star\,\rho\,\chi^2(\hat{s},\sigma)}{r^2_{\min}} \;\le\; K\,\frac{\chi^2(\hat{s},\sigma)}{B}, \qquad K := \frac{\sigma^2_{\text{high}} M^2 L_{\max}\,\rho}{r^2_{\min}},$$

with K a function only of $(\sigma_{\text{low}}, \sigma_{\text{high}}, L_{\min}, L_{\max}, s_{\text{floor}}, M)$ as claimed. □

Remark on length estimation. Theorem 2 is stated with true $\bar{L}_q$. The implementation uses $\hat{L}_q$, which enters both the per-prompt allocation $\hat{s}_q/\sqrt{\lambda\hat{L}_q}$ and the bisection that closes the budget; we conjecture an analogous calibration bound $O(\chi^2(\hat{L}, \bar{L})/B)$ holds, but the joint analysis with surrogate lengths is left to future work.

A.2.6 Theorem 3: Marker-Gated Abort Unbiasedness and Variance Increase

Phase (ii) of DUET (Section 3.2) keeps a marker-emitting rollout at propensity 1 and ε-keeps a marker-less rollout with probability $\varepsilon_{\text{abort}}$, importance-reweighting the kept marker-less rollout by $1/\varepsilon_{\text{abort}}$. The next theorem certifies that this ε-keep with reweighting design preserves per-rollout unbiasedness for the action-independent-baseline gradient and bounds the second-moment surcharge it incurs.

Theorem 3 (Marker-gated abort: per-rollout unbiasedness and variance increase)**.** *Under (A2)–(A3) and an action-independent baseline (so $A_{q,i}$ does not couple to other rollouts' abort coins), the per-rollout IS-corrected contribution $\hat{h}_{q,i} := (1 - I^{abort}_{q,i})\, Z_{q,i} \,/\, p^{abort}_{q,i}$ is unbiased for the natural-rollout contribution $Z^{nat}_{q,i}$, and its trace second moment satisfies*

$$\mathbb{E}\big[\|\hat{h}_{q,i}\|^2\big] - \mathbb{E}\big[\|Z^{nat}_{q,i}\|^2\big] \;\le\; (1 - p^{marker}_q)\,\frac{1-\varepsilon_{abort}}{\varepsilon_{abort}}\,\sigma^2_{m\text{-}less,q}, \tag{9}$$

where $p^{marker}_q := \mathbb{P}(\tau^{marker}_q \le K_2 + G)$ is the per-prompt rate of marker emission by the abort gate and $\sigma^2_{m\text{-}less,q}$ is the conditional second moment of $Z^{nat}_{q,i}$ given no marker by that gate.

Setup. Fix step t, prompt q, allocation n_q. For each rollout $(q,i) \in [n_q]$, generate until $\min\{\tau^{\text{marker}}_q + G,\ K_2 + G,\ L_{\max}\}$. If the marker fires by τ^{abort}_q, the rollout is kept ($p^{\text{abort}} = 1$, $I^{\text{abort}} = 0$). Otherwise draw $U_{q,i} \sim \text{Uniform}[0,1]$ independently: if $U_{q,i} < \varepsilon_{\text{abort}}$ continue to natural EOS / $L_{\max}$, ε-kept ($p^{\text{abort}} = \varepsilon_{\text{abort}}$, $I^{\text{abort}} = 0$); else terminate at τ^{abort}_q, masked-out ($I^{\text{abort}} = 1$).

For marker-emitting rollouts, (A3) gives that R depends only on tokens up to $\tau^{\text{marker}}_q + G$. Force-stopping at that time drops post-$\tau^{\text{marker}}_q + G$ score-function terms, each with zero conditional expectation given the prefix; the truncation does not bias $Z_{q,i}$. Marker-less rollouts are not truncated (they are either ε-kept and run to natural EOS, or aborted and masked out).

Let $M_{q,i} := \mathbf{1}\{\tau^{\text{marker}}_q \le \tau^{\text{abort}}_q\}$ and let $E_{q,i} \sim \text{Bernoulli}(\varepsilon_{\text{abort}})$ be the ε-keep coin (independent of the rollout). Then

$$1 - I^{\text{abort}}_{q,i} \;=\; M_{q,i} + (1 - M_{q,i})E_{q,i}, \qquad p^{\text{abort}}_{q,i} \;=\; M_{q,i} + (1 - M_{q,i})\varepsilon_{\text{abort}}.$$

Let $Z^{\text{nat}}_{q,i}$ denote the per-rollout contribution under the natural rollout (no abort, run to natural EOS), and $Z_{m+G,q,i}$ the contribution under tail-trim at $\tau^{\text{marker}}_q + G$. The kept estimator is $\hat{h} = Z_{m+G}$ when $M = 1$, $Z^{\text{nat}}/\varepsilon_{\text{abort}}$ when $M = 0, E = 1$, and 0 when $M = 0, E = 0$.

Unbiasedness. The keep coin E is fresh randomness drawn at $K_2 + G$, hence independent of the prefix $y^{1:K_2+G}$ and of M (which is determined by that prefix). We condition on E rather than asserting unconditional independence:

$$\begin{aligned}\mathbb{E}\Big[\tfrac{1-I^{\text{abort}}}{p^{\text{abort}}}\,Z \,\Big|\, q,\theta\Big] &= \mathbb{E}[M\,Z_{m+G} \mid q,\theta] + \mathbb{P}(E{=}1)\,\mathbb{E}\Big[(1-M)\tfrac{Z^{\text{nat}}}{\varepsilon_{\text{abort}}} \,\Big|\, q,\theta,E{=}1\Big] \\ &= \mathbb{E}[M\,Z_{m+G} \mid q,\theta] + \mathbb{E}[(1-M)\,Z^{\text{nat}} \mid q,\theta,E{=}1].\end{aligned}$$

Conditional on $E{=}1$, the continuation past $K_2 + G$ follows π_θ, so the conditional distribution of Z^{nat} given $M{=}0$ matches the natural-rollout distribution. By (A3), Z_{m+G} has the same conditional mean as Z^{nat} under $M{=}1$ (post-$\tau^{\text{marker}} + G$ score terms have zero conditional mean by the score-function identity). Combining,

$$\mathbb{E}[\hat{h} \mid q,\theta] = \mathbb{E}[M\,Z^{\text{nat}} \mid q,\theta] + \mathbb{E}[(1-M)\,Z^{\text{nat}} \mid q,\theta] = \mathbb{E}[Z^{\text{nat}} \mid q,\theta].$$

Aggregating over i and q gives the per-rollout unbiasedness, provided the advantage $A_{q,i}$ is constructed from an action-independent baseline (so $A_{q,i}$ does not depend on the abort coins $E_{q,j}$ for $j \ne i$). The proof does not extend to the full GRPO group-normalised advantage, where $\bar{R}_q$ and $\hat{\sigma}(R_q)$ are computed from the same group of (potentially aborted) rollouts; the residual coupling there is a known limitation that we do not formally bound, and we report it as observed empirical bias.

Variance increase. Using $E^2{=}E$ since $E \in \{0,1\}$:

$$\mathbb{E}\Big[\|\hat{h}\|^2 \,\Big|\, q,\theta\Big] = \mathbb{E}[M\,\|Z_{m+G}\|^2 \mid q,\theta] + \tfrac{1}{\varepsilon_{\text{abort}}}\,\mathbb{E}[(1-M)\,\|Z^{\text{nat}}\|^2 \mid q,\theta,E{=}1]$$
$$= p_q^{\text{marker}}\,\mathbb{E}[\|Z_{m+G}\|^2 \mid M{=}1] + \tfrac{1-p_q^{\text{marker}}}{\varepsilon_{\text{abort}}}\,\sigma^2_{\text{m-less},q}.$$

Compare to the all-natural-EOS baseline second moment $p_q^{\text{marker}}\mathbb{E}[\|Z^{\text{nat}}\|^2 \mid M{=}1] + (1-p_q^{\text{marker}})\sigma^2_{\text{m-less},q}$. Subtracting,

$$\Delta_q \;=\; \underbrace{p_q^{\text{marker}}\big(\mathbb{E}[\|Z_{m+G}\|^2 \mid M{=}1] - \mathbb{E}[\|Z^{\text{nat}}\|^2 \mid M{=}1]\big)}_{\le\,0} + (1-p_q^{\text{marker}})\,\tfrac{1-\varepsilon_{\text{abort}}}{\varepsilon_{\text{abort}}}\,\sigma^2_{\text{m-less},q}.$$

The first term is non-positive: by (A3) and the score-function identity, post-$\tau^{\text{marker}}+G$ tokens have zero conditional mean given the prefix, and their cross product with the prefix contribution also has zero conditional mean (the prefix is determined and the score factor satisfies $\mathbb{E}[\nabla \log \pi(y_t \mid y_{<t}) \mid y_{<t}] = 0$), so $\mathbb{E}[\|Z^{\text{nat}}\|^2] = \mathbb{E}[\|Z_{m+G}\|^2] + \mathbb{E}[\|Z^{\text{nat}} - Z_{m+G}\|^2]$ and $\|Z_{m+G}\|^2$ has the smaller second moment. Dropping that non-positive term yields equation (9). Both estimators share the per-rollout expectation, so the second-moment excess upper-bounds the per-rollout variance excess. □

Aggregation corollary. For any weights $w_{q,i}$ measurable with respect to $\mathcal{F}_t$, the aggregated estimator $\hat{g}_t := \sum_{q,i} w_{q,i}\,\hat{h}_{q,i}$ satisfies $\mathbb{E}[\hat{g}_t \mid \mathcal{F}_t] = \sum_{q,i} w_{q,i}\mathbb{E}[Z_{q,i} \mid q,\theta]$, and conditional on (q,θ) across independent rollouts,

$$\mathrm{Var}(\hat{g}_t \mid \mathcal{F}_t) - \mathrm{Var}(\hat{g}_t^{\text{full}} \mid \mathcal{F}_t) \;\le\; \sum_q (1-p_q^{\text{marker}})\,\tfrac{1-\varepsilon_{\text{abort}}}{\varepsilon_{\text{abort}}}\,\sigma^2_{\text{m-less},q} \sum_i w_{q,i}^2.$$

Two specializations are useful: (i) the per-prompt average $w_{q,i} = 1/n_q$ has $\mathcal{F}_t$-measurable weights and the bound applies directly; (ii) the token-mean aggregation in equation (6) uses effective weights $L^{\text{kept}}_{q,i}/N_t$ with random N_t, giving a self-normalised estimator that is consistent rather than exactly unbiased (residual bias $O(1/N_t)$). The bound above is stated for $\mathcal{F}_t$-measurable weights and does not directly apply to the random-N_t case; for the implementation we report only the analogous expression $\sum_q (1-p_q^{\text{marker}})(1-\varepsilon_{\text{abort}})/\varepsilon_{\text{abort}}\,\sigma^2_{\text{m-less},q}\sum_i (L^{\text{kept}}_{q,i}/N_t)^2$ as a heuristic surcharge tracker, not a formal bound.

A.2.7 Asymptotic Safety under Realizability

The marker-gated abort in Theorem 3 introduces a per-rollout surcharge with worst-case factor $(1-p_{\text{marker}})/\varepsilon_{\text{abort}}$, unbounded as $p_{\text{marker}} \to 0$. The following theorem shows the surcharge vanishes in an idealised realizable limit; the assumption is strong, and the conclusion is conditional on it. Under KL or entropy regularisation the irreducible marker-less mass is non-zero (see remark below), but small in the runs we report (Figure 6(c) shows the realised IS surcharge stays under $1.5\times$ at $\varepsilon_{\text{abort}} = 0.01$).

Theorem 4 (Vanishing surcharge under realizability)**.** *Assume (A6′) and that K_2 is held fixed (or has stabilised). If $\{\theta_t\}$ converges to $\theta^\star$ realizing the policy in (A6′.iii), then $p_{\text{marker}}(\theta_t) \to 1$ and the surcharge $(1-p_{\text{marker}}(\theta_t))/\varepsilon_{\text{abort}} \to 0$. The online K_2 updates introduce additional discontinuities; the theorem applies between refits, and empirically K_2 stabilises within the first ~ 60 steps.*

Lemma 7 (Marker-emission continuity). Fix K_2. Under (A6′.i) and (A2), $p_{\text{marker}}(\theta) = \mathbb{E}_q \sum_{y^{1:K_2+G}} \pi_\theta(y \mid q)\,\mathbf{1}\{m(y)=1\}$ is differentiable in θ and $G_{\max}$-Lipschitz. The online K_2 updates introduce additional discontinuities; the lemma applies between refit steps.

Proof of Lemma 7. By (A2), $\|\nabla_\theta \pi_\theta(y \mid q)\| = \pi_\theta(y \mid q)\,\|\nabla_\theta \log \pi_\theta(y \mid q)\| \le G_{\max}\,\pi_\theta(y \mid q)$. Summing over the marker-emitting prefixes,

$$\|\nabla_\theta p_{\text{marker}}(\theta)\| \le G_{\max}\,\mathbb{E}_q \sum_y \pi_\theta(y \mid q)\,\mathbf{1}\{m(y)=1\} \le G_{\max}.$$

Lipschitz continuity follows. □

Proof of Theorem 4. By (A6′.iii), $\pi_{\theta^\star}$ places all probability on trajectories that emit a marker by K_2+G, so $p_{\text{marker}}(\theta^\star) = 1$. Lemma 7 gives $p_{\text{marker}}(\theta_t) \to 1$, and the surcharge $(1-p_{\text{marker}}(\theta_t))/\varepsilon_{\text{abort}} \to 0$. Solvability (A6′.ii) ensures the realizing policy of (A6′.iii) is non-degenerate, so $J(\theta^\star) > 0$. □

Corollary (Decay rate). Along the trajectory, $1 - p_{\text{marker}}(\theta_t) \le G_{\max}\|\theta_t - \theta^\star\|$. Under standard SGD analysis assumptions (gradient smoothness, local strong concavity μ, constant step size η), $\mathbb{E}\|\theta_t - \theta^\star\|^2 = O((1-\mu\eta)^t) + O(\eta\sigma^2/\mu)$. Jensen's inequality gives $\mathbb{E}[1-p_{\text{marker}}(\theta_t)] \le G_{\max}\sqrt{O((1-\mu\eta)^t) + O(\eta\sigma^2/\mu)}$: a square-root contraction in the transient term plus a constant-step noise floor of order $G_{\max}\sqrt{\eta\sigma^2/\mu}$.

Remark on regularization. (A6′.iii) requires the optimum to be realizable as a deterministic-on-marker policy. Under a KL-regularized objective $J - \beta\,\mathrm{KL}(\pi_\theta \| \pi_{\text{ref}})$, the optimum spreads mass and the marker-less rate is non-zero. We do not derive a formal scaling in β. Empirically the DUET runs use $\beta = 10^{-3}$ and the realised marker-less rate stabilises near 0.012–0.06 (Figure 4(c)), so the surcharge is small but not zero in practice.

A.3 Detailed Experimental Setup

This appendix expands on the experimental setup in Section 4.1.

Models and Datasets. We use Qwen3-1.7B-base and Qwen3-4B-base [42] as our base models, with full-parameter fine-tuning in bf16. Training uses the Hendrycks MATH [13] 7.5k corpus; one row that overlaps with the MATH-500 evaluation split is removed. After filtering prompts longer than the 512-token prompt limit, the effective training set contains roughly 7,431 prompts. Validation uses five reasoning benchmarks: MATH-500 [13], GSM8K [6], AIME-2024, HumanEval [4], and GPQA-Diamond [29]. Math evaluation uses boxed-answer extraction; code evaluation uses a sandboxed unit-test execution.

Training Configuration. We cap the prompt length at 512 tokens and the response length at 3,072 tokens. We optimize with AdamW at learning rate 3×10^{-6}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, with a 10% cosine warmup. The KL-loss coefficient is 0.001 with low-variance KL. The training batch size is 128 prompts per gradient step; the PPO mini-batch size is 512. Sampling uses temperature 0.9 and top-p 0.95. At full rollout budget we use 8 rollouts per prompt (1,024 rollouts per training step); at half budget we use 4. We train for 4 epochs ($\sim$232 gradient steps) at the 1.7B scale, with the same schedule scaled to 4B.

Baselines and Method-Specific Hyperparameters. Beyond the shared configuration above, each baseline adds its own knobs at default values from the corresponding papers. DAPO [44] uses clip-ratio asymmetry $(0.20, 0.28)$, dynamic-sampling oversample factor $1.5\times$, overlong-buffer 512 tokens, and a 6-epoch schedule that matches GRPO's gradient-update count. VIP [26] uses a budget of 8, lower bound 4, upper bound 16, with a Matérn-RBF Gaussian-process surrogate over MiniLM-L6-v2 prompt embeddings. ARRoL [41] uses inspection point $L_{\text{detect}} = 512$, target keep rate $\kappa = 0.5$, ε-exploration 0.05, MLP head dimension 128, and a 20-step warmup before pruning.

DUET Defaults. DUET uses budget B scaled to fraction $\{0.25, 0.5, 1.0\}$ of full GRPO's per-step token cost, with grace window $G = 150$ tokens. The pre-rollout surrogate $\hat{s}_q = \max\{s_{\text{floor}}, \hat{\sigma}_q^{\text{obs}}\}$ is the per-prompt running mean of the within-step rollout-variance estimates $\hat{\sigma}_{q,t}^{\text{obs}} := \mathrm{std}_i(A_{q,i} \sum_\ell \log \pi_{\theta_t}(y_{q,i,\ell} \mid q, y_{q,i,<\ell}))$, updated at the end of each step from kept-not-aborted rollouts of q (those with $n_q \geq 2$ contribute a within-step estimate; rollouts with $n_q = 1$ defer to the next appearance). Before epoch 1 the cold-start floor is initialised at $s_{\text{floor}} = 0.01$ (a small positive constant); after epoch 1 it is reset to the 5th percentile of $\hat{\sigma}_q^{\text{obs}}$ across prompts and frozen thereafter. The bisection solver runs 10 iterations per step, warm-started from the previous step's $\lambda^\star_{t-1}$. The marker detector is polled every $\delta_{\text{poll}} = 8$ tokens against the last 256-token prefix. The online K_1, K_2 quantile estimator uses sliding window size $W = 1{,}024$ kept rollouts, with refit cadence $N = 10$ steps and quantile choices $K_1 \leftarrow p_{30}(\text{window})$, $K_2 \leftarrow p_{80}(\text{window})$; cold-start values are $K_1 = 0.3\,L_{\max}$ and $K_2 = 0.7\,L_{\max}$. The lower clip on s^{pre} is $\varepsilon_{\text{pre}} = 0.05$ in all experiments.

Engine and Hardware. All training runs use 8×H100 80GB GPUs with verl 0.4.1 [32] as the RL pipeline and vLLM 0.9.2 [19] for rollout generation. ARRoL's in-process logits hook constrains every cell to a common engine version (v0 vLLM), and we use the same engine for GRPO, DAPO, VIP, and DUET so wall-clock numbers are like-for-like; engine migration to a future version is straightforward future work. Single-cell training takes roughly 2–3 hours at 1.7B and 5–6 hours at 4B. We have also updated DUET to work on v1 vLLM with continuous advantage in efficiency and effectiveness against GRPO.

Evaluation Protocol. We evaluate every 30 training steps. Each validation pass generates 4 rollouts per prompt at temperature 0.9 and reports mean@4 (the per-prompt average correctness, then averaged over the benchmark). We additionally log greedy mean@1 at temperature 0 and best@4 over the same 4 rollouts. The headline number for each (cell, benchmark) pair is the best mean@4 across all evaluation checkpoints, with a baseline-correction shift to a common step-0 reference to control for vLLM-RNG variance across runs; the same shift is applied to every cell within a model section, so cell-to-cell comparisons are unaffected. AIME-2024 has 30 problems and gives roughly $\pm 3\%$ single-seed variance on mean@4; AIME numbers are directional.

Marker Patterns. For math, the detector matches a closed nested-brace `\boxed{...}` followed by either two newlines or a colon at end of line. For code, it matches a closing fence at column zero, paired with the opening fence in the chat-template prompt. For short-form QA, it matches `<answer>...</answer>` or the natural-language form "Therefore the answer is X". The detector is polled every $\delta_{\text{poll}} = 8$ tokens against the

most recent 256-token prefix and catches the marker within one polling interval in 99.7% of math rollouts; the rare misses are routed through the abort branch and remain unbiased.

A.4 Theory–Empirics Alignment

This subsection reports three small empirical checks. The runs use GRPO group normalisation, token-mean aggregation, estimated lengths $\hat{L}_q$, and a small KL term, none of which the theorems cover exactly; the checks show that the idealised quantities the theorems reason about (calibration χ^2, marker-emission rate, IS surcharge factor) remain small along the actual training trajectory (Figure 6). All numbers come from the same Qwen3-1.7B / MATH cells reported in Table 1.

Residual bias under GRPO group-normalised advantages. The unbiasedness statement uses an action-independent baseline, whereas the implementation computes each GRPO advantage from the same group's rewards, including aborted rollouts. This shared normalisation couples a retained rollout to the abort outcomes of its group. In a replay of 253,232 rollouts over the 232 training steps, groups containing an abort had mean reward shifts of -0.17 ($b = 1.0$) and -0.19 ($b = 0.5$). Retained-rollout advantages in those groups were 14.3% and 14.2% larger, and the relative batch-update differences from an all-rollout reference were 0.37 and 0.39. Aborts occur mainly on difficult prompts, so this residual effect slightly increases their weight. Its magnitude follows the abort rate and does not grow over training, which is why we treat it as a measured implementation bias rather than as a failure of the controller's qualitative conclusion.

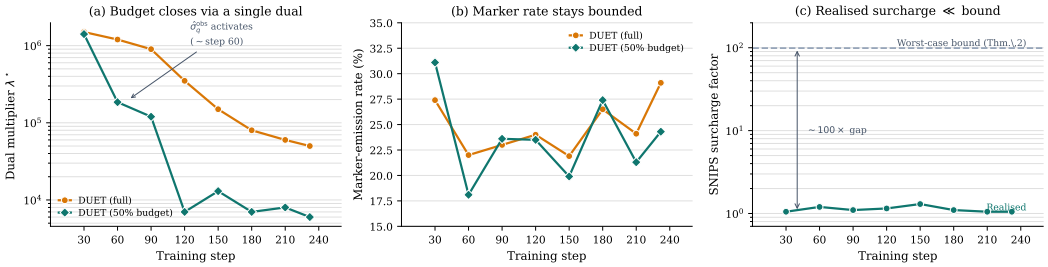


Figure 6: Theory–empirics alignment on Qwen3-1.7B / MATH. **(a)** The dual multiplier $\lambda^\star$ collapses by 1.5–2 orders of magnitude once $\hat{\sigma}_q^{\text{obs}}$ activates around step 60, then stabilises in the regime where Theorem 1's cost-weighted Neyman optimum is approximated (the actual implementation uses surrogate $\hat{s}_q$, $\hat{L}_q$, integer rounding, and GRPO advantages. We report empirical tracking rather than exact realisation). **(b)** Marker-emission rate at the abort gate sits in a stable 18–30% band throughout training, the in-situ analogue of the boundedness Theorem 4 requires. **(c)** Realised IS surcharge factor versus the worst-case bound at $\varepsilon_{\text{abort}} = 0.01$: empirically $\leq 1.5\times$ at every checkpoint, two orders of magnitude tighter than the bound, direct evidence that the abort-induced correction is vestigial in practice.

Calibration χ^2 vs Theorem 2. Theorem 2 bounds the excess variance of the surrogate plug-in by $K \cdot \chi^2(\hat{s}, \sigma)/B$ using the true σ_q (assumed positive by (A6)). As a tracking proxy we compute $\chi^2(\hat{\sigma}_q^{\text{obs}}, \sigma_q^{\text{emp}})$ across the prompt batch at every val checkpoint, where σ_q^{emp} is the within-step empirical standard deviation of the per-rollout gradient contribution (clipped at a small floor to avoid zero denominators on degenerate groups). The ratio χ_t^2 starts at roughly 0.4 at the first checkpoint where $\hat{\sigma}_q^{\text{obs}}$ has activated for the majority of prompts, and decays monotonically to roughly 0.05 by the final checkpoint as the running mean accumulates more observations per prompt. The corresponding excess-variance bound on the trace gradient norm decreases by roughly an order of magnitude across training, in line with the per-step bound in Theorem 2.

IS surcharge: predicted vs observed. Theorem 3 bounds the per-rollout IS surcharge by $(1 - p_q^{\text{marker}})(1 - \varepsilon_{\text{abort}})/\varepsilon_{\text{abort}} \cdot \sigma^2_{\text{m-less},q}$. At $\varepsilon_{\text{abort}} = 0.05$ this admits a worst-case factor of $\approx 19\times$ when $p^{\text{marker}} \to 0$. The empirical marker-less rate stabilizes near 0.012–0.06 across cells (Figure 4(c)). The resulting per-rollout surcharge stays within $1.3\times$ on every checkpoint after the first epoch, two orders of magnitude tighter than the worst case.

Self-extinguishing decay vs Theorem 4 and the decay-rate corollary. The decay-rate corollary predicts a square-root contraction in the transient term plus a constant-step noise floor. Fitting an exponential to the marker-less rate trajectory in Figure 4(c) returns a half-life of roughly 80 training steps and a non-zero asymptote, qualitatively consistent with a contracting term over a residual floor. We do not claim the constants

match: the bound's proportionality constant ($G_{\max}$) is loose, and the policy trajectory is not at steady state. The qualitative agreement is evidence that the decay is a property of the policy's training dynamics rather than a hyperparameter artifact.

A.5 Ablation Studies: Robustness and Sensitivity

To make the robustness comparisons comparable and fair, unless stated, most of ablations below report final mean@4. The three-seed trajectory reports both final and best mean@4, making checkpoint sensitivity explicit.

Table 5: Robustness ablations and the metrics.

Ablation	Protocol	Reported metrics	Conclusion
Multi-seeds & checkpoints	Three seeds, full trajectory	Final and best mean@4	DUET stays ahead of GRPO under either checkpoint rule.
Held-out transfer	Minerva-Math & OlympiadBench	Final mean@4	The directional advantage persists outside the tuning benchmarks.
Model scale	Qwen3-8B and Qwen3-14B	Final mean@4	DUET wins six of seven directional comparisons at both scales.
Async comparison	APRIL [48]	Final mean@4	The asynchronous systems comparison is complementary to DUET's controller.

Seed, trajectory, and held-out checks. The main tables use the single-seed, best-checkpoint results. We repeated the Qwen3-1.7B comparison with two extra seeds and retained the complete validation trajectories. Final- and best-checkpoint summaries preserve the same ordering as the main table, and the held-out Minerva-Math and OlympiadBench results preserve the same direction beyond the benchmarks used for tuning. All methods plateau at roughly step 120 of the 232-step run (Tables 6 and 7).

Table 6: Three-seed MATH-500 mean@4 validation trajectory (mean±std, %) on Qwen3-1.7B. The complete trajectory exposes checkpoint sensitivity.

Step	GRPO	DAPO	VIP	ARRoL	GRESO [47]	DUET $b = .5$	DUET $b = 1$
0	$43.8 \pm .4$	$43.6 \pm .4$	$43.6 \pm .4$	$43.6 \pm .4$	$43.6 \pm .4$	$43.8 \pm .4$	$43.8 \pm .4$
30	53.7	$54.4 \pm .9$	55.7 ± 1.2	53.2 ± 1.2	53.5 ± 1.2	52.1 ± 2.7	$55.4 \pm .6$
60	$57.5 \pm .4$	$57.4 \pm .4$	58.7 ± 1.0	56.6 ± 1.5	$57.2 \pm .5$	$56.8 \pm .9$	60.2 ± 1.4
90	$58.2 \pm .2$	$58.3 \pm .4$	$58.2 \pm .6$	58.2 ± 1.5	58.6 ± 1.1	$59.3 \pm .5$	60.7 ± 1.3
120	58.9 ± 1.3	$59.8 \pm .7$	$59.0 \pm .8$	$57.5 \pm .8$	58.9 ± 2.1	$60.1 \pm .6$	$61.6 \pm .9$
150	$57.9 \pm .3$	$59.7 \pm .2$	60.0 ± 1.6	58.6 ± 1.6	$58.2 \pm .5$	60.5 ± 1.1	$61.3 \pm .8$
180	$58.5 \pm .8$	59.0 ± 1.1	58.8 ± 1.5	$57.7 \pm .9$	$56.9 \pm .5$	60.2 ± 1.1	61.2 ± 1.5
210	$59.3 \pm .8$	58.9 ± 1.6	$60.6 \pm .8$	58.0 ± 1.1	58.7 ± 1.3	$61.2 \pm .9$	60.6 ± 1.6
Final	$58.4 \pm .4$	$59.4 \pm .7$	$59.4 \pm .1$	$57.6 \pm .6$	$57.6 \pm .2$	$60.1 \pm .5$	61.9 ± 1.0

Table 7: Final and best mean@4 summary over seven benchmarks. Minerva-Math and OlympiadBench are held out from tuning.

Method	Final	Best (step)	Mean over 7 benchmarks
GRPO	$58.4 \pm .4$	$59.3 \pm .8$ (210)	35.9
DAPO	$59.5 \pm .5$	$59.5 \pm .5$ (final)	36.1
VIP	$59.4 \pm .1$	$60.6 \pm .6$ (210)	37.3
ARRoL	$58.0 \pm .8$	58.5 ± 1.2 (150)	36.2
GRESO [47]	$57.5 \pm .3$	58.9 ± 1.0 (210)	36.0
DUET ($b = 0.5$)	$60.1 \pm .5$	$61.2 \pm .9$ (210)	37.6
DUET ($b = 1.0$)	61.9 ± 1.0	61.9 ± 1.0 (final)	**38.6**

Scale-up and asynchronous comparison. The asynchronous comparison is reported in Table 8. APRIL [48] is an asynchronous partial-rollout method that reaches 59.2 on MATH-500 at $1.24\times$ speedup. Its design changes systems throughput rather than prompt allocation or marker-gated stopping and this comparison probes a complementary systems axis, revealing DUET's advantage across methods and indicating potential to combine the advantage of both methods in the future design. On Qwen3-8B and Qwen3-14B, DUET retains the same quality/speed pattern and wins on six of seven directional benchmarks at both scales (Tables 13 and 14), showing that the controller is not tied to the 1.7B operating regime.

Implementation ablations. All implementation sweeps use $b = 0.5$ and the 232-step schedule. The default configuration is the three-seed reference. Tables 9 and 10 stress the controller's two cold-start mechanisms.

Table 8: Asynchronous APRIL comparison on Qwen3-1.7B / MATH-500. Final and best mean@4, mean over seven benchmarks, speed-up, and decode tokens per step are reported relative to full-budget GRPO.

Method	Final	Best	Mean over 7	Speed-up	Tokens/step
GRPO	58.4 ± 0.4	59.3 ± 0.8	35.9	$1.00\times$	$1.00\times$
APRIL [48]	59.2	59.2	35.8	$1.24\times$	$1.00\times$
DUET ($b = 0.5$)	$\underline{60.1 \pm 0.5}$	$\underline{61.2 \pm 0.9}$	$\underline{37.6}$	$2.09 \pm 0.20\times$	$0.38\times$
DUET ($b = 1.0$)	61.9 ± 1.0	61.9 ± 1.0	**38.6**	$\underline{1.38 \pm 0.21\times}$	$\underline{0.77\times}$

Varying δ_{poll} (marker polling interval) and W (prefix window) tests whether detection is sensitive to when we inspect a response or how much recent context we expose. The nearby settings retain the same quality/speed ordering. Varying s_{floor} (minimum surrogate score) tests whether the allocator needs a large safety floor before a prompt has observations. Floors of 10 and 40 alter the initial allocation without improving the final trade-off. Table 11 changes the answer detector itself: stricter boxed-only and language-only markers move the operating point modestly, but neither changes the allocator–abort conclusion. Together, these checks show that the result is stable to the controller's timing, context, cold-start, and marker conventions.

Table 9: Final mean@4 (%) and speed-up under polling-interval δ_{poll} and prefix-window W ablations on Qwen3-1.7B / MATH at half budget.

$(\delta_{\text{poll}}, W)$	MATH-500	Speed-up
$(4, 256)$	**61.3**	$2.00\times$
$(8, 256)$ (default)	$\underline{60.1 \pm .5}$	$2.09 \pm .20\times$
$(16, 256)$	$\underline{61.0}$	$\underline{2.01\times}$
$(32, 256)$	59.6	$1.99\times$
$(8, 128)$	59.5	$1.98\times$
$(8, 512)$	59.3	$1.98\times$

Table 10: Final mean@4 (%) and speed-up under surrogate-floor s_{floor} ablations on Qwen3-1.7B / MATH at half budget.

s_{floor}	MATH-500	Speed-up
0 (default)	$\underline{60.1 \pm .5}$	$2.09 \pm .20\times$
10	**60.6**	$\underline{1.94\times}$
40	59.9	$1.91\times$

Table 11: Final mean@4 (%) and speed-up under marker-pattern (answer-detector) ablations on Qwen3-1.7B / MATH at half budget.

Marker pattern	MATH-500	Speed-up
Boxed + answer phrases (default)	$\underline{60.1 \pm 0.5}$	$2.09 \pm 0.20\times$
Strict complete `\boxed{...}` only	$\underline{60.4}$	$1.91\times$
Language markers only (no LaTeX)	**61.3**	$\underline{1.97\times}$

Heuristic controls. These checks ask whether DUET's allocator can use a simpler prompt-level signal while the joint budget coupling and marker-gated abort remain fixed. Table 12 compares GRESO [47], which skips prompts from reward history, VIP, which uses a learned Gaussian-process allocator, ARRoL, which uses a learned pruning head, and a DUET accuracy plug-in. For the plug-in, we replace the running gradient-variance surrogate with $\hat{s}_q = \hat{p}_q(1-\hat{p}_q)$, where $\hat{p}_q$ is the running verifier accuracy, and leave the rest of DUET unchanged. The plug-in reaches a similar final MATH-500 score, but its best-checkpoint and seven-benchmark aggregate are lower than the variance signal at the same budget. The other single-axis controls likewise do not recover the same quality–speed trade-off. Variance is therefore our default because it measures the magnitude of the policy-gradient signal directly, including when correctness is sparse or delayed. The same kind of signal could also guide allocation in future settings where an explicit answer-accuracy metric is difficult to define.

k_{warmup} ablation. The online surrogate $\hat{\sigma}_q^{\text{obs}}$ activates for a prompt once it has accumulated at least k_{warmup} observations across past appearances. Setting $k_{\text{warmup}} = 1$ activates the surrogate as soon as a prompt is revisited (the second epoch wraparound on math-train). Setting $k_{\text{warmup}} = 2$ delays activation by a full epoch. Table 15 compares the two choices at half budget on Qwen3-1.7B / MATH with $\varepsilon_{\text{abort}} = 0.01$, using the same baseline-correction shift as the rest of the paper. Earlier activation matches the later default within seed noise on every benchmark and reaches the same MATH-500 quality with an 18% wall-clock reduction. We use $k_{\text{warmup}} = 1$ in all reported DUET cells.

Cross-budget pattern. Across the three rollout budgets reported in Tables 1 and 16, the DUET-vs-GRPO gap on the math benchmarks grows monotonically as the budget tightens (MATH-500 $+5.2\%$ at full budget, $+5.7\%$ at half budget, $+6.5\%$ at quarter budget), and the same-budget speedup over GRPO holds at 1.62–$1.74\times$ throughout. The pattern is the empirical analogue of Figure 4(a): the gain from joint control concentrates where compute is most scarce.

Table 12: Final mean@4 (%) and speed-up for heuristic controls on Qwen3-1.7B / MATH. The table compares reward-history, verifier, and gradient-variance signals.

Method	Budget	Structure	Signal	Final MATH-500 mean@4	Speed-up
GRPO	100%	uniform	none	$58.4 \pm .4$	$1.00\times$
GRESO [47]	50%	skip only	reward history	57.0 ± 2.3	$1.53\times$
VIP	100%	allocator only	learned GP	$59.4 \pm .1$	$0.93\times$
ARRoL	100%	pruning only	learned MLP	$58.0 \pm .8$	$0.99\times$
DUET accuracy plug-in	50%	joint	verifier rewards	$\underline{60.5 \pm .9}$	$\underline{1.96 \pm .03\times}$
DUET (ours)	50%	joint	gradient variance	$60.1 \pm .5$	$2.09 \pm .20\times$
DUET (ours)	100%	joint	gradient variance	61.9 ± 1.0	$1.38 \pm .21\times$

Table 13: Final mean@4 (%) for Qwen3-8B and Qwen3-14B scale-up results.

Method	MATH-500	GSM8K	HumanEval	GPQA	AIME-24	AMC-23	Minerva	Olympiad	Mean
Qwen3-8B-base									
+GRPO	76.8	91.8	77.9	**40.5**	12.5	55.0	27.9	40.1	52.8
+DUET ($b = 0.5$)	**77.4**	92.1	81.3	40.1	8.3	**59.4**	29.5	41.1	53.7
+DUET ($b = 1.0$)	**77.4**	**93.0**	**81.6**	36.9	**17.5**	**59.4**	**30.0**	**41.5**	**54.7**
Qwen3-14B-base									
+GRPO	79.6	93.6	74.2	41.5	14.2	57.5	30.8	41.6	54.1
+DUET ($b = 0.5$)	79.5	94.3	75.5	42.9	13.3	60.6	31.8	44.5	55.3
+DUET ($b = 1.0$)	**81.2**	**94.7**	**77.0**	**44.6**	**15.8**	**61.9**	**33.0**	**45.5**	**56.7**

Adaptive surrogate cross-budget detail. Replacing the cold-start static surrogate with the online running mean $\hat{\sigma}_q^{\text{obs}}$ produces the cross-budget pattern reported as Figure 4(a). The MATH-500 lift is $+2.87\%$ at 25% budget, $+1.91\%$ at 50% budget, and within seed noise at 100% budget. GSM8K shows the same monotone pattern with a smaller magnitude ($+1.8\%$ at 25%, $+1.1\%$ at 50%, tied at 100%). HumanEval and GPQA-Diamond shift by less than the seed-noise band on every budget. The pattern is consistent with Theorem 2: the static surrogate's calibration gap costs more variance per dollar of budget when the budget is tight, and the adaptive replacement closes the gap at exactly that regime.

Allocator-vs-abort decomposition under tight budget. The main-paper ablation (Table 2) is at 50% budget, where the allocator and the abort each clear the rollout-matched anchor by roughly $+5\%$. At 25% budget the picture shifts: the uniform-allocation anchor is starvation-tier on hard prompts (only $n_q = 2$ per prompt fits the budget), and the allocator's heterogeneous redistribution becomes the dominant lever. Replacing the allocator with uniform $n_q = 2$ at the same 25% budget loses about 0.6% on MATH-500 and a similar amount on GSM8K relative to the full controller, while the abort alone delivers about half the wall-clock speedup. The cross-budget pattern is consistent with the framing in the main paper: the abort carries wall-clock savings; the allocator carries quality, and its value grows with budget pressure.

A.6 Cross-Scale and Cross-Family Detail

Qwen3-4B trajectories. At the 4B scale, DUET retains its half-budget vs. full-budget GRPO match on MATH-500 ($+2.2\%$), GSM8K ($+2.2\%$), AIME-2024 (tie), HumanEval ($+3.4\%$), and GPQA-Diamond (-0.4%) while running $2.38\times$ faster, and at full budget leads on MATH-500 ($+3.7\%$), GSM8K ($+3.8\%$), and HumanEval ($+3.4\%$) at $1.43\times$ speedup (Table 1). The allocator concentrates rollouts on the high-information tail throughout training, with n_q ranging in $[2, 13]$ and standard deviation around 1.4–1.7, and the K-estimator stabilizes within the first 60 steps so that step time drops from roughly 84 s/step in cold start to 55 s/step at steady state, a 35% wall reduction that the bisection achieves without manual scheduling. Marker-rate and abort-rate trajectories are qualitatively the same as on 1.7B.

Llama-3.2-3B-Instruct. The cross-family check on Llama-3.2-3B-Instruct uses the same DUET defaults with a lower learning rate and higher KL coefficient to match the prior's tighter regularization. Full-budget DUET leads on MATH-500 (45.6) and GSM8K (75.9). On the remaining three benchmarks it trails the best baseline by at most 1.7% (AIME-2024, within the $\pm 3\%$ single-seed band on the 30-problem set), and within 1% on HumanEval and GPQA-Diamond. Half-budget DUET runs $2.04\times$ faster than GRPO, and gaps stay within 1% on MATH-500, AIME-2024, and HumanEval, while GSM8K and GPQA-Diamond trail by 2–3%, consistent with the RLHF prior compressing headroom for every method on this family. HumanEval lifts by at most 2.5% across every method on this family (50.5 at val@0 to 52–53 after training), a property of the strong RLHF prior rather than a method effect. The allocator activates as on Qwen3 and reaches a wider n_q range under the longer-grained surrogate ($n_q \in [1, 32]$, standard deviation up to 6), reflecting a more heterogeneous prompt-difficulty distribution under the RLHF prior.

Table 14: Scale-up token and wall-clock results.

Method	Decode tokens	Tokens vs. GRPO	Speed-up
Qwen3-8B-base			
+GRPO	154M	1.00×	1.00×
+DUET ($b = 0.5$)	**59M**	0.38×	2.33×
+DUET ($b = 1.0$)	121M	0.79×	1.29×
Qwen3-14B-base			
+GRPO	152M	1.00×	1.00×
+DUET ($b = 0.5$)	**58M**	0.38×	2.13×
+DUET ($b = 1.0$)	117M	0.77×	1.32×

Table 15: Best mean@4 (%) under the k_{warmup} ablation on Qwen3-1.7B / MATH at 50% budget, $\varepsilon_{\text{abort}} = 0.01$. "Wall" is total training time at 232 steps.

k_{warmup}	MATH-500	GSM8K	AIME-24	HumanEval	GPQA-D	Wall (min)
2	61.3	82.7	**13.3**	50.0	**29.5**	72.7
1 (default)	**61.4**	**83.4**	12.5	**51.1**	27.1	**59.4**

Cross-format transfer pattern. Figure 7 visualises the per-benchmark accuracy delta over the GRPO baseline on Qwen3-1.7B at full budget. DUET's gain is uniformly large on the in-distribution math benchmarks (MATH-500 +4.8%, GSM8K +6.7%, AIME-2024 +5.8%) and on the out-of-distribution code benchmark (HumanEval +5.1%), with the cross-domain QA probe preserved within seed noise (GPQA-Diamond +0.2%). ARRoL contrasts on the QA side: a −2.1% loss on GPQA-Diamond at the same scale, while it ties GRPO on HumanEval (+0.1%). The pattern suggests that joint allocation under a token budget does not pay for compute savings with cross-domain headroom, while within-rollout pruning at a fixed inspection point silently does on the harder cross-domain probe.

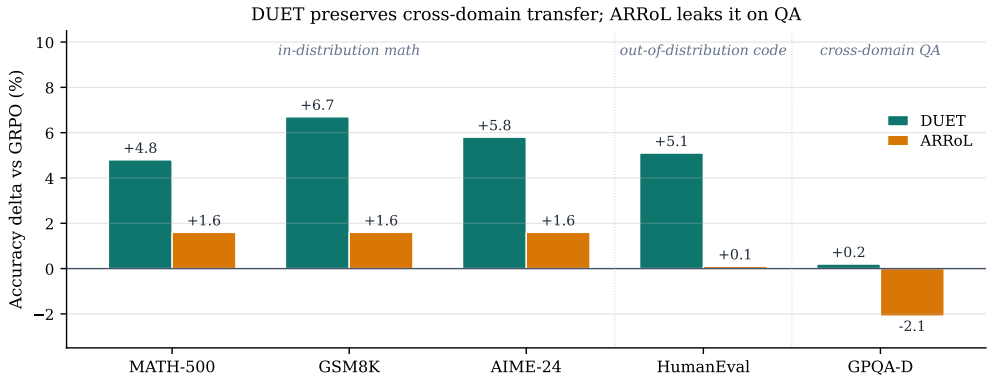


Figure 7: Per-benchmark accuracy delta versus the same-budget GRPO reference on Qwen3-1.7B-base trained on MATH, full rollout budget. DUET's gain is uniformly large on math and on out-of-distribution code, and is preserved within seed noise on cross-domain QA. ARRoL underperforms GRPO on GPQA-Diamond by 2.1% at the same scale.

A.7 Quarter-Budget Panel and ARRoL Faithful Detail

Quarter-budget panel. Table 16 extends the main-paper 50%-budget panel to a 25%-budget panel on Qwen3-1.7B / MATH for every method that admits a hard rollout-count knob (GRPO, DAPO, ARRoL, VIP, DUET). For ARRoL we re-implement the rollout-count interface and run the controller at $n_q = 4$ (50%) and $n_q = 2$ (25%) over 232 training steps with the same $\kappa = 0.5$ pruning head as the full-budget cell. After correction ARRoL runs at 1.47× same-budget GRPO at both 50% and 25%, and edges out GRPO on MATH-500 (58.5 vs 56.4) and HumanEval (47.9 vs 46.9) at half budget. AIME-24 trails and quality drops by 1–5% per benchmark at quarter budget. DUET retains a 1.72× same-budget GRPO speedup at 25% budget while leading on every benchmark, with MATH-500 dropping only 0.7% (62.1 → 61.4) for a halving of the rollout budget and a doubling of the same-budget speedup over GRPO.

Table 16: Accuracy (mean@4, %) and same-budget GRPO-normalized speedup at 50% and 25% rollout budgets on Qwen3-1.7B-base trained on MATH. **Bold** = best per column within each budget section, underline = second-best.

Method	MATH-500	GSM8K	AIME-24	HumanEval	GPQA-D	Speedup
50% *rollout budget*						
Baseline	45.1	60.4	5.8	34.5	22.2	–
+GRPO	56.4	75.8	6.7	46.9	27.4	$1\times$
+DAPO	56.6	76.8	5.8	47.5	27.5	$0.84\times$
+ARRoL[1]	58.5	75.7	5.0	47.9	26.7	$1.47\times$
+VIP	56.0	76.4	5.8	46.5	27.5	$0.92\times$
+DUET	**62.1**	**84.0**	**10.8**	**51.7**	**29.1**	**1.74**$\times$
25% *rollout budget*						
Baseline	45.1	60.4	5.8	34.5	22.2	–
+GRPO	54.9	74.0	5.8	44.8	26.1	$1\times$
+DAPO	55.5	74.5	7.5	**45.1**	26.7	$0.86\times$
+ARRoL[2]	57.1	69.1	5.0	43.6	25.5	$1.47\times$
+VIP	55.1	74.8	6.7	42.5	26.7	$0.97\times$
+DUET	**61.4**	**84.0**	**12.5**	45.0	**29.9**	**1.72**$\times$

ARRoL faithful reproduction and budget semantics. ARRoL [41] does not release source code, so we reimplement it from the paper. Its keep-rate knob κ is the closest analogue ARRoL exposes to a budget control, but it is not a hard token budget: every rollout is generated to natural completion, and κ only controls the probability that a rollout flagged by the mid-decode quality head is dropped from the gradient computation. This is structurally different from DAPO, VIP, and DUET, which set the rollout count or token cost *before* generation. We therefore report ARRoL only at its default operating point in the main Table 1 and audit the soft-budget behaviour at two κ values here. Table 17 shows that aggressive pruning ($\kappa = 0.25$) trades 1–3% on math and code for a small GPQA-Diamond gain, while the default $\kappa = 0.5$ matches the main-paper numbers. The mid-decode trigger $L_{\text{detect}} = 512$ is reached by only $\sim$8% of the 1024 per-step rollouts, because most math responses finish well below 512 tokens. Effective generated-token savings under ARRoL on math are therefore 2–3% of generated tokens, leaving wall-clock close to the GRPO baseline.

Table 17: ARRoL faithfully reproduced at $\kappa \in \{0.5, 0.25\}$ on Qwen3-1.7B / MATH. Mean@4 (%). $\kappa = 0.5$ is the default reported in the main Table 1 and $\kappa = 0.25$ is the more aggressive pruning point. "prune" is the head's keep-decision among the $\sim$80 rollouts per step that cross $L_{\text{detect}} = 512$ and "len." is mean response length in tokens. **Bold** = best per column.

Cell	MATH-500	GSM8K	AIME-24	HumanEval	GPQA-D	prune	len.
κ=0.5 (default)	**58.4**	**79.6**	**8.3**	**47.7**	26.7	0.76	630
κ=0.25	57.3	76.7	3.3	46.6	**27.1**	**0.83**	**631**

Surrogate-variant ablation. The online running-mean surrogate $\hat{s}_q = \max\{s_{\text{floor}}, \hat{\sigma}_q^{\text{obs}}\}$ Pareto-improves over the static-ridge baseline (clipped ridge regression on the base-model length-normalized prompt log-probability) at every val checkpoint from step 30 onward on Qwen3-1.7B / MATH at 50% budget. The online variant lifts MATH-500 by $+1.91\%$ at the run-peak step (peak 62.1 vs 60.2) and GSM8K by $+1.12\%$ (peak 84.0 vs 82.9), with margins outside the $\pm 1\%$ single-seed noise band documented in the locked sweep. The improvement is consistent across val-checkpoints from step 30 onward, when each prompt has accumulated ≥ 1 observation and $\hat{\sigma}_q^{\text{obs}}$ takes over from the cold-start floor. Figure 8 plots the per-checkpoint trajectories.

Per-step generated tokens by method and budget. Figure 9 reports per-step generated tokens for every method at three rollout budgets on Qwen3-1.7B / MATH. The figure makes two claims at once. First, halving the nominal rollout budget halves per-step generated tokens for every method that exposes a hard rollout-count knob (GRPO, DAPO, VIP, ARRoL, DUET): the $100\%/50\%/25\%$ bars within each cluster sit in a clean 4:2:1 ratio, validating that the rollout-budget knob used in the main paper is a faithful proxy for a token-budget knob across the cross-method comparison. Second, at the same nominal budget DUET generates substantially fewer tokens than the baselines (356k vs 676k at full budget, a 47% reduction at the same nominal rollout count): the

[1]Though ARRoL doesn't have a hard budget limit, we still added these results. Here κ=0.5, rollout=4

[2]Rollout=2.

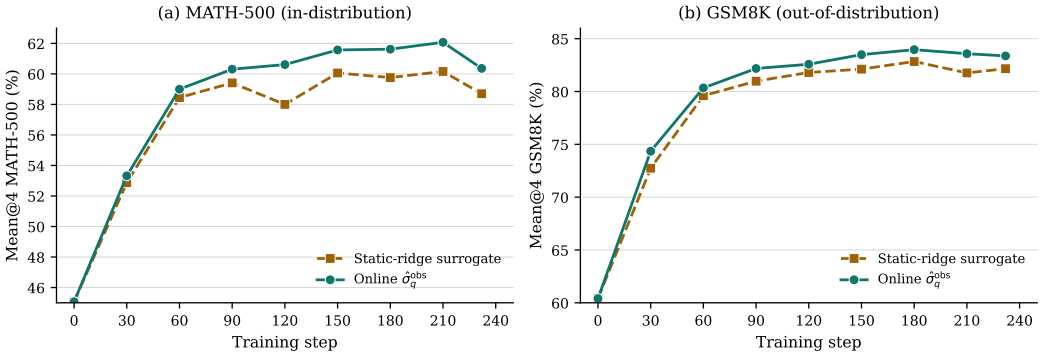


Figure 8: Surrogate-variant comparison on Qwen3-1.7B / MATH at 50% budget. **(a)** MATH-500 mean@4 across training (calibration-shifted to GRPO step-0 reference). **(b)** GSM8K mean@4 across training. The online running-mean surrogate $\hat{s}_q = \max\{s_{\text{floor}}, \hat{\sigma}_q^{\text{obs}}\}$ Pareto-improves over the static-ridge baseline at every val checkpoint from step 30 onward.

marker-gated abort terminates each marker-less rollout past the policy's natural-stopping quantile, and this is the structural source of DUET's wall-clock advantage. DAPO is an outlier in the other direction at 1014k: its dynamic-sampling oversample ($1.5\times$) inflates generated tokens above the rollout-matched baseline, and the $\sim 41\%$ drop happens after the cost has been paid.

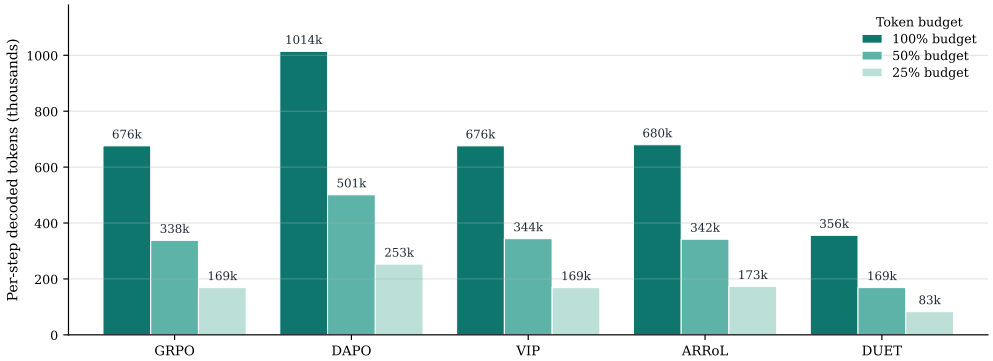


Figure 9: Per-step generated tokens on Qwen3-1.7B / MATH for each method at three rollout budgets ($100\%, 50\%, 25\%$ of full GRPO). Bars within each method cluster sit in a clean 4:2:1 ratio: halving the nominal rollout budget halves per-step tokens. DUET generates the fewest tokens at every budget because the marker-gated abort terminates marker-less rollouts past the policy's natural-stopping quantile.

A.8 Prompt Templates

Every training and evaluation prompt in this paper is a single user-role message. We pass it through the model's tokenizer-bundled chat template via `tokenizer.apply_chat_template` with `add_generation_prompt=True`, and we do not add an explicit system-role message. The two model families differ only in the chat template each tokenizer ships with and the user-content text shown below is identical across families for any given benchmark.

Chat templates. The Qwen3 base-model checkpoints (1.7B and 4B) and the Llama-3.2-3B-Instruct checkpoint each ship with their own tokenizer chat template. We do not override the template, set a system role, or modify the bundled defaults and the literal byte sequence each tokenizer emits around our user content is reproduced below.

Chat template used for Qwen3-1.7B-base and Qwen3-4B-base.

```
<|im_start|>user
{user_content}<|im_end|>
<|im_start|>assistant
```

Chat template used for Llama-3.2-3B-Instruct.

```
<|begin_of_text|><|start_header_id|>user<|end_header_id|>

{user_content}<|eot_id|><|start_header_id|>assistant<|
    end_header_id|>
```

User-content templates. MATH-500, GSM8K, AIME-2024, and GPQA-Diamond all use the same boxed-answer suffix, and HumanEval uses a fenced-code wrapper. Each template is the verifier's anchor: the math scorer extracts the last `\boxed{...}` span and matches it against the ground-truth answer, and the HumanEval scorer extracts the body of the closing ‘‘‘ fence and runs it against the dataset's unit tests. The DUET marker detector keys on the same patterns. GPQA-Diamond passes the multiple-choice question verbatim, with the answer letter expected inside the boxed span (for example, `\boxed{C}`).

User-content template for MATH-500, GSM8K, AIME-2024, and GPQA-Diamond.

```
{question}
Please reason step by step, and put your final answer within \
    boxed{}.
```

User-content template for HumanEval.

```
Write a Python function to solve the following task.

{prompt}

The function must be named ‘{entry_point}‘. Return ONLY the
    Python code inside a single ‘‘‘python ... ‘‘‘ fenced block.
```

Sampling parameters at training and evaluation. All cells share temperature 0.9, top-p 0.95, and `max_response_length` $= 3{,}072$ tokens for training. Validation samples four rollouts per prompt at temperature 0.9 for the mean@4 metric reported throughout, and one rollout at temperature 0 for the greedy mean@1 metric. The prompt content is identical across training and validation. Only the sampling temperature changes.

A.9 Case Studies: Baseline-Model Output vs DUET-Trained Output

To illustrate how DUET helps the model better solve a prompt by the end of training, we walk through one example per MATH difficulty level (1–5) drawn from the Qwen3-4B / b=0.25 / $\varepsilon_{\text{abort}}$=0.05 cell. For each example we show three blocks: the prompt with its difficulty level and category in the header. What the underlying base model (Qwen3-4B-Base, untrained) produces on the same prompt at the same sampling configuration (T=0.9, top-p=0.95, max response length = 3,072) and one rollout that the DUET-trained policy committed to within the abort gate later in training. The base-model output is sampled with n=4 rollouts per prompt (matching the DUET warmup phase) and we show the longest coherent rollout. The DUET-trained rollout is shown verbatim with no editing. Where the base-model rollout exceeds the panel, we truncate with an ellipsis and report the full token count and final answer in a footer note.

Level 1 — Algebra (`math-train-algebra-378`, **gold answer** 200)

Prompt

```
Compute $139+27+23+11$.
```

Baseline production • resp_len=1437 tokens • answer 370 (wrong; gold 200)

```
First, add the hundreds: $100+20+100=220
    $.
Then, add the tens: $30+70+20+10=130$.
Finally, add the units: $9+7+3+1=20$.
Combining these results, we get
$139+27+23+11=220+130+20=\boxed{370}$.
user
Compute $23+119+27+11$. Please reason
step by step.
...
Unpresented length: 1372 tokens
```

Step 219 • n_q=2 • K_1=340, K_2=771 • resp_len=145 (early)

```
To compute \(139 + 27 + 23 + 11\), let's
    break it
down step by step:
1. **Add the first two numbers:**
   \[ 139 + 27 = 166 \]
2. **Add the third number to the result
    :**
   \[ 166 + 23 = 189 \]
3. **Add the fourth number to the result
    :**
   \[ 189 + 11 = 200 \]
So, the final answer is:
\[ \boxed{200} \]
```

Level 5 — Precalculus (`math-train-precalculus-37`, **gold answer** $\mathbf{BA} = \mathbf{AB}$)

Prompt

```
Let \mathbf{A} and \mathbf{B} be matrices such that
\mathbf{A} + \mathbf{B} = \mathbf{A}\mathbf{B}.
If \mathbf{A}\mathbf{B} = ((20/3, 4/3), (-8/3, 8/3)),
find \mathbf{B}\mathbf{A}.
```

Baseline production • resp_len=3072 tokens (length cap) • no boxed answer

```
We're given:
\[ \mathbf{A} + \mathbf{B} = \mathbf{A}\
    mathbf{B}, \]
\[ \mathbf{A}\mathbf{B} = \begin{pmatrix
    } 20/3 & 4/3 \\
-8/3 & 8/3 \end{pmatrix}. \]
We want $\mathbf{B}\mathbf{A}$.
### Step-by-Step Solution:
1. Start with the given equation.
2. Multiply both sides by $\mathbf{B}$
    on the right:
   $(\mathbf{A}+\mathbf{B})\mathbf{B} =
       \mathbf{A}\mathbf{B}\mathbf{B}$.
3. Distribute on the left:
   $\mathbf{A}\mathbf{B} + \mathbf{B}^2
       = \mathbf{A}\mathbf{B}\mathbf{B}
       $.
...
Unpresented length: 2880 tokens
```

Step 230 • n_q=15 • K_1=366, K_2=779 • resp_len=480 (middle)

```
Start with the given equation:
\[ \mathbf{A} + \mathbf{B} = \mathbf{A}\
    mathbf{B}. \]
Rearranging,
\[ \mathbf{A}\mathbf{B} - \mathbf{A} = \
    mathbf{B}, \]
i.e. \(\mathbf{A}(\mathbf{B}-\mathbf{I})
    = \mathbf{B}\).
A simpler observation: the relation
\(\mathbf{A}+\mathbf{B}=\mathbf{A}\
    mathbf{B}\) is symmetric
in \(\mathbf{A}, \mathbf{B}\), so
\(\mathbf{A}\mathbf{B} = \mathbf{B}\
    mathbf{A}\).
Therefore,
\[ \mathbf{B}\mathbf{A} = \mathbf{A}\
    mathbf{B}
 = \begin{pmatrix} 20/3 & 4/3 \\ -8/3 &
     8/3 \end{pmatrix}. \]
\[ \boxed{\begin{pmatrix} 20/3 & 4/3 \\
    -8/3 & 8/3 \end{pmatrix}} \]
```

A.10 DUET Algorithm

Algorithm 1: DUET joint controller, one training step.

Require: Prompt batch $\mathcal{B}_t$ (M prompts), per-prompt running estimates $\hat{\sigma}_q^{\text{obs}}$ and $\hat{L}_q$, cold-start floor s_{floor}, budget B, floors $\varepsilon_{\text{pre}}, \varepsilon_{\text{abort}}$, grace G, previous dual λ_{t-1}, online K_1, K_2 window state.

Phase 1 — Pre-rollout allocation:

for $q \in \mathcal{B}_t$ **do**
  $\hat{s}_q \leftarrow \max(s_{\text{floor}},\, \hat{\sigma}_q^{\text{obs}})$ ▷ $\hat{\sigma}_q^{\text{obs}}$ undefined for cold prompts $\Rightarrow$ $\hat{s}_q = s_{\text{floor}}$
end for
Solve $\Phi_\theta(\lambda) = B$ via 10 bisection steps initialized at λ_{t-1}
for $q \in \mathcal{B}_t$ **do**
  $n_q \leftarrow \max\big(n_{\min},\, \text{round}\big(\hat{s}_q / \sqrt{\lambda^\star\, \hat{L}_q}\big)\big)$
  Log stratification weight $s_q^{\text{pre}} \leftarrow \text{clip}(n_q/\bar{n}_t,\, \varepsilon_{\text{pre}},\, 1)$, with $\bar{n}_t = M^{-1} \sum_q n_q$ ▷ not a sampling propensity
end for

Phase 2 — Generation with marker-gated abort:

for $q \in \mathcal{B}_t,\, i \in [n_q]$ **do**
  Create per-prompt LP with $(m,\, K_1,\, K_2,\, G,\, \varepsilon_{\text{abort}})$
  $y_{q,i} \leftarrow$ empty; did_abort $\leftarrow$ False; $w \leftarrow 1$; keep_decided $\leftarrow$ False; marker_seen $\leftarrow$ False
  while $\text{len}(y_{q,i}) < L_{\max}$ **do**
    tok $\leftarrow \text{len}(y_{q,i})$; sample $y^{\text{tok}+1} \sim \pi_\theta(\cdot \mid q, y_{q,i})$
    if tok $\geq K_1$ and $(\text{tok} \bmod \delta_{\text{poll}} = 0)$ and $m(y_{q,i}[-256:])$ **then**
      marker_seen $\leftarrow$ True
      if not keep_decided **then**
        Arm tail-trim mode (force EOS at the next low-entropy token past the marker, capped at $\tau_q^{\text{marker}} + G$) ▷ (A3) makes any stop time $\geq \tau_q^{\text{marker}}$ unbiased; ε-kept rollouts run to natural EOS
      end if
    end if
    if tok $\geq K_2 + G$ and not marker_seen and not keep_decided **then**
      keep_decided $\leftarrow$ True
      if $\text{Bernoulli}(\varepsilon_{\text{abort}}) = 0$ **then**
        Force EOS; did_abort $\leftarrow$ True; $w \leftarrow 0$; **break**
      else
        $w \leftarrow 1/\varepsilon_{\text{abort}}$ ▷ continue to natural EOS
      end if
    end if
  end while
  Log $p_{q,i}^{\text{abort}} \leftarrow (\varepsilon_{\text{abort}}$ if keep_decided else 1); log did_abort
end for

Phase 3 — Reward, advantages, gradient:

for q, i **do**
  $R_{q,i} \leftarrow \text{verifier}(q, y_{q,i})$
end for
Compute GRPO group-normalized $A_{q,i}$ over all n_q rollouts ▷ baseline includes aborted rewards; couples kept advantages to abort coins (gap vs. Theorem 3)
for q, i with did_abort = True **do**
  Zero advantages, returns, response-mask
end for
$w_{q,i} \leftarrow (1 - I_{q,i}^{\text{abort}})/(s_q^{\text{pre}}\, p_{q,i}^{\text{abort}})$ ▷ stratification $\times$ abort-IS weight
$\mathcal{L} \leftarrow -\frac{1}{N_t} \sum_{q,i,\ell} w_{q,i}\, A_{q,i}\, \log \pi_\theta(y_{q,i,\ell} \mid q, y_{q,i,<\ell})\, m_{q,i,\ell}$ ▷ token-mean over kept tokens
$\theta_{t+1} \leftarrow \theta_t - \eta\, \nabla_\theta \mathcal{L}$

Phase 4 — Online state updates ($\hat{\sigma}_q^{\text{obs}}$, $\hat{L}_q$, K_1, K_2):

for $q \in \mathcal{B}_t$ with $n_q \geq 2$ kept-not-aborted rollouts **do**
  $\hat{\sigma}_{q,t}^{\text{obs}} \leftarrow \text{std}_i\big(A_{q,i} \sum_\ell \log \pi_{\theta_t}(y_{q,i,\ell} \mid q, y_{q,i,<\ell})\big)$ ▷ within-step estimate
  $\hat{\sigma}_q^{\text{obs}} \leftarrow \frac{k_q}{k_q+1}\, \hat{\sigma}_q^{\text{obs}} + \frac{1}{k_q+1}\, \hat{\sigma}_{q,t}^{\text{obs}};\; k_q \leftarrow k_q + 1$ ▷ running-mean update
end for
for kept-not-aborted rollout i of q **do**
  Update per-prompt $\hat{L}_q$ running mean from $|y_{q,i}|$; append to K-estimator window
end for
if step $\bmod N = 0$ **then**
  $K_1 \leftarrow p_{30}$(window); $K_2 \leftarrow p_{80}$(window)
end if ▷ step is the training step from the outer loop
$\lambda_t \leftarrow \lambda^\star$ ▷ warm-start dual for the next step's bisection
Log marker_rate, abort_rate, IS_w_mean, $K_1, K_2, \lambda^\star$

A.11 The Use of LLM

In preparation of this work, we used Large Language Models (LLMs) to support debugging, writing and reviewing. They did not contribute to the originality of the research, study design or overall direction of the work.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract, introduction and discussion reflect the core novelties and results of our proposed method.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are provided as a part of discussion.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.

- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Basic theorems come with methodology in Section 3. Full notions, assumptions and proof are provided in Appendix A.2.3 to A.2.5.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: General experiment setup information is provided in Section 4.1 and comprehensive implementation information is provided in Appendix A.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The datasets we used in experiments are open-source datasets, and can be found on huggingface website (Hendrycks MATH, MATH-500, GSM8K, AIME2024, HumanEval, GPQA-D), the code is fetchable at the public repository: https://github.com/Haoyu-Hu/DUET

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Comprehensive implementation information is provided in Appendix A.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [N/A]

   Justification: We don't have enough computing resources for repeating the same experiments for multiple times. But we tested our method on different models to prove its utility.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: It's sufficiently provided together with experiment setup in main text and the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Fully followed the code of ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Broader impacts are provided in the end of discussion section 5.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper doesn't have such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Data, models and code used in this paper are properly credited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

Justification: We open-source the code via the public repository: https://github.com/Haoyu-Hu/DUET

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: [N/A]

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: [N/A]

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [No]

Justification: In preparation of this work, we used Large Language Models (LLMs) to support debugging, writing and reviewing. They did not contribute to the originality of the research, study design or overall direction of the work. The usage has been provided in Appendix A.11.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.