Learning When to Think: Dual-Reference Offline Optimization for Adaptive VLM Reasoning

Hongbin Lin[1] **Sizhe Zou**[1] **Juangui Xu**[2] **Xinyue Xu**[3]
Zhuoli Ouyang[1] **Stanislav Kolchin**[1] **Chengwei Qin**[1,4]
Zhongxiang Dai[5] **Yao Shu**[1,4*]

[1]The Hong Kong University of Science and Technology (Guangzhou)
[2]CISPA Helmholtz Center for Information Security
[3]The Chinese University of Hong Kong
[4]The Hong Kong University of Science and Technology
[5]The Chinese University of Hong Kong, Shenzhen

Abstract

Reasoning traces improve vision-language models on complex multimodal tasks, but forcing every input to follow a long reasoning process leads to overthinking and unnecessary inference cost. We study adaptive fast/slow thinking for VLMs, where one model answers directly for simple inputs and reasons step by step when needed. Fast answers and slow traces often come from distinct behavior policies, making a single-reference offline objective misweight one mode. We propose DUAL-REFERENCE BOLT (DR-BOLT), which distinguishes the desired fast/slow mixture reference from the sampling distribution and corrects their mismatch in reward-weighted training. Fast/slow partition competition explains routing in the optimal target; an efficient source-wise correction approximates the exact objective when response formats are well separated. Matched Qwen3-VL comparisons use identical candidate banks to isolate the objective from slow-teacher quality. DR-BOLT improves score–length trade-offs over the evaluated offline and online baselines on ScienceQA and three held-out transfer benchmarks, with a 24.1% cross-benchmark slow ratio and a $2.90\times$ end-to-end training speedup over reproduced GRPO. These results support adaptive reasoning in the studied two-reference setting, while broader robustness remains open. Our code is available at https://github.com/HongbinLin3589/DR-BOLT.

1 Introduction

Many multimodal tasks require more than direct visual recognition. Visual mathematical reasoning, scientific question answering, and expert-level multimodal exams often require a VLM to inspect visual details, combine them with textual context, and perform multi-step inference [15, 27, 41, 39]. Reasoning traces have therefore become an important mechanism for improving model accuracy on complex inputs. Chain-of-thought and related reasoning methods have shown strong gains in language models [29, 28, 38, 7, 11], and recent VLMs extend this idea by training models to produce explicit intermediate reasoning before answering visual problems [32, 37, 6, 40, 25, 36].

However, always using slow reasoning is inefficient. Many practical VLM inputs can be answered by direct recognition, short visual grounding, or simple text-image matching. For such inputs, forcing a long reasoning trace increases response length, latency, memory usage, and serving cost, and prior studies report that unnecessary thinking can hurt simple cases or create avoidable failure

*Corresponding author.

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

modes [12, 9, 8, 33]. The desired behavior is therefore input-dependent: a VLM should reason carefully when multi-step inference is needed, but answer directly when slow reasoning provides little benefit.

We study *adaptive thinking* for VLMs, where a single model internalizes when to produce a concise answer and when to generate a longer reasoning process. Recent methods explore this goal through thinking and non-thinking modes, length-based mode labels, difficulty-aware rewards, thought dropout, and bi-mode optimization [34, 13, 31, 26, 35]. These methods demonstrate that adaptive thinking can improve the accuracy–efficiency trade-off. We focus on a complementary offline setting: generate candidate responses in advance, score them by correctness and efficiency, and train a model to prefer the better response family for each input.

The key observation of this work is that offline adaptive thinking is not merely a problem of controlling response length, but a dual-reference optimization problem. Fast direct answers and slow reasoning traces are often generated by distinct behavior policies: concise answers may come from an instruct or no-think policy, while reasoning traces may come from a thinking or chain-of-thought policy. This matters for offline reward optimization. BOLT decouples response generation from optimization and projects a prompt-normalized, reward-tilted reference distribution through weighted next-token training [23]. Yet standard BOLT assumes a single reference policy. For adaptive VLM thinking, forcing fast and slow trajectories into one shared reference can misweight one response family and drive the learned model toward always-fast or always-slow behavior.

We propose DR-BOLT, an offline optimization framework for adaptive VLM thinking. Instead of treating direct answers and reasoning traces as two styles sampled from one policy, our method models them as samples from two behavior policies and optimizes under a fast/slow mixture reference. We distinguish the desired reference mixture from the distribution used to sample candidates; their ratio corrects the offline weights. Partition competition explains the target's fast/slow allocation, and one trained checkpoint produces adaptive responses at inference. Matched internal comparisons share candidate banks, initialization, rewards, and training budget, including mode-conditioned reward-weighted SFT and multi-reference DPO. This isolates the objective's contribution from the stronger slow teacher. We evaluate ScienceQA [15] in distribution and transfer from ViRL39K to MathVision [27], DynaMath [41], and MMMU-Pro [39]. The latter are held-out evaluations; shared task types and incomplete source-level overlap audits preclude a strict OOD claim.

Our contributions are summarized as follows:

- We formulate offline adaptive VLM thinking as a *dual-reference optimization* problem, exposing why single-reference BOLT is mismatched for fast/slow trajectories generated by distinct behavior policies.
- We derive the behavior-reference correction and partition-based routing interpretation of DUAL-REFERENCE BOLT, with an efficient source-wise approximation. Our contribution is the mismatch diagnosis and its practical correction; the mixture-reference optimum follows standard KL-regularized optimization.
- Matched baselines, correction ablations, and sensitivity analyses show improved score–length trade-offs on ScienceQA and three held-out transfer benchmarks, while identifying limitations of the curated two-reference setting.

2 Related Work

2.1 Reasoning-oriented vision-language models

Large vision-language models have rapidly advanced multimodal understanding through visual instruction tuning and large-scale image-text alignment [1, 10, 14, 5, 2, 4, 3]. Beyond direct perception and instruction following, recent work has explored explicit reasoning in VLMs by transferring chain-of-thought style supervision and reinforcement learning from language models to multimodal tasks [29, 24, 22, 7]. Models such as LLaVA-CoT, Mulberry, OpenVLThinker, Vision-R1, VL-Rethinker, and R1-OneVision train VLMs to generate intermediate reasoning traces before producing final answers [32, 37, 6, 40, 25, 36]. These methods improve performance on complex visual reasoning tasks, but they typically encourage slow thinking for all inputs, which increases response length and can cause overthinking on simple examples. Our work focuses on adaptive reasoning:

preserving the benefits of slow thinking while avoiding unnecessary reasoning when direct answering is sufficient.

2.2 Adaptive fast/slow thinking in VLMs

A growing line of work studies how VLMs can decide whether to think. DualMindVLM learns fast and slow thinking modes using length-based mode labels and GRPO, enabling automatic switching between concise and reasoning-based responses [13]. FAST-GRPO introduces difficulty-aware rewards and adaptive length regularization to improve the accuracy–efficiency trade-off in large VLM reasoning [31]. TON trains VLMs to select whether to reason through thought dropout followed by reinforcement learning [26]. R-4B builds a general-purpose auto-thinking MLLM with bi-mode annealing and bi-mode policy optimization [35]. These methods show that adaptive thinking can reduce unnecessary computation while maintaining or improving accuracy. However, most existing approaches rely on online RL, explicit mode labels, handcrafted difficulty estimates, or bi-mode rollouts. In contrast, we study a fully offline setting where fast and slow candidate responses are generated in advance and optimized with reward-weighted learning [18, 30].

2.3 Offline reward optimization and the dual-reference gap

Offline alignment replaces online rollouts with supervised objectives over pre-generated responses. DPO optimizes preference pairs through a reference-regularized objective [21]; reward-augmented maximum likelihood and advantage-weighted regression connect reward-weighted learning with policy optimization [19, 20, 17]. BOLT provides a forward-KL projection onto the Boltzmann target of fixed-reference KL-regularized RL with verifiable rewards [23]. We use this framework because our scalar rewards retain correctness and length information for every candidate, and the mixture partition exposes fast/slow allocation. DPO remains a valid alternative; Section 5 compares a mixture-reference DPO baseline and mode-conditioned reward-weighted SFT on the same bank. Our contribution concerns the mismatch between the desired fast/slow reference and the candidate-sampling distribution, rather than a new general solution to KL-regularized optimization.

3 Preliminaries

Adaptive fast/slow thinking. Let $x = (I, Q)$ denote a multimodal input with image I and text query Q. A VLM can answer x in two modes. In the *fast* mode, it produces a concise direct response y_{f}. In the *slow* mode, it produces a reasoning trace followed by a final answer, denoted as $y_{\mathrm{s}} = (c, a)$, where c is the trace and a is the answer. We write the mode as $z \in \{\mathrm{f}, \mathrm{s}\}$. Adaptive thinking aims to train a single policy $\pi_\theta(y|x)$ whose generation behavior internalizes when to answer directly and when to reason step by step, without a separately trained test-time router.

Single-reference BOLT. We build on KL-regularized RL with verifiable rewards. Given a reference policy π_{ref}, a reward $r(x, y)$, and a temperature $\beta > 0$, the objective is

$$J_{\mathrm{RL}}(\pi_\theta) = \mathbb{E}_x \left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)} r(x, y) - \beta \mathrm{KL}\left(\pi_\theta(\cdot|x) \| \pi_{\mathrm{ref}}(\cdot|x)\right) \right]. \tag{1}$$

BOLT shows that this objective can be optimized offline under a fixed reference policy [23]. For each input x, the optimal policy is a reward-reweighted reference distribution:

$$\pi^\star(y|x) = \frac{\pi_{\mathrm{ref}}(y|x) \exp(r(x, y)/\beta)}{Z(x)}, \quad Z(x) = \mathbb{E}_{y\sim\pi_{\mathrm{ref}}(\cdot|x)} \left[\exp(r(x, y)/\beta)\right]. \tag{2}$$

Minimizing the forward KL from $\pi^\star$ to π_θ yields the weighted SFT objective

$$\mathcal{L}_{\mathrm{BOLT}}(\theta) = -\mathbb{E}_x \mathbb{E}_{y\sim\pi_{\mathrm{ref}}(\cdot|x)} \left[\frac{\exp(r(x, y)/\beta)}{Z(x)} \log \pi_\theta(y|x) \right]. \tag{3}$$

The derivation is given in Appendix C. In practice, $Z(x)$ is estimated with Monte Carlo samples from the fixed reference. Thus, BOLT decouples generation from optimization: responses are generated and weighted once, and the policy is trained by weighted supervised fine-tuning.

Why BOLT for adaptive reasoning? BOLT identifies a prompt-normalized Boltzmann target rather than arbitrary reward-weighted SFT, retains scalar reward magnitudes, and exposes fast/slow partition masses. Generation, scoring, and weight construction precede weighted next-token optimization, avoiding repeated online rollouts and verifier calls. DPO [21] is also a valid offline alternative, but reward-derived preference pairs retain relative order rather than all reward magnitudes. Section 5 compares mixture-reference DPO and mode-conditioned reward-weighted SFT; we do not claim universal superiority over other offline objectives.

Why single-reference BOLT is insufficient. The above formulation assumes that all samples come from one reference distribution π_{ref}. Adaptive VLM thinking often violates this assumption because fast direct answers and slow reasoning traces are commonly generated by different behavior policies. A slow trace can have very low likelihood under a fast reference, while a concise answer can have low likelihood under a slow reference. Single-reference BOLT can therefore misweight one response family and may lead to always-fast or always-slow behavior. The next section formalizes this mismatch as a dual-reference optimization problem.

4 Dual-Reference BOLT

Standard BOLT assumes a single fixed reference policy. Adaptive VLM thinking often violates this assumption because fast direct answers and slow reasoning traces are generated by distinct behavior policies. We therefore replace the single reference with a fast/slow mixture reference and derive an offline reward-weighted objective under this dual-reference distribution. This formulation reveals two useful structures: routing in the optimal target is governed by fast/slow partition competition, and the behavior-reference correction admits a mode-wise approximation under mode-format separation.

4.1 Dual-reference objective

We denote the fast direct-answer policy by π_{f} and the slow reasoning policy by π_{s}. We define the desired reference mixture as

$$\rho(y|x) = \alpha(x)\pi_{\mathrm{f}}(y|x) + \big(1-\alpha(x)\big)\pi_{\mathrm{s}}(y|x), \tag{4}$$

where $\alpha(x) \in (0,1)$ is the prior probability of the fast mode. The actual offline generation process may use a different sampling ratio:

$$\mu(y|x) = q(x)\pi_{\mathrm{f}}(y|x) + \big(1-q(x)\big)\pi_{\mathrm{s}}(y|x), \tag{5}$$

where $q(x)$ is the empirical probability of sampling a fast response. The distinction between ρ and μ is important: ρ is the reference distribution that defines the regularized target, while μ is the distribution that actually produces the offline data. Unless otherwise specified, we use $\alpha(x) = 0.5$ and $q(x) = 0.7$, corresponding to a response bank with 70% fast candidates and 30% slow candidates. This makes the behavior-reference correction active in the main setting: the mode-wise constants are $\kappa(y_{\mathrm{f}}) = 0.5/0.7 = 0.714$ and $\kappa(y_{\mathrm{s}}) = 0.5/0.3 = 1.667$. We additionally use the balanced case $\alpha = q = 0.5$ as a sanity check for the correction term.

We use an accuracy-efficiency reward

$$r(x,y) = r_{\mathrm{acc}}(x,y) + \gamma r_{\mathrm{fmt}}(y) - \lambda \ell(y), \tag{6}$$

where r_{acc} measures answer correctness, r_{fmt} encourages valid answer or reasoning format, and $\ell(y)$ is the normalized output length. The coefficient λ controls the trade-off between accuracy and inference cost.

Theorem 1 (Dual-reference optimal policy)**.** *Consider the KL-regularized objective*

$$J_{\mathrm{DR}}(\pi_\theta) = \mathbb{E}_x\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)} r(x,y) - \beta \mathrm{KL}\left(\pi_\theta(\cdot|x)\|\rho(\cdot|x)\right)\right]. \tag{7}$$

For each fixed input x, the optimal policy is

$$\pi^\star_{\mathrm{DR}}(y|x) = \frac{\rho(y|x)\exp(r(x,y)/\beta)}{Z_\rho(x)}, \tag{8}$$

where

$$Z_\rho(x) = \mathbb{E}_{y\sim\rho(\cdot|x)}\left[\exp(r(x,y)/\beta)\right]. \tag{9}$$

The proof follows the standard Lagrange multiplier derivation and is given in Appendix D.1. Theorem 1 states that the target policy under the dual-reference objective is a reward-reweighted version of the fast/slow mixture. Standard BOLT is recovered as a special case when $\pi_{\mathrm{f}} = \pi_{\mathrm{s}} = \pi_{\mathrm{ref}}$, in which case $\rho = \mu = \pi_{\mathrm{ref}}$.

4.2 Router-free adaptive thinking

The dual-reference partition function decomposes into fast and slow components:

$$Z_\rho(x) = \alpha(x) Z_{\mathrm{f}}(x) + \big(1 - \alpha(x)\big) Z_{\mathrm{s}}(x), \tag{10}$$

where

$$Z_{\mathrm{f}}(x) = \mathbb{E}_{y \sim \pi_{\mathrm{f}}(\cdot|x)} \left[\exp(r(x,y)/\beta)\right], \quad Z_{\mathrm{s}}(x) = \mathbb{E}_{y \sim \pi_{\mathrm{s}}(\cdot|x)} \left[\exp(r(x,y)/\beta)\right]. \tag{11}$$

Corollary 1 (Latent routing by partition competition)**.** *The fast-source marginal of the reward-reweighted joint target is*

$$P^\star(z = \mathrm{f}|x) = \frac{\alpha(x) Z_{\mathrm{f}}(x)}{\alpha(x) Z_{\mathrm{f}}(x) + \big(1 - \alpha(x)\big) Z_{\mathrm{s}}(x)}. \tag{12}$$

Similarly, $P^\star(z = \mathrm{s}|x) = 1 - P^\star(z = \mathrm{f}|x)$.

The proof is given in Appendix D.3. Here z denotes the reference source in the joint mixture. This marginal corresponds to observable fast/slow output frequencies when sources align with response formats; the trained policy's format-based slow ratio is an empirical diagnostic. Corollary 1 shows that the optimal dual-reference target contains an implicit soft routing rule rather than a separately trained classifier. For easy inputs, fast and slow responses may both be correct, but fast responses are shorter, making $Z_{\mathrm{f}}(x)$ dominate. For hard inputs, fast responses may fail while slow responses succeed, making $Z_{\mathrm{s}}(x)$ dominate despite the length penalty. Thus, Eq. (12) can be interpreted as a reward-normalized routing score induced by the two references.

4.3 From dual-reference RL to weighted SFT

Theorem 1 defines the target policy, but direct sampling from $\pi^\star_{\mathrm{DR}}$ is intractable. Following BOLT, we minimize the forward KL from the optimal policy to the trainable policy.

Proposition 1 (Dual-reference weighted SFT)**.** *Minimizing*

$$\mathbb{E}_x \left[\mathrm{KL}\left(\pi^\star_{\mathrm{DR}}(\cdot|x) \| \pi_\theta(\cdot|x)\right)\right]$$

is equivalent, up to constants independent of θ*, to minimizing*

$$\mathcal{L}_\rho(\theta) = -\mathbb{E}_x \mathbb{E}_{y \sim \rho(\cdot|x)} \left[\frac{\exp(r(x,y)/\beta)}{Z_\rho(x)} \log \pi_\theta(y|x)\right]. \tag{13}$$

If samples are drawn from μ *rather than* ρ*, and* ρ *is absolutely continuous with respect to* μ*, the corrected offline objective is*

$$\mathcal{L}_\mu(\theta) = -\mathbb{E}_x \mathbb{E}_{y \sim \mu(\cdot|x)} \left[\kappa(x,y) \frac{\exp(r(x,y)/\beta)}{Z_\rho(x)} \log \pi_\theta(y|x)\right], \tag{14}$$

where

$$\kappa(x,y) = \frac{\rho(y|x)}{\mu(y|x)}. \tag{15}$$

The proof is a forward-KL expansion plus a change of measure; see Appendix D.2. Proposition 1 shows that the offline weight has two factors: a reward-induced factor and a behavior-reference correction factor. Exact correction remains valid under arbitrary overlap; mode separation is required only for the pointwise mode-wise approximation. When $q(x) = \alpha(x)$, we have $\mu = \rho$, so $\kappa(x,y) = 1$ exactly. This balanced case is a sanity check, while the main non-balanced setting exposes and validates the correction term.

4.4 Practical mode-wise correction

Exact computation of $\kappa(x, y)$ requires evaluating both π_{f} and π_{s} on every sampled response, which is expensive for large VLMs. We use the following empirical structure of fast/slow outputs: the two modes are often separated by response format and generation prefix, while length further amplifies the mismatch for long reasoning traces. This motivates an approximate mode-format separation assumption. Offline source labels are recorded from the generating policy, rather than inferred from response length or a format classifier.

Assumption 1 (Approximate mode-format separation)**.** *For fast samples y_{f} and slow samples y_{s}, define*

$$\delta_{\mathrm{f}}(x, y_{\mathrm{f}}) = \frac{\pi_{\mathrm{s}}(y_{\mathrm{f}}|x)}{\pi_{\mathrm{f}}(y_{\mathrm{f}}|x)}, \quad \delta_{\mathrm{s}}(x, y_{\mathrm{s}}) = \frac{\pi_{\mathrm{f}}(y_{\mathrm{s}}|x)}{\pi_{\mathrm{s}}(y_{\mathrm{s}}|x)}.$$

We assume there exist small constants $\varepsilon_{\mathrm{f}}, \varepsilon_{\mathrm{s}} \ll 1$ such that

$$\delta_{\mathrm{f}}(x, y_{\mathrm{f}}) \le \varepsilon_{\mathrm{f}}, \quad \delta_{\mathrm{s}}(x, y_{\mathrm{s}}) \le \varepsilon_{\mathrm{s}}. \tag{16}$$

The constants capture mode-format mismatch. Both references and their offline samples are fixed, so these ratios do not evolve as the trainable policy changes during Phase 2. In practice, ε_{s} may be further reduced by length compounding on long reasoning traces, while ε_{f} may be dominated by prefix-level mismatch.

Proposition 2 (Mode-wise correction)**.** *Under Assumption 1, if $\alpha(x), q(x) \in [\eta, 1-\eta]$ for some $\eta > 0$, then*

$$\kappa(x, y_{\mathrm{f}}) = \frac{\alpha(x)}{q(x)} + O(\varepsilon_{\mathrm{f}}), \quad \kappa(x, y_{\mathrm{s}}) = \frac{1-\alpha(x)}{1-q(x)} + O(\varepsilon_{\mathrm{s}}). \tag{17}$$

The proof is given in Appendix D.4. Proposition 2 shows that dual-reference correction can be approximated without expensive dual-model likelihood evaluation when mode-format separation holds. Appendix E.1 validates the assumption with a cross-likelihood matrix, and Appendix E.2 compares exact and mode-wise corrections. The exact residual magnitudes are $|q-\alpha|\delta_{\mathrm{f}}/[q(q+(1-q)\delta_{\mathrm{f}})]$ and $|q-\alpha|\delta_{\mathrm{s}}/[(1-q)(1-q+q\delta_{\mathrm{s}})]$, so overlap increases pointwise approximation error smoothly. On the main path, exact and mode-wise log corrections have correlation 0.9902, with 1.2% of samples differing by more than 0.01. An ablation path has 3–5% reasoning-style completions in the fast pool and lower correlation 0.9730. Exact correction is available for ambiguous samples; filtering reasoning markers or fast-pool outputs longer than 30 generated tokens is another format-control option.

Single-reference collapse mechanism. Assumption 1 also explains why single-reference BOLT can collapse in this setting. If a fast reference is used for all samples, then slow samples are suppressed by a factor proportional to δ_{s}. If a slow reference is used for all samples, then fast samples are suppressed by a factor proportional to δ_{f}. Thus, single-reference BOLT does not merely use a weaker approximation; under mode separation, it systematically downweights one response family and can drive the model toward always-fast or always-slow behavior.

4.5 Empirical objective

For each input x, we sample fast and slow response sets:

$$\mathcal{Y}_{\mathrm{f}}(x) = \{y_i^{\mathrm{f}}\}_{i=1}^{N_{\mathrm{f}}}, \quad y_i^{\mathrm{f}} \sim \pi_{\mathrm{f}}(\cdot|x),$$

$$\mathcal{Y}_{\mathrm{s}}(x) = \{y_j^{\mathrm{s}}\}_{j=1}^{N_{\mathrm{s}}}, \quad y_j^{\mathrm{s}} \sim \pi_{\mathrm{s}}(\cdot|x).$$

We estimate

$$\widehat{Z}_{\mathrm{f}}(x) = \frac{1}{N_{\mathrm{f}}} \sum_{i=1}^{N_{\mathrm{f}}} \exp(r(x, y_i^{\mathrm{f}})/\beta), \quad \widehat{Z}_{\mathrm{s}}(x) = \frac{1}{N_{\mathrm{s}}} \sum_{j=1}^{N_{\mathrm{s}}} \exp(r(x, y_j^{\mathrm{s}})/\beta), \tag{18}$$

and

$$\widehat{Z}_{\rho}(x) = \alpha(x)\widehat{Z}_{\mathrm{f}}(x) + \big(1-\alpha(x)\big)\widehat{Z}_{\mathrm{s}}(x). \tag{19}$$

Algorithm 1 DUAL-REFERENCE BOLT

Require: Inputs $\mathcal{X}$, fast policy π_{f}, slow policy π_{s}, prior $\alpha(x)$, sampling ratio $q(x)$, samples N_{f}, N_{s}, reward r, temperature β

Phase 0: Mixed warm-up

Train π_θ on a small unweighted mixture of fast and slow samples.

Phase 1: Dual-reference generation

Initialize $\mathcal{D}_{\mathrm{DR}} \leftarrow \emptyset$.

for each $x \in \mathcal{X}$ **do**

  Sample $\mathcal{Y}_{\mathrm{f}}(x) = \{y_i^{\mathrm{f}}\}_{i=1}^{N_{\mathrm{f}}}$ from $\pi_{\mathrm{f}}(\cdot|x)$.

  Sample $\mathcal{Y}_{\mathrm{s}}(x) = \{y_j^{\mathrm{s}}\}_{j=1}^{N_{\mathrm{s}}}$ from $\pi_{\mathrm{s}}(\cdot|x)$.

  Compute $r(x, y)$ for all $y \in \mathcal{Y}_{\mathrm{f}}(x) \cup \mathcal{Y}_{\mathrm{s}}(x)$.

  Estimate $\widehat{Z}_{\mathrm{f}}(x)$, $\widehat{Z}_{\mathrm{s}}(x)$, and $\widehat{Z}_{\rho}(x)$.

  Compute $w_{\mathrm{DR}}(x, y)$ for all sampled responses using Eq. (20).

  Add all weighted triples $(x, y, w_{\mathrm{DR}}(x, y))$ to $\mathcal{D}_{\mathrm{DR}}$.

end for

Phase 2: Offline optimization

Optimize π_θ on $\mathcal{D}_{\mathrm{DR}}$ using Eq. (21).

return adaptive policy π_θ.

The practical DR-BOLT weight is

$$w_{\mathrm{DR}}(x, y) = \widehat{\kappa}(x, y) \cdot \frac{\exp(r(x, y)/\beta)}{\widehat{Z}_{\rho}(x)}, \tag{20}$$

where $\widehat{\kappa}$ is either the exact correction in Eq. (15) or the mode-wise approximation in Eq. (17). The final training objective is

$$\boxed{\mathcal{L}_{\text{DR-BOLT}}(\theta) = -\sum_{(x,y)\in\mathcal{D}_{\mathrm{DR}}} w_{\mathrm{DR}}(x, y) \log \pi_\theta(y|x)}. \tag{21}$$

4.6 Training algorithm

DR-BOLT uses three phases. Phase 0 performs a short mixed warm-up, because the mixture reference ρ is a distribution rather than a single physical checkpoint. This warm-up is used to initialize the trainable policy near both response formats before reward-weighted optimization. Phase 1 generates fast and slow responses and computes DR-BOLT weights. Phase 2 optimizes weighted next-token likelihood. The references, response bank, rewards, mixture parameters, and correction weights are frozen before Phase 2; no on-policy rollouts recompute these weights.

4.7 Discussion

DR-BOLT differs from standard BOLT by changing the reference distribution from a single policy to a fast/slow mixture. This makes the optimal policy, partition function, routing probability, and sample weights mode-dependent. The formulation provides a router-free mechanism in the target distribution: fast responses dominate when their reward-weighted partition mass is larger, while slow responses dominate when their correctness gain outweighs their length cost. The behavior-reference correction κ is needed because fast and slow trajectories come from different policies, and Proposition 2 makes this correction practical under mode-format separation. When the two policies produce highly overlapping formats, exact correction or additional filtering may be needed; we empirically examine routing calibration, mode collapse, mode-format separation, and exact-versus-mode-wise correction in Section 5. Both behaviors are useful when each has a reward advantage on part of the input distribution. If slow reasoning adds no compensating correctness benefit, concentration toward fast responses is the intended reward-optimal solution, rather than pathological collapse. A reference's single-decoding accuracy is not a theoretical ceiling because reward tilting and multi-candidate coverage can improve the target distribution. Nevertheless, the population target remains on the mixture support, and the empirical method receives reward evidence only from sampled candidates; it provides no unconstrained online exploration.

Table 1: Our reproductions of adaptive-thinking methods and GRPO. ScienceQA uses same-benchmark training; the other datasets measure transfer after ViRL39K training. Cells report Score/Length in words. Runtime includes candidate generation or online rollouts and all training stages; memory is peak per GPU. Speedup is relative to GRPO.

Dataset	GRPO	FAST-GRPO [31]	TON [26]	DualMindVLM [13]	**DR-BOLT (Ours)**
			Score / Length		
ScienceQA (IID)	0.962/201.0	0.964/37.6	0.963/35.9	0.872/87.2	**0.967/20.4**
MathVision (transfer)	0.463/2498.2	0.527/1245.7	0.488/1505.8	0.302/446.0	**0.606/1058.1**
DynaMath (transfer)	0.769/1402.2	0.782/512.4	0.775/495.5	–	**0.799/370.6**
MMMU-Pro (transfer)	0.498/2006.4	0.540/892.6	0.519/747.3	–	**0.581/385.7**
Training Type	online RL / GRPO	online RL / GRPO	online RL / GRPO	online RL / GRPO	offline dual-ref BOLT
Time (h)	202.4	176.6	248.9	–	**69.7**
Memory (GB)	70.2	71.8	67.5	66.2	**60.2**
Speedup	1.00×	1.15×	0.81×	–	**2.90×**

Table 2: Matched internal comparisons. Cells report Score/Length in words; ScienceQA and cross-benchmark results use separately trained checkpoints. Slow ratio (%) aggregates only MathVision, DynaMath, and MMMU-Pro.

Dataset	Qwen3-Instruct	Qwen3-Thinking	Mix SFT	Reward SFT	Single-ref Fast	Single-ref Slow	**DR-BOLT**
				Score / Length			
ScienceQA (IID)	0.943/7.2	0.954/288.3	0.955/100.1	0.956/89.3	0.945/9.3	0.955/317.7	**0.967/20.4**
MathVision (transfer)	0.244/395.8	0.368/5775.3	0.406/3777.6	0.421/3802.9	0.301/474.7	0.439/5901.8	**0.606/1058.1**
DynaMath (transfer)	0.613/205.5	0.654/2696.7	0.730/1896.1	0.741/1675.0	0.627/241.4	0.752/2022.9	**0.799/370.6**
MMMU-Pro (transfer)	0.278/265.3	0.377/3082.3	0.472/1352.6	0.484/1105.9	0.412/443.3	0.408/2575.1	**0.581/385.7**
Slow Ratio (%)	0.0	99.8	67.2	60.4	1.9	97.4	24.1

5 Experiments

5.1 Setup

We train on the ScienceQA training split and evaluate on its official test split [15]. We also study cross-benchmark transfer by training on ViRL39K [25] and evaluating on MathVision [27], DynaMath [41], and MMMU-Pro [39]. These are held-out evaluations rather than strictly verified source- or template-level OOD settings. We add a second same-benchmark evaluation on a constructed disjoint 3DSRBench split [16] (Appendix H.7).

Our fast and slow references are Qwen3-VL-8B-Instruct and Qwen3-VL-8B-Thinking. Matched offline methods share the candidate bank, reward, initialization, and optimization budget. In addition to the baselines in Table 2, we evaluate multi-reference DPO (MR-DPO) and mode-conditioned reward-weighted SFT (Cond-RW). We reproduce GRPO, FAST-GRPO [31], TON [26], and DualMindVLM [13] using the common training inputs and evaluation pipeline; their method-specific rewards remain different. Appendix F and Appendix H give protocols and implementation details.

Length is mean response word count, excluding the prompt. Slow ratio is format-based, using the reasoning marker or slow-thinking format rather than a length threshold. In mixed benchmark tables, this ratio aggregates only MathVision, DynaMath, and MMMU-Pro evaluations of the ViRL39K-trained checkpoint; ScienceQA uses a separately trained checkpoint.

5.2 Main Results

Tables 1 and 2 report Score/Length. Scores use the same benchmark-specific answer extraction across methods. Training time is measured end to end on four A100-80GB GPUs: DR-BOLT includes offline fast/slow candidate generation, reward computation, weight construction, and policy optimization; online baselines include rollout generation and updates. Memory is peak per-GPU usage, and speedup is relative to reproduced GRPO.

Comparison with external adaptive-thinking baselines. DR-BOLT has the highest score among the evaluated methods on all four benchmarks. Relative to reproduced GRPO, its score gains are 0.005, 0.143, 0.030, and 0.083, and responses are 9.85×, 2.36×, 3.78×, and 5.20× shorter, respectively. End-to-end training takes 69.7 rather than 202.4 hours (2.90× speedup), with peak per-GPU memory of 60.2 rather than 70.2GB. These are method-level comparisons under our common-data reproduction protocol.

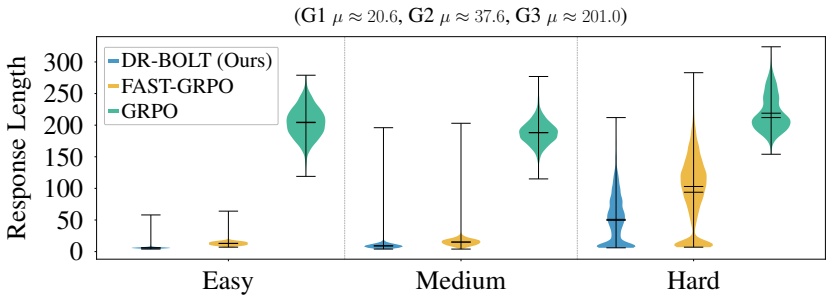


Figure 1: Difficulty-wise response-Length distributions. Each group contains the response-Length distributions for DR-BOLT, FAST-GRPO, and GRPO. DR-BOLT stays short on easy examples and increases reasoning Length on harder examples, while GRPO-style training remains much longer.

Internal baselines and single-reference collapse. The matched response bank controls for slow-teacher quality: all offline methods see the same fast and slow candidates. Mix SFT and Reward SFT remain longer and lower-scoring than DR-BOLT. Single-ref Fast and Single-ref Slow have cross-benchmark slow ratios of 1.9% and 97.4%, versus 24.1% for DR-BOLT, consistent with systematic mode suppression in these runs.

Stronger offline alternatives and sensitivity. MR-DPO uses the same mixture reference and reward-ranked preference pairs; Cond-RW explicitly learns a fast/slow source token. Under matched candidate banks and response-token budgets, their MathVision results are $0.568/1398.4$ and $0.513/1810.7$, versus $0.606/1058.1$ for DR-BOLT. The same ranking holds on the other three benchmarks (Table 3). This supports the observed advantage for this scalar-reward bank, without establishing universal superiority over preference optimization. One-factor sensitivity studies show that α controls slow usage, while changing q from 0.7 to 0.3 changes ScienceQA score by 0.002 after correction. Increasing λ from 0 to 0.10 reduces ScienceQA Length from 28.9 to 15.7 words; scores lie in $[0.964, 0.969]$. The shared main setting is not tuned separately on the transfer test sets (Appendix H.2).

5.3 Score–Length Dynamics

Figure 2 tracks the training dynamics on ScienceQA. DR-BOLT rapidly moves toward the upper-left region of the score–Length space: it increases accuracy from the initial slow model while reducing Length from 288.3 to about 20.4. GRPO improves accuracy but remains much longer, staying above 200 words at the end of training. FAST-GRPO reduces Length more aggressively than GRPO, but converges to a lower final score and longer responses than DR-BOLT. This result illustrates the benefit of offline reward-weighted optimization under a dual-reference mixture: the model learns to keep the accuracy benefit of reasoning while dropping unnecessary reasoning tokens on easy or already-solved examples.

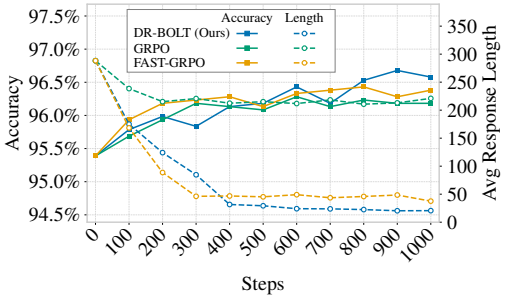


Figure 2: Score–Length dynamics on ScienceQA. Solid lines show accuracy and dashed lines show response Length.

5.4 Routing Behavior

We next evaluate whether DR-BOLT learns difficulty-dependent routing rather than simply shortening all responses. Figure 1 shows response-Length distributions across easy, medium, and hard groups. The desired behavior is not a fixed response length: easy examples should be answered directly, while harder examples should activate longer reasoning more often. DR-BOLT follows this pattern, producing very short responses on easy examples and increasing Length on harder examples. By contrast, GRPO-style training remains substantially longer across difficulty groups, reflecting less

selective use of slow reasoning. This supports the claim that DR-BOLT learns input-dependent allocation of reasoning effort.

5.5 Partition-based Routing Calibration

Finally, we validate the partition-competition mechanism in Corollary 1. For each input, we estimate $\widehat{Z}_{\mathrm{f}}(x)$ and $\widehat{Z}_{\mathrm{s}}(x)$, compute the predicted fast probability

$$P^{\star}(z = \mathrm{f}|x) = \frac{\alpha(x)\widehat{Z}_{\mathrm{f}}(x)}{\alpha(x)\widehat{Z}_{\mathrm{f}}(x) + (1-\alpha(x))\widehat{Z}_{\mathrm{s}}(x)},$$

and compare it with the empirical fast rate under repeated auto-mode decoding. Because each prompt is decoded only a finite number of times, we group prompts into equal-frequency bins according to predicted fast probability. Figure 3 shows that the empirical fast rate closely follows the predicted fast probability across groups. Across 100 prompts, the predicted probability and empirical fast rate have Pearson correlation $r = 0.939$, Spearman correlation $\rho = 0.929$, and grouped calibration error 0.029. These results indicate that the model's routing behavior is well explained by fast/slow partition competition rather than by an external classifier or random length variation.

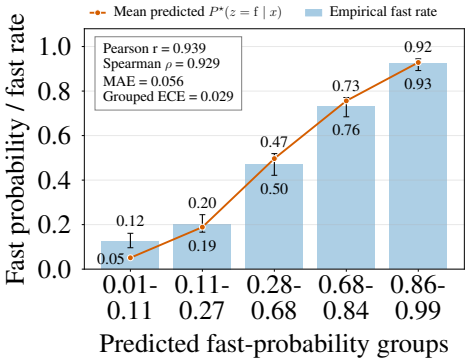


Figure 3: Partition-based routing calibration. Bars show empirical fast rate and markers show predicted fast probability.

5.6 Controlled Offline Comparisons

Table 3: Controlled offline alternatives on identical candidate banks. Cells report score / response words. ScienceQA uses a separate checkpoint; slow ratio aggregates the three transfer benchmarks.

Method	ScienceQA	MathVision	DynaMath	MMMU-Pro	Slow ratio (%)
MR-DPO	0.964/31.8	0.568/1398.4	0.787/506.9	0.557/528.6	29.4
Cond-RW	0.960/48.6	0.513/1810.7	0.761/714.8	0.521/689.5	39.7
DR-BOLT	0.967/20.4	0.606/1058.1	0.799/370.6	0.581/385.7	24.1

MR-DPO and Cond-RW share the same candidate bank, reward, frozen references, initialization, and optimization budget as DR-BOLT. Table 3 isolates the objective's gain from the stronger slow teacher: DR-BOLT improves both score and length on every benchmark in this comparison.

Ablation of behavior-reference correction. With the ViRL39K bank and optimization setup fixed at $\alpha = 0.5, q = 0.7$, removing κ drives the cross-benchmark slow ratio to 0.13% and scores to $0.260/0.626/0.314$ on MathVision/DynaMath/MMMU-Pro. Exact correction restores a 22.0% slow ratio and scores of $0.623/0.804/0.594$; mode-wise correction yields 24.1% and $0.606/0.799/0.581$. These controls attribute the restored trade-off to the correction applied to identical candidates (Appendix G.1).

Exact and mode-wise correction differ by at most 0.017 in score in these runs.

6 Conclusion

We presented DR-BOLT for offline adaptive VLM reasoning. Distinguishing the desired fast/slow reference from the candidate-sampling distribution yields a behavior-reference correction and a partition-based interpretation of target routing. Controlled comparisons with identical candidate banks show improved score–length trade-offs over the evaluated offline alternatives on ScienceQA and held-out transfer benchmarks. End-to-end accounting includes candidate generation and shows lower training cost than reproduced GRPO. Future work may extend the formulation to more reasoning modes, continuous budgets, bank refresh, and offline-online multimodal reasoning.

7 Acknowledgement

This work was supported by the Guangdong Basic and Applied Basic Research Foundation (Grant No. 2025A1515110115).

References

[1] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[2] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[3] S. Bai, Y. Cai, R. Chen, K. Chen, X. Chen, Z. Cheng, L. Deng, W. Ding, C. Gao, C. Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[4] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[5] W. Dai, J. Li, D. Li, A. Tiong, J. Zhao, W. Wang, B. Li, P. N. Fung, and S. Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *Advances in neural information processing systems*, 36:49250–49267, 2023.

[6] Y. Deng, H. Bansal, F. Yin, N. Peng, W. Wang, and K.-W. Chang. Openvlthinker: Complex vision-language reasoning via iterative sft-rl cycles. *Advances in Neural Information Processing Systems*, 38:123817–123846, 2026.

[7] D. Guo, D. Yang, H. Zhang, J. Song, P. Wang, Q. Zhu, R. Xu, R. Zhang, S. Ma, X. Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[8] M. Hassid, G. Synnaeve, Y. Adi, and R. Schwartz. Don't overthink it. preferring shorter thinking chains for improved llm reasoning. *arXiv preprint arXiv:2505.17813*, 2025.

[9] A. Kumar, J. Roh, A. Naseh, M. Karpinska, M. Iyyer, A. Houmansadr, and E. Bagdasarian. Overthink: Slowdown attacks on reasoning llms. *arXiv preprint arXiv:2502.02542*, 2025.

[10] J. Li, D. Li, S. Savarese, and S. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[11] J. Li, X. Xu, S. Ma, D. Zhang, and S. Li. Faithful-first reasoning, planning, and acting for multimodal llms. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 6777–6793, 2026.

[12] M. Li, J. Zhong, S. Zhao, Y. Lai, H. Zhang, W. B. Zhu, and K. Zhang. Think or not think: A study of explicit thinking in rule-based visual reinforcement fine-tuning. *arXiv preprint arXiv:2503.16188*, 2025.

[13] C. Lin, C. Chi, J. Wu, S. Li, and K. Zhou. Learning to think fast and slow for visual language models. *arXiv preprint arXiv:2511.16670*, 2025.

[14] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[15] P. Lu, S. Mishra, T. Xia, L. Qiu, K.-W. Chang, S.-C. Zhu, O. Tafjord, P. Clark, and A. Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in neural information processing systems*, 35:2507–2521, 2022.

[16] W. Ma, H. Chen, G. Zhang, Y.-C. Chou, J. Chen, C. De Melo, and A. Yuille. 3dsrbench: A comprehensive 3d spatial reasoning benchmark. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6924–6934. IEEE, 2025.

[17] J. Mu, T. Lin, C. Qin, Z. Dai, and Y. Shu. Drift: Decoupled rollouts and importance-weighted fine-tuning for efficient multi-turn optimization. *arXiv preprint arXiv:2605.31455*, 2026.

[18] J. Mu, Q. Zhang, Z. Wang, M. Yang, S. Qiu, C. Qin, Z. Dai, and Y. Shu. Self-reflective generation at test time. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10221–10239, 2026.

[19] M. Norouzi, S. Bengio, N. Jaitly, M. Schuster, Y. Wu, D. Schuurmans, et al. Reward augmented maximum likelihood for neural structured prediction. *Advances In Neural Information Processing Systems*, 29, 2016.

[20] X. B. Peng, A. Kumar, G. Zhang, and S. Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

[21] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

[22] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[23] Y. Shu, C. Wei, H. Lin, S. Qiu, and H. Xiong. Reference-sampled Boltzmann projection for KL-regularized RLVR: Target-matched weighted SFT, finite one-shot gaps, and policy mirror descent. arXiv:2605.02469, arXiv, 2026.

[24] A. Wang, L. Song, Y. Tian, B. Peng, L. Jin, H. Mi, J. Su, and D. Yu. Self-consistency boosts calibration for math reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6023–6029, 2024.

[25] H. Wang, C. Qu, Z. Huang, W. Chu, F. Lin, and W. Chen. Vl-rethinker: Incentivizing self-reflection of vision-language models with reinforcement learning. *arXiv preprint arXiv:2504.08837*, 2025.

[26] J. Wang, K. Q. Lin, J. Cheng, and M. Z. Shou. Think or not? selective reasoning via reinforcement learning for vision-language models. *arXiv preprint arXiv:2505.16854*, 2025.

[27] K. Wang, J. Pan, W. Shi, Z. Lu, H. Ren, A. Zhou, M. Zhan, and H. Li. Measuring multimodal mathematical reasoning with math-vision dataset. *Advances in Neural Information Processing Systems*, 37:95095–95169, 2024.

[28] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[29] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[30] H. Wu, B. Xu, Y. Shu, M. Yang, and C. Qin. Thinking with nothinking calibration: A new in-context learning paradigm in reasoning large language models. *arXiv preprint arXiv:2508.03363*, 2025.

[31] W. Xiao and L. Gan. Fast-slow thinking grpo for large vision-language model reasoning. *Advances in Neural Information Processing Systems*, 38:171601–171631, 2026.

[32] G. Xu, P. Jin, Z. Wu, H. Li, Y. Song, L. Sun, and L. Yuan. Llava-cot: Let vision language models reason step-by-step. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2087–2098, 2025.

[33] X. Xu, H. Lin, J. Xu, H. Wang, L. Wang, L. Hu, W. Liu, A. Weller, and X. Li. Concept-grounded reasoning with prompt-driven localization for interpretable structured report generation. *arXiv preprint arXiv:2609.15334*, 2026.

[34] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[35] Q. Yang, B. Ni, S. Xiang, H. Hu, H. Peng, and J. Jiang. R-4b: Incentivizing general-purpose auto-thinking capability in mllms via bi-mode annealing and reinforce learning. *arXiv preprint arXiv:2508.21113*, 2025.

[36] Y. Yang, X. He, H. Pan, X. Jiang, Y. Deng, X. Yang, H. Lu, D. Yin, F. Rao, M. Zhu, et al. R1-onevision: Advancing generalized multimodal reasoning through cross-modal formalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2376–2385, 2025.

[37] H. Yao, J. Huang, W. Wu, J. Zhang, Y. Wang, S. Liu, Y. Wang, Y. Song, H. Feng, L. Shen, et al. Mulberry: Empowering mllm with o1-like reasoning and reflection via collective monte carlo tree search. *arXiv preprint arXiv:2412.18319*, 2024.

[38] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

[39] X. Yue, T. Zheng, Y. Ni, Y. Wang, K. Zhang, S. Tong, Y. Sun, B. Yu, G. Zhang, H. Sun, et al. Mmmu-pro: A more robust multi-discipline multimodal understanding benchmark. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15134–15186, 2025.

[40] Y. Zhan, Y. Zhu, S. Zheng, H. Zhao, F. Yang, M. Tang, and J. Wang. Vision-r1: Evolving human-free alignment in large vision-language models via vision-guided reinforcement learning. *arXiv preprint arXiv:2503.18013*, 2025.

[41] C. Zou, X. Guo, R. Yang, J. Zhang, B. Hu, and H. Zhang. Dynamath: A dynamic visual benchmark for evaluating mathematical reasoning robustness of vision language models. In *13th International Conference on Learning Representations, ICLR 2025*, pages 89682–89728. International Conference on Learning Representations, ICLR, 2025.

A Notation

Table 4: Notation used in the paper.

Symbol	Meaning
$x = (I, Q)$	Multimodal input consisting of image I and text query Q.
y	Model response.
y_{f}	Fast response, usually a concise direct answer.
$y_{\mathrm{s}} = (c, a)$	Slow response containing reasoning trace c and final answer a.
$z \in \{\mathrm{f}, \mathrm{s}\}$	Latent response mode: fast or slow.
$\pi_\theta(y\|x)$	Trainable VLM policy.
$\pi_{\mathrm{f}}(y\|x)$	Fast behavior/reference policy.
$\pi_{\mathrm{s}}(y\|x)$	Slow behavior/reference policy.
$\pi_{\mathrm{ref}}(y\|x)$	Single reference policy used in standard BOLT.
$\pi^\star(y\|x)$	Optimal policy under single-reference BOLT.
$\pi^\star_{\mathrm{DR}}(y\|x)$	Optimal policy under dual-reference BOLT.
$\rho(y\|x)$	Desired dual-reference mixture distribution.
$\mu(y\|x)$	Empirical data-generating mixture distribution.
$\alpha(x)$	Desired fast-mode prior in ρ.
$q(x)$	Empirical fast sampling ratio in μ.
$r(x, y)$	Scalar reward for response y on input x.
$r_{\mathrm{acc}}(x, y)$	Correctness reward.
$r_{\mathrm{fmt}}(y)$	Format reward.
$\ell(y)$	Normalized length or compute cost used in the reward.
γ, λ	Weights for format reward and length penalty.
β	Temperature in KL-regularized RL / BOLT weighting.
$Z(x)$	Single-reference partition function.
$Z_\rho(x)$	Dual-reference partition function.
$Z_{\mathrm{f}}(x), Z_{\mathrm{s}}(x)$	Fast and slow reward-weighted partition functions.
$\widehat{Z}_{\mathrm{f}}(x), \widehat{Z}_{\mathrm{s}}(x), \widehat{Z}_\rho(x)$	Monte Carlo estimates of the corresponding partition functions.
$P^\star(z = \mathrm{f}\|x)$	Optimal fast-mode marginal probability induced by partition competition.
$\kappa(x, y)$	Exact behavior-reference correction $\rho(y\|x)/\mu(y\|x)$.
$\widehat{\kappa}(x, y)$	Practical correction used in empirical training.
$\delta_{\mathrm{f}}(x, y_{\mathrm{f}})$	Cross-mode likelihood ratio $\pi_{\mathrm{s}}(y_{\mathrm{f}}\|x)/\pi_{\mathrm{f}}(y_{\mathrm{f}}\|x)$.
$\delta_{\mathrm{s}}(x, y_{\mathrm{s}})$	Cross-mode likelihood ratio $\pi_{\mathrm{f}}(y_{\mathrm{s}}\|x)/\pi_{\mathrm{s}}(y_{\mathrm{s}}\|x)$.
$\varepsilon_{\mathrm{f}}, \varepsilon_{\mathrm{s}}$	Small constants controlling approximate mode-format separation.
η	Lower bound ensuring $\alpha(x), q(x) \in [\eta, 1 - \eta]$.
$\mathcal{Y}_{\mathrm{f}}(x), \mathcal{Y}_{\mathrm{s}}(x)$	Sampled fast and slow candidate response sets for input x.
$N_{\mathrm{f}}, N_{\mathrm{s}}$	Number of fast and slow samples per input.
$\mathcal{D}_{\mathrm{DR}}$	Weighted dual-reference training dataset.
$w_{\mathrm{DR}}(x, y)$	Final DR-BOLT sample weight.
$\mathcal{L}_{\mathrm{BOLT}}$	Single-reference BOLT weighted SFT objective.
$\mathcal{L}_{\mathrm{DR\text{-}BOLT}}$	Dual-reference BOLT weighted SFT objective.
$\mathbb{E}[\cdot]$	Expectation.
$\mathrm{KL}(\cdot\|\|\cdot)$	Kullback–Leibler divergence.
Length	Response length reported in experiments, measured by word count.
Slow Ratio	Fraction of generated responses classified as slow-thinking responses.

B Limitations

Our study focuses on adaptive fast/slow thinking with two response modes and evaluates the method on a representative set of multimodal reasoning benchmarks. Although the formulation can extend to more modes, multi-level reasoning budgets and tool-augmented settings remain future work. Offline optimization depends on the quality and coverage of the fixed response bank and does not provide unconstrained exploration beyond its reward evidence. The slow reference supplies successful reasoning trajectories; matched-bank comparisons isolate the objective's relative gain but do not remove this dependence.

Approximate mode-format separation is used for the efficient marginal-ratio approximation, rather than the exact dual-reference objective. When formats overlap, exact correction or additional filtering may be preferable. Format-based slow-ratio diagnostics may also be ambiguous for blended outputs.

The held-out transfer benchmarks may share task types or sources with ViRL39K, so these results do not establish strict OOD generalization. Finally, our efficiency comparison is measured under our reproduced setup, and absolute runtime may vary with hardware, inference engine, rollout length, and implementation details.

C Proof of Single-Reference BOLT

C.1 Closed-form optimal policy

For a fixed input x, consider the KL-regularized objective

$$J(\pi) = \sum_y \pi(y|x) r(x,y) - \beta \sum_y \pi(y|x) \log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)}. \tag{22}$$

We optimize $J(\pi)$ subject to the normalization constraint $\sum_y \pi(y|x) = 1$. The Lagrangian is

$$\mathcal{J}(\pi, \lambda) = \sum_y \pi(y|x) \left[r(x,y) - \beta \log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)} \right] + \lambda \left(\sum_y \pi(y|x) - 1 \right). \tag{23}$$

Taking the derivative with respect to $\pi(y|x)$ gives

$$r(x,y) - \beta \left(\log \pi(y|x) - \log \pi_{\text{ref}}(y|x) + 1 \right) + \lambda = 0. \tag{24}$$

Therefore,

$$\pi(y|x) = C(x) \pi_{\text{ref}}(y|x) \exp(r(x,y)/\beta), \tag{25}$$

where $C(x)$ is a normalization constant. Using $\sum_y \pi(y|x) = 1$, we obtain

$$C(x) = \frac{1}{\sum_y \pi_{\text{ref}}(y|x) \exp(r(x,y)/\beta)}. \tag{26}$$

Thus,

$$\pi^\star(y|x) = \frac{\pi_{\text{ref}}(y|x) \exp(r(x,y)/\beta)}{Z(x)}, \tag{27}$$

where

$$Z(x) = \sum_y \pi_{\text{ref}}(y|x) \exp(r(x,y)/\beta) = \mathbb{E}_{y \sim \pi_{\text{ref}}(\cdot|x)} \left[\exp(r(x,y)/\beta) \right]. \tag{28}$$

C.2 Forward KL to weighted SFT

The forward-KL objective is

$$\min_\theta \mathbb{E}_x \left[\text{KL} \left(\pi^\star(\cdot|x) \| \pi_\theta(\cdot|x) \right) \right]. \tag{29}$$

Expanding the KL divergence gives

$$\text{KL} \left(\pi^\star(\cdot|x) \| \pi_\theta(\cdot|x) \right) = \mathbb{E}_{y \sim \pi^\star(\cdot|x)} \left[\log \pi^\star(y|x) - \log \pi_\theta(y|x) \right]. \tag{30}$$

The first term does not depend on θ. Thus, the objective is equivalent to minimizing

$$-\mathbb{E}_x \mathbb{E}_{y \sim \pi^\star(\cdot|x)} \left[\log \pi_\theta(y|x) \right]. \tag{31}$$

Substituting Eq. (27) yields

$$-\mathbb{E}_x \mathbb{E}_{y \sim \pi_{\text{ref}}(\cdot|x)} \left[\frac{\exp(r(x,y)/\beta)}{Z(x)} \log \pi_\theta(y|x) \right], \tag{32}$$

which is the weighted SFT objective in Eq. (3).

D Proofs for Dual-Reference BOLT

D.1 Proof of Theorem 1

For a fixed input x, the dual-reference KL-regularized objective is

$$J(\pi) = \sum_y \pi(y|x) r(x,y) - \beta \sum_y \pi(y|x) \log \frac{\pi(y|x)}{\rho(y|x)}. \tag{33}$$

We optimize $J(\pi)$ subject to $\sum_y \pi(y|x) = 1$. The Lagrangian is

$$\mathcal{J}(\pi, \lambda) = \sum_y \pi(y|x) \left[r(x,y) - \beta \log \frac{\pi(y|x)}{\rho(y|x)} \right] + \lambda \left(\sum_y \pi(y|x) - 1 \right). \tag{34}$$

Taking the derivative with respect to $\pi(y|x)$ gives

$$r(x,y) - \beta \left(\log \pi(y|x) - \log \rho(y|x) + 1 \right) + \lambda = 0. \tag{35}$$

Therefore,

$$\pi(y|x) = C(x) \rho(y|x) \exp(r(x,y)/\beta). \tag{36}$$

The normalization condition gives

$$C(x) = \frac{1}{\sum_y \rho(y|x) \exp(r(x,y)/\beta)}. \tag{37}$$

Thus,

$$\pi^\star_{\mathrm{DR}}(y|x) = \frac{\rho(y|x) \exp(r(x,y)/\beta)}{Z_\rho(x)}, \tag{38}$$

where

$$Z_\rho(x) = \sum_y \rho(y|x) \exp(r(x,y)/\beta) = \mathbb{E}_{y \sim \rho(\cdot|x)} \left[\exp(r(x,y)/\beta) \right]. \tag{39}$$

This proves Theorem 1.

D.2 Proof of Proposition 1

The forward KL objective is

$$\min_\theta \mathbb{E}_x \left[\mathrm{KL} \left(\pi^\star_{\mathrm{DR}}(\cdot|x) \| \pi_\theta(\cdot|x) \right) \right]. \tag{40}$$

Expanding the KL divergence gives

$$\mathrm{KL} \left(\pi^\star_{\mathrm{DR}}(\cdot|x) \| \pi_\theta(\cdot|x) \right) = \mathbb{E}_{y \sim \pi^\star_{\mathrm{DR}}(\cdot|x)} \left[\log \pi^\star_{\mathrm{DR}}(y|x) - \log \pi_\theta(y|x) \right]. \tag{41}$$

The first term is independent of θ. Therefore, minimizing the forward KL is equivalent to minimizing

$$-\mathbb{E}_x \mathbb{E}_{y \sim \pi^\star_{\mathrm{DR}}(\cdot|x)} \left[\log \pi_\theta(y|x) \right]. \tag{42}$$

Using

$$\pi^\star_{\mathrm{DR}}(y|x) = \frac{\rho(y|x) \exp(r(x,y)/\beta)}{Z_\rho(x)}, \tag{43}$$

we obtain

$$-\mathbb{E}_x \mathbb{E}_{y \sim \rho(\cdot|x)} \left[\frac{\exp(r(x,y)/\beta)}{Z_\rho(x)} \log \pi_\theta(y|x) \right], \tag{44}$$

which gives Eq. (13).

If samples are instead drawn from μ, then for any function $g(x,y)$,

$$\mathbb{E}_{y \sim \rho(\cdot|x)}[g(x,y)] = \mathbb{E}_{y \sim \mu(\cdot|x)} \left[\frac{\rho(y|x)}{\mu(y|x)} g(x,y) \right], \tag{45}$$

assuming ρ is absolutely continuous with respect to μ. Taking

$$g(x,y) = \frac{\exp(r(x,y)/\beta)}{Z_\rho(x)} \log \pi_\theta(y|x) \tag{46}$$

gives Eq. (14) with

$$\kappa(x,y) = \frac{\rho(y|x)}{\mu(y|x)}. \tag{47}$$

This proves Proposition 1.

D.3 Proof of Corollary 1

Define the joint reference distribution over mode and response as

$$\rho(z, y|x) = \begin{cases} \alpha(x)\pi_{\mathrm{f}}(y|x), & z = \mathrm{f}, \\ \big(1-\alpha(x)\big)\pi_{\mathrm{s}}(y|x), & z = \mathrm{s}. \end{cases} \tag{48}$$

The reward-reweighted joint optimal policy is

$$\pi^{\star}_{\mathrm{DR}}(z, y|x) = \frac{\rho(z, y|x)\exp(r(x, y)/\beta)}{Z_{\rho}(x)}. \tag{49}$$

For the fast mode,

$$P^{\star}(z = \mathrm{f}|x) = \sum_{y} \pi^{\star}_{\mathrm{DR}}(z = \mathrm{f}, y|x) \tag{50}$$

$$= \sum_{y} \frac{\alpha(x)\pi_{\mathrm{f}}(y|x)\exp(r(x, y)/\beta)}{Z_{\rho}(x)} \tag{51}$$

$$= \frac{\alpha(x)\sum_{y}\pi_{\mathrm{f}}(y|x)\exp(r(x, y)/\beta)}{Z_{\rho}(x)} \tag{52}$$

$$= \frac{\alpha(x)Z_{\mathrm{f}}(x)}{\alpha(x)Z_{\mathrm{f}}(x) + \big(1-\alpha(x)\big)Z_{\mathrm{s}}(x)}. \tag{53}$$

The slow-mode probability follows by normalization:

$$P^{\star}(z = \mathrm{s}|x) = 1 - P^{\star}(z = \mathrm{f}|x). \tag{54}$$

This proves Corollary 1.

D.4 Proof of Proposition 2

The exact behavior-reference correction is

$$\kappa(x, y) = \frac{\rho(y|x)}{\mu(y|x)} = \frac{\alpha(x)\pi_{\mathrm{f}}(y|x) + \big(1-\alpha(x)\big)\pi_{\mathrm{s}}(y|x)}{q(x)\pi_{\mathrm{f}}(y|x) + \big(1-q(x)\big)\pi_{\mathrm{s}}(y|x)}. \tag{55}$$

Fast samples. For a fast sample y_{f}, define

$$\delta_{\mathrm{f}}(x, y_{\mathrm{f}}) = \frac{\pi_{\mathrm{s}}(y_{\mathrm{f}}|x)}{\pi_{\mathrm{f}}(y_{\mathrm{f}}|x)}. \tag{56}$$

Dividing the numerator and denominator of Eq. (55) by $\pi_{\mathrm{f}}(y_{\mathrm{f}}|x)$, we obtain

$$\kappa(x, y_{\mathrm{f}}) = \frac{\alpha(x) + \big(1-\alpha(x)\big)\delta_{\mathrm{f}}}{q(x) + \big(1-q(x)\big)\delta_{\mathrm{f}}}. \tag{57}$$

Subtracting the mode-wise constant $\alpha(x)/q(x)$ gives

$$\kappa(x, y_{\mathrm{f}}) - \frac{\alpha(x)}{q(x)} = \frac{\alpha(x) + \big(1-\alpha(x)\big)\delta_{\mathrm{f}}}{q(x) + \big(1-q(x)\big)\delta_{\mathrm{f}}} - \frac{\alpha(x)}{q(x)} \tag{58}$$

$$= \frac{\delta_{\mathrm{f}}\left[q(x) - \alpha(x)\right]}{q(x)\left[q(x) + \big(1-q(x)\big)\delta_{\mathrm{f}}\right]}. \tag{59}$$

By Assumption 1, $\delta_{\mathrm{f}} \le \varepsilon_{\mathrm{f}}$. If $q(x) \in [\eta, 1-\eta]$, the denominator in Eq. (59) is bounded away from zero. Therefore,

$$\kappa(x, y_{\mathrm{f}}) = \frac{\alpha(x)}{q(x)} + O(\varepsilon_{\mathrm{f}}). \tag{60}$$

Slow samples. For a slow sample y_{s}, define

$$\delta_{\mathrm{s}}(x, y_{\mathrm{s}}) = \frac{\pi_{\mathrm{f}}(y_{\mathrm{s}}|x)}{\pi_{\mathrm{s}}(y_{\mathrm{s}}|x)}. \tag{61}$$

Dividing the numerator and denominator of Eq. (55) by $\pi_{\mathrm{s}}(y_{\mathrm{s}}|x)$, we obtain

$$\kappa(x, y_{\mathrm{s}}) = \frac{1 - \alpha(x) + \alpha(x)\delta_{\mathrm{s}}}{1 - q(x) + q(x)\delta_{\mathrm{s}}}. \tag{62}$$

Subtracting the mode-wise constant $(1 - \alpha(x))/(1 - q(x))$ gives

$$\kappa(x, y_{\mathrm{s}}) - \frac{1 - \alpha(x)}{1 - q(x)} = \frac{1 - \alpha(x) + \alpha(x)\delta_{\mathrm{s}}}{1 - q(x) + q(x)\delta_{\mathrm{s}}} - \frac{1 - \alpha(x)}{1 - q(x)} \tag{63}$$

$$= \frac{\delta_{\mathrm{s}}\left[\alpha(x) - q(x)\right]}{\big(1 - q(x)\big)\left[1 - q(x) + q(x)\delta_{\mathrm{s}}\right]}. \tag{64}$$

By Assumption 1, $\delta_{\mathrm{s}} \le \varepsilon_{\mathrm{s}}$. If $q(x) \in [\eta, 1 - \eta]$, the denominator in Eq. (64) is bounded away from zero. Therefore,

$$\kappa(x, y_{\mathrm{s}}) = \frac{1 - \alpha(x)}{1 - q(x)} + O(\varepsilon_{\mathrm{s}}). \tag{65}$$

This proves Proposition 2.

E Empirical Validation of Dual-Reference Assumptions

E.1 Validation of Mode-Format Separation

The pointwise approximation to the marginal correction in Proposition 2 relies on the smallness of the cross-mode ratios

$$\delta_{\mathrm{f}}(x, y_{\mathrm{f}}) = \frac{\pi_{\mathrm{s}}(y_{\mathrm{f}}|x)}{\pi_{\mathrm{f}}(y_{\mathrm{f}}|x)}, \quad \delta_{\mathrm{s}}(x, y_{\mathrm{s}}) = \frac{\pi_{\mathrm{f}}(y_{\mathrm{s}}|x)}{\pi_{\mathrm{s}}(y_{\mathrm{s}}|x)}.$$

We validate this assumption with teacher-forced cross-likelihood evaluation. For each side, we sample 1000 completions and compute their log-likelihoods under both the fast policy π_{f} and the slow policy π_{s}. We run the evaluation on the main generation path and on an ablation path. The key diagnostic is whether the off-diagonal likelihoods are exponentially smaller than the diagonal likelihoods.

Main-path result. Figure 4 shows the main-path cross-likelihood heatmaps, and Table 5 reports the numerical values. Each cell reports mean per-token log-likelihood followed by mean sequence log-likelihood in nats.

Table 5: Cross-likelihood matrix on the main path. Each entry reports mean per-token log-likelihood / mean sequence log-likelihood in nats.

	y_{f} ($\bar{L} = 6.21$)	y_{s} ($\bar{L} = 276.64$)
π_{f}	$\mathbf{-0.181 / -1.192}$	$-1.434/\mathbf{-274.380}$
π_{s}	$-4.514/\mathbf{-19.465}$	$\mathbf{-0.190 / -66.764}$

The diagonal per-token log-likelihoods are close to zero, around -0.19, indicating that each policy assigns high confidence to its own response family. The off-diagonal sequence likelihoods are much smaller. For fast samples, the relative sequence log-likelihood ratio is

$$\log \frac{\pi_{\mathrm{s}}(y_{\mathrm{f}}|x)}{\pi_{\mathrm{f}}(y_{\mathrm{f}}|x)} = -19.465 - (-1.192) = -18.273. \tag{66}$$

For slow samples, the relative sequence log-likelihood ratio is

$$\log \frac{\pi_{\mathrm{f}}(y_{\mathrm{s}}|x)}{\pi_{\mathrm{s}}(y_{\mathrm{s}}|x)} = -274.380 - (-66.764) = -207.616. \tag{67}$$

Thus, the cross-mode ratios are on the order of e^{-18} for $\pi_{\mathrm{s}}(y_{\mathrm{f}}|x)/\pi_{\mathrm{f}}(y_{\mathrm{f}}|x)$ and e^{-208} for $\pi_{\mathrm{f}}(y_{\mathrm{s}}|x)/\pi_{\mathrm{s}}(y_{\mathrm{s}}|x)$. These values strongly support the approximate mode-format separation assumption used in Proposition 2.

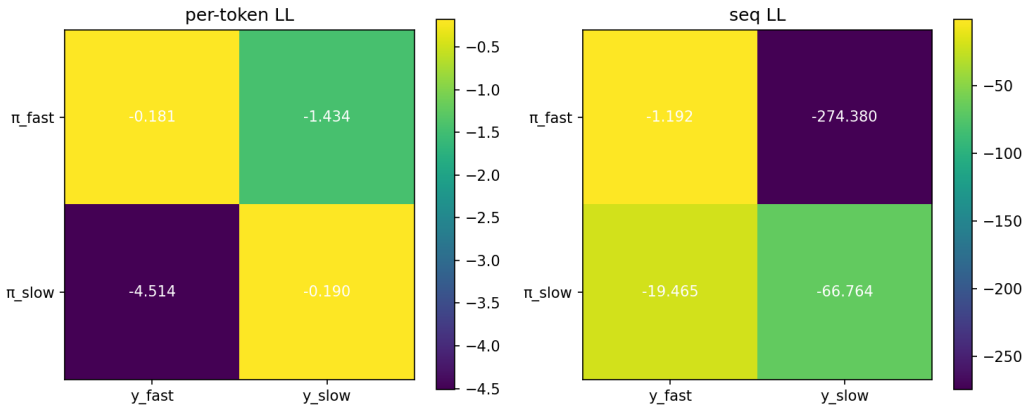


Figure 4: Cross-likelihood heatmaps on the main path. The left panel reports mean per-token log-likelihood; the right panel reports mean sequence log-likelihood. Diagonal entries correspond to in-mode likelihoods, while off-diagonal entries measure cross-mode likelihoods.

Two mechanisms behind approximate separation. The two off-diagonal directions are separated for different reasons. Table 6 summarizes the mechanisms.

Table 6: Different mechanisms behind the two cross-mode likelihood gaps.

Direction	Mechanism	Length dependence
$\pi_{\mathrm{f}}(y_{\mathrm{s}})$	**Length-compounded decay.** The fast policy assigns slightly lower probability to many reasoning tokens; over long traces this compounds into an exponentially small sequence ratio. Empirically, the per-token mismatch is amplified over $\bar{L} \approx 277$ tokens.	Strong: $r(L, \Delta_{\mathrm{seq}}) = -0.902$.
$\pi_{\mathrm{s}}(y_{\mathrm{f}})$	**Prefix bottleneck.** The slow policy tends to start with a reasoning-format prefix such as `<think>`; direct fast answers start with answer tokens instead. The mismatch is therefore concentrated near the first token rather than accumulated over length.	Weak: $r(L, \Delta_{\mathrm{seq}}) = +0.226$.

This asymmetry is important. Approximate separation is not merely a length effect. For $\pi_{\mathrm{f}}(y_{\mathrm{s}}|x)$, length amplifies the mismatch over long reasoning traces. For $\pi_{\mathrm{s}}(y_{\mathrm{f}}|x)$, the dominant cause is a prefix-level format bottleneck: after the assistant header, the slow policy strongly prefers a reasoning marker, while y_{f} begins directly with the answer. This supports our use of the weaker $\varepsilon_{\mathrm{f}}, \varepsilon_{\mathrm{s}}$-separation assumption instead of a symmetric length-dependent assumption.

Ablation path. We further evaluate an ablation path to check whether the slow-to-fast gap is merely a data-source artifact. Figure 5 shows the ablation-path heatmaps.

The sequence log-likelihood of $\pi_{\mathrm{s}}(y_{\mathrm{f}}|x)$ changes only slightly, from -19.465 on the main path to approximately -19.07 on the ablation path. This shows that the prefix bottleneck is a checkpoint-level structural feature and cannot be removed simply by changing the source of y_{f}. However, the ablation fast pool contains a small number of reasoning-style long answers: about 3%–5% of samples bypass the disable-thinking prefix and generate reasoning-style completions, with the longest reaching $L = 216$. These contaminated samples weaken the strict separation of the fast pool and motivate length- or marker-based filtering, such as removing fast samples with reasoning markers or length exceeding a threshold such as $L > 30$ generated tokens. Exact marginal correction is another fallback. These are suggested controls for contaminated candidates, rather than a demonstrated general robustness guarantee.

Conclusion. The cross-likelihood results support three conclusions. First, the approximate mode-format separation assumption holds strongly on the main path: diagonal probabilities are high, while

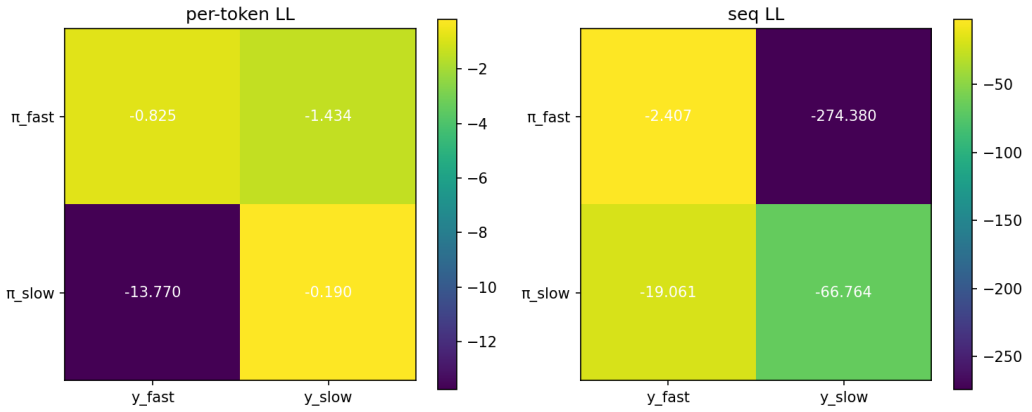


Figure 5: Cross-likelihood heatmaps on the ablation path. The slow policy still assigns low likelihood to direct fast answers, indicating that the prefix bottleneck is a structural property of the checkpoint. A small fraction of contaminated fast samples weaken the fast-pool purity.

off-diagonal sequence likelihood ratios fall in the e^{-18} to e^{-208} range. Second, the separation mechanisms are asymmetric and structural: fast-to-slow mismatch is length-compounded, while slow-to-fast mismatch is caused by a prefix bottleneck. Third, ablation-path failures are driven by contamination in the fast pool rather than by a failure of the dual-reference assumption itself.

E.2 Validation of Mode-wise Correction

We next validate whether the exact behavior-reference correction

$$\kappa_{\text{exact}}(x, y) = \frac{\rho(y|x)}{\mu(y|x)}$$

can be replaced by the mode-wise correction derived in Proposition 2. We reuse the log-likelihood files from Appendix E.1, join samples by their sample IDs, and obtain $n = 1958$ samples on the main path and $n = 1964$ samples on the ablation path. For each sample, we compare three quantities:

$$\log w_{\text{exact}}(x, y) = \log \frac{\alpha\pi_{\text{f}}(y|x) + (1-\alpha)\pi_{\text{s}}(y|x)}{q\pi_{\text{f}}(y|x) + (1-q)\pi_{\text{s}}(y|x)}, \tag{68}$$

$$\log w_{\text{theory}}(x, y_{\text{f}}) = \log \frac{\alpha}{q}, \tag{69}$$

$$\log w_{\text{theory}}(x, y_{\text{s}}) = \log \frac{1-\alpha}{1-q}. \tag{70}$$

As a diagnostic of single-reference failure, we also compute

$$\log w_{\text{single}}(x, y) = \log \pi_{\text{f}}(y|x) - \log \pi_{\text{s}}(y|x). \tag{71}$$

This quantity shows how severely a single reference can suppress samples from the other mode.

Balanced case: $\alpha = q = 0.5$. When $\alpha = q = 0.5$, the desired reference mixture and the empirical sampling mixture are identical:

$$\rho = \mu.$$

Therefore,

$$\kappa_{\text{exact}}(x, y) = 1, \quad \log w_{\text{exact}}(x, y) = 0.$$

The mode-wise correction also gives

$$\log w_{\text{theory}} = 0$$

for both fast and slow samples. Thus, all samples satisfy

$$\log w_{\text{theory}} = \log w_{\text{exact}} = 0.$$

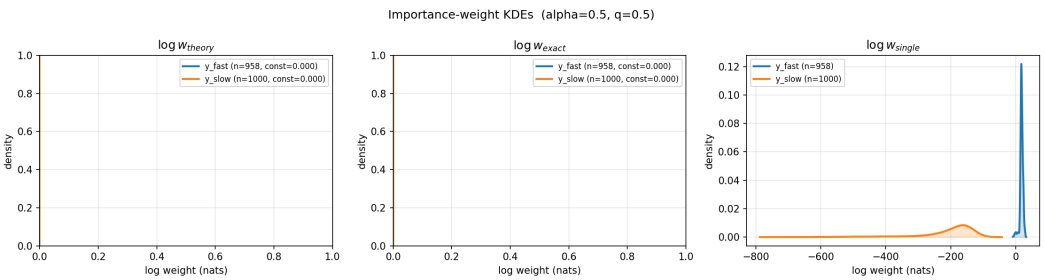


Figure 6: Correction weights in the balanced setting $\alpha = q = 0.5$. Both $\log w_{\text{theory}}$ and $\log w_{\text{exact}}$ collapse to zero, so correlation is undefined. The single-reference diagnostic $\log w_{\text{single}}$ still shows a strong bimodal structure.

This is a useful sanity check, but it cannot validate correlation because both variables have zero variance. Figure 6 shows this degenerate case.

Although w_{theory} and w_{exact} are degenerate in this setting, w_{single} still reveals the failure mode of single-reference weighting. Table 7 reports the statistics.

Table 7: Statistics of $\log w_{\text{single}} = \log \pi_{\text{f}}(y|x) - \log \pi_{\text{s}}(y|x)$ in the balanced setting.

Samples	Mean	Std	Min	Max
$\log w_{\text{single}} \mid y_{\text{f}}$ $(n = 958)$	$+18.27$	4.76	-7.96	$+29.49$
$\log w_{\text{single}} \mid y_{\text{s}}$ $(n = 1000)$	-207.62	91.96	-769.90	-101.92

The two modes are almost disjoint on the log scale, with a gap of roughly 100 nats between the fast and slow regions. For slow samples, 100% of ratios are below e^{-100}, and about 34% are below e^{-200}. Feeding such single-reference weights directly into training would underflow in common low-precision formats and would effectively remove the slow samples from the gradient. This directly explains the always-fast failure mode of single-reference BOLT with a fast reference.

Non-degenerate settings: $\alpha \neq q$. To test equivalence between w_{theory} and w_{exact}, we need $\alpha \neq q$. We evaluate two main-path settings, $(\alpha, q) = (0.5, 0.3)$ and $(0.5, 0.7)$, and one ablation-path setting, $(\alpha, q) = (0.5, 0.3)$. Figures 7, 8, and 9 show the corresponding KDE and scatter plots.

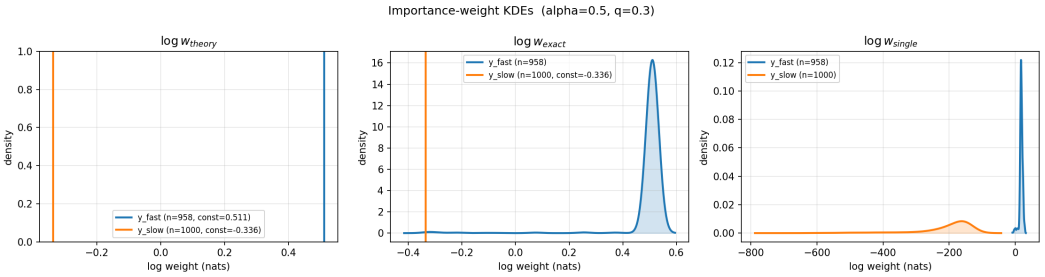


Figure 7: Correction weights on the main path with $\alpha = 0.5, q = 0.3$. Since $\alpha \neq q$, $\log w_{\text{theory}}$ is non-constant and can be compared with $\log w_{\text{exact}}$.

The mode gap between the fast and slow constants is 0.847 nats. Table 8 reports the quantitative comparison.

The main-path results show that the mode-wise correction is quantitatively close to the exact correction. The $q = 0.7$ setting reaches the target $r > 0.99$, while $q = 0.3$ is slightly below the threshold but still highly correlated. The ablation setting degrades more noticeably, driven by the heavy left tail of the fast margin caused by reasoning-style contaminated fast samples.

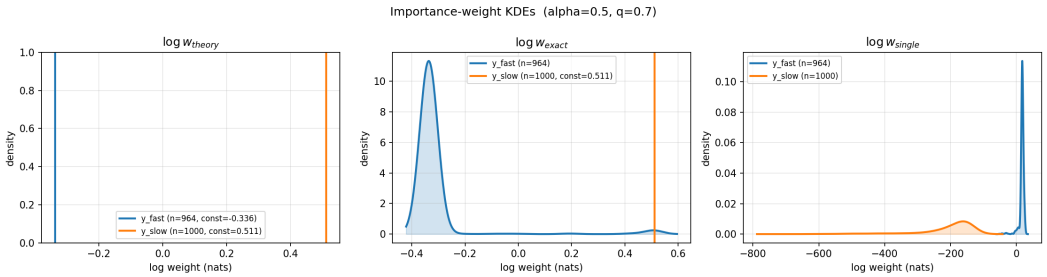


Figure 8: Correction weights on the main path with $\alpha = 0.5, q = 0.7$. The agreement between w_{theory} and w_{exact} is strongest in this setting.

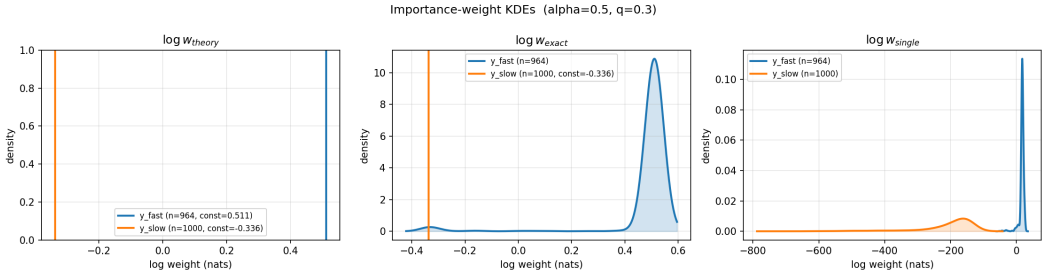


Figure 9: Correction weights on the ablation path with $\alpha = 0.5, q = 0.3$. The equivalence degrades because the ablation fast pool contains reasoning-style contaminated samples.

Key quantitative structure. The equivalence has three useful quantitative properties. First, the slow side is extremely stable: the median margin is about 177 nats, and the ratio $\pi_{\text{f}}(y_{\text{s}}|x)/\pi_{\text{s}}(y_{\text{s}}|x)$ underflows to zero even in high-precision arithmetic. As a result, w_{exact} and w_{theory} agree up to numerical precision on slow samples, with standard deviation around 7.85×10^{-15}. Second, the fast side is controlled by the margin: the error decays exponentially with the margin, and increasing the margin by 5 nats reduces the error by roughly 10^3. Third, the maximum possible deviation is bounded by the mode gap of 0.847 nats, because w_{exact} is a monotone function of the likelihood ratio and interpolates between the two mode constants α/q and $(1-\alpha)/(1-q)$.

Conclusion. These results support three conclusions. First, the mode-wise constant approximation is quantitatively valid on the main path, although the exact $r > 0.99$ criterion is sensitive to (α, q). Second, the single-reference diagnostic w_{single} exhibits an extreme dynamic range: slow samples are suppressed by more than 200 nats on average, which explains the numerical and optimization failure of single-reference BOLT. Third, equivalence degrades on the ablation path because the fast pool contains reasoning-style contaminated samples. This motivates cleaning the fast pool with reasoning-marker filters and length thresholds before using it as a strict dual-reference comparison.

E.3 Optimization Dynamics of Exact, Mode-wise, and Single-Reference Weights

The previous section validates the static equivalence between the exact correction and the mode-wise correction. We further test whether this equivalence persists during optimization. We compare four training buckets under $(\alpha, q) = (0.5, 0.7)$: **mode-wise**, which uses the theoretical mode-wise correction in Proposition 2; **exact**, which uses the exact correction $\kappa(x, y) = \rho(y|x)/\mu(y|x)$; **single**, which uses a degenerate single-reference weight; and **mode-wise-full**, which keeps the source-wise constants without group normalization and retains all all-wrong groups. All buckets share the same response bank, batch order, optimizer, and LoRA setup; the only difference is the sample-weight column. We train for 500 steps with AdamW and evaluate implicit KL by replaying 200 held-out samples every 25 steps using teacher forcing.

Table 8: Equivalence between exact correction and mode-wise correction. The main path shows strong agreement, while the ablation path degrades due to contaminated fast samples.

Metric	main, $q = 0.3$	main, $q = 0.7$	ablation, $q = 0.3$
Pooled $r(w_{\text{theory}}, w_{\text{exact}})$	0.9877	**0.9902**	0.9730
$\lvert\text{err}\rvert > 0.01$	1.5%	1.2%	2.2%
Max $\lvert\text{err}\rvert$	0.847	0.847	0.847
Fast margin std	4.76	4.76	**7.87**
Fast margin min	-7.96	-7.96	$\mathbf{-66.38}$

Main-path dynamics. Figure 10 shows the implicit KL trajectory on the main path. Figures 11 shows the training loss and gradient norm. Table 9 summarizes the key statistics.

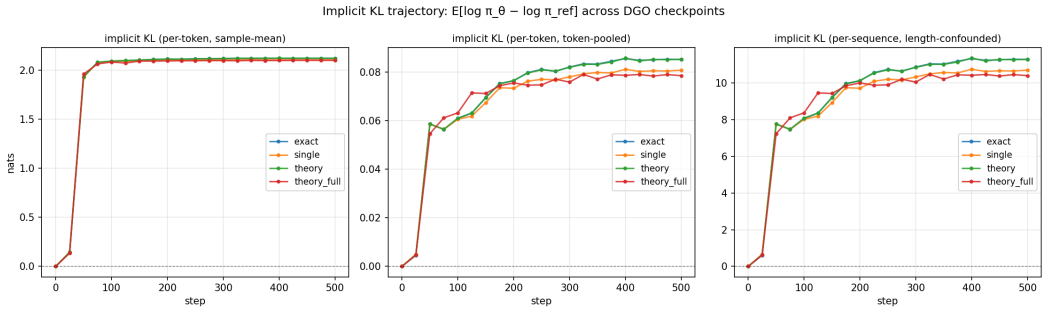


Figure 10: Implicit KL trajectory on the main path under $(\alpha, q) = (0.5, 0.7)$. The mode-wise and exact buckets remain nearly indistinguishable throughout training.

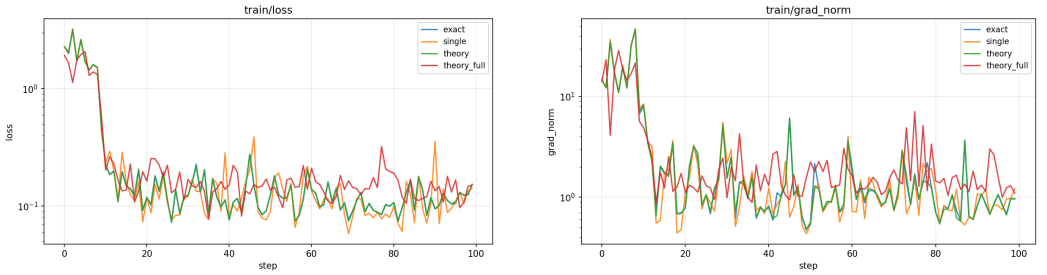


Figure 11: Training loss and gradient norm on the main path. The mode-wise and exact corrections have nearly identical optimization dynamics, while the full mode-wise implementation yields a smoother gradient profile.

The mode-wise and exact buckets are almost identical: their step-wise KL difference stays below 0.1 nats, their gradient-norm AUC differs by less than 1%, and their final losses differ by only 0.001. This shows that the static equivalence observed in Appendix E.2 does not accumulate into a meaningful optimization gap over 500 steps. In contrast, the single-reference bucket obtains a slightly lower final loss and lower final KL, but this is a misleading signal: the single-reference weight underflows for slow samples and effectively drops part of the slow-response family from the gradient. In our run, $28/930$ slow rows underflow to zero in bf16, and the remaining slow rows are also assigned extremely small weights. Thus, the single-reference dynamics are not better optimization, but rather a collapse toward a fast-path SFT-like objective.

The mode-wise-full bucket gives the smoothest trajectory, with the gradient-norm peak reduced from roughly 47 to 28.8. This implementation is closer to the mathematical source-wise correction, but it uses a larger effective dataset and covers fewer epochs under the same 500-step budget. We therefore treat it as an informative stability check rather than the main apples-to-apples comparison.

Table 9: Optimization dynamics on the main path. KL values are sequence-level implicit KL in nats.

Bucket	Step 25	Step 50	Step 100	Step 500	Final Loss	Grad Peak
Mode-wise	0.64	7.75	8.08	**11.28**	0.154	47.0
Exact	0.61	7.77	8.06	**11.27**	0.153	46.5
Single-ref	0.67	7.76	8.02	10.69	0.142	46.7
Mode-wise-full	0.62	7.23	8.36	10.39	0.151	**28.8**

Ablation-path dynamics. We repeat the same comparison on the ablation path. Figure 12 shows the implicit KL trajectory.

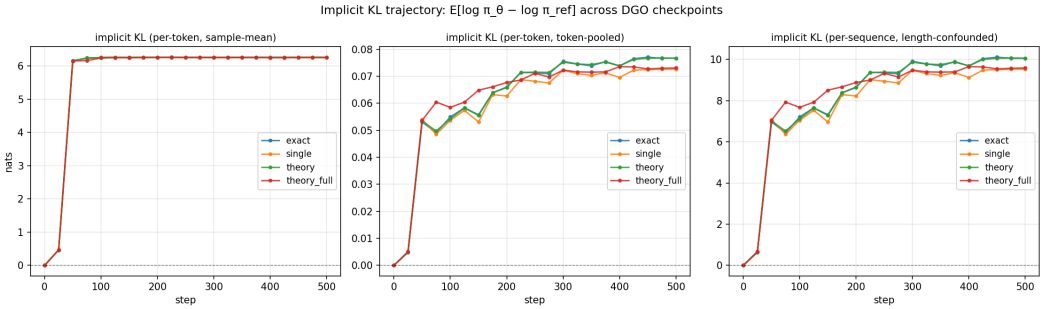


Figure 12: Implicit KL trajectory on the ablation path. The relative differences between buckets remain similar to the main path, but the absolute optimization scale is much larger due to first-token mismatch.

The bucket-relative behavior is similar to the main path, but the absolute scale is much larger. For the mode-wise bucket, KL per token at step 500 increases from 2.121 on the main path to 6.261 on the ablation path; the initial training loss increases from 2.28 to 6.45; and the peak gradient norm increases from 47.0 to 128.4. This is caused by a stronger base-data mismatch: the base model is a thinking checkpoint, while ablation-path fast samples are extremely short and require the model to move the first output token away from the reasoning prefix. This directly exposes the first-token bottleneck during optimization.

Takeaways. These dynamics support three conclusions. First, mode-wise and exact correction show closely matched static weights and optimization dynamics in this diagnostic; this is empirical agreement rather than exact finite-sample equivalence. Second, the single-reference objective fails by numerical underflow and gradient removal for one response family, explaining the mode-collapse behavior observed in the main experiments. Third, the absolute optimization difficulty is driven by base-data alignment, especially the first-token reasoning-prefix bottleneck, while the relative bucket differences are driven by the choice of sample weights.

F Experimental Details

F.1 Datasets and Evaluation Protocol

We evaluate DR-BOLT in one in-distribution (IID) setting and three held-out cross-benchmark transfer settings. For IID evaluation, we train on the ScienceQA training split and evaluate on the official ScienceQA test split [15]. For cross-benchmark transfer evaluation, we train on ViRL39K [25] and evaluate on MathVision [27], DynaMath [41], and MMMU-Pro [39]. All reported scores use the official metric when available and are normalized to $[0, 1]$. Table 10 summarizes the evaluation protocol.

Table 10: Datasets and evaluation protocol.

Dataset	Setting	Metric
ScienceQA [15]	IID: train on ScienceQA, test on ScienceQA	Accuracy
MathVision [27]	Transfer: train on ViRL39K, test on MathVision	Accuracy
DynaMath [41]	Transfer: train on ViRL39K, test on DynaMath	Accuracy
MMMU-Pro [39]	Transfer: train on ViRL39K, test on MMMU-Pro	Accuracy

F.2 Base Model, Prompts, and Baselines

All internal methods use the same Qwen3-VL base family and the same evaluation prompts. The fast reference policy is the Qwen3-VL direct-answer / instruct policy, and the slow reference policy is the Qwen3-VL thinking policy. The trainable policy is initialized from the same Qwen3-VL checkpoint family for all internal training methods. We use identical input prompts, image preprocessing, answer extraction, and evaluation scripts across Qwen3-Instruct, Qwen3-Thinking, Mix SFT, Reward SFT, single-reference BOLT, and DR-BOLT. Thus, matched trainable internal comparisons help isolate the training objective from prompt, model-family, and teacher-bank changes. The frozen fast/slow reference rows provide context rather than an objective-only comparison.

External baselines. We reproduce the external adaptive-thinking methods under the common data and evaluation protocol; reported score/length values are from our runs rather than copied from their papers. Appendix H.5 records the implementation provenance. DualMindVLM learns controllable fast and slow thinking modes with mode supervision and GRPO [13]. FAST-GRPO uses difficulty-aware rewards and adaptive length regularization [31]. TON trains VLMs to decide whether to reason using thought dropout and reinforcement learning [26]. We also include a reproduced GRPO baseline under our evaluation setup. Because external methods may differ in base model and training implementation, they are treated as reference comparisons; the most controlled comparisons are the matched internal baselines.

Internal baselines. **Qwen3-Instruct** is the fast direct-answer baseline. **Qwen3-Thinking** is the slow reasoning baseline. **Mix SFT** trains on the union of fast and slow responses with uniform weights. **Reward SFT** uses reward-dependent weights without BOLT normalization or behavior-reference correction. **Single-ref Fast** applies single-reference BOLT using the fast reference. **Single-ref Slow** applies single-reference BOLT using the slow reference. **DR-BOLT** uses the dual-reference objective with mode-wise correction.

F.3 Response Bank Construction

For each input, we build an offline response bank containing fast and slow candidates. Unless otherwise specified, we use $N_{\mathrm{f}} = 7$ fast candidates and $N_{\mathrm{s}} = 3$ slow candidates per input, corresponding to an empirical fast sampling ratio $q = 0.7$. The desired reference prior is fixed to $\alpha = 0.5$. Thus, the main experiments use a non-balanced response bank, which makes the behavior-reference correction active:

$$\kappa(y_{\mathrm{f}}) = \frac{\alpha}{q} = \frac{0.5}{0.7} = 0.714, \quad \kappa(y_{\mathrm{s}}) = \frac{1-\alpha}{1-q} = \frac{0.5}{0.3} = 1.667.$$

For exact correction experiments, we compute

$$\kappa_{\mathrm{exact}}(x, y) = \frac{\alpha\pi_{\mathrm{f}}(y|x) + (1-\alpha)\pi_{\mathrm{s}}(y|x)}{q\pi_{\mathrm{f}}(y|x) + (1-q)\pi_{\mathrm{s}}(y|x)}$$

in log space using teacher-forced likelihoods under both references. For the default DR-BOLT run, we retain the known generation source and use the mode-wise correction in Proposition 2.

F.4 Reward, Decoding, and Mode Detection

The reward is

$$r(x, y) = r_{\mathrm{acc}}(x, y) + \gamma r_{\mathrm{fmt}}(y) - \lambda\ell(y).$$

Table 11: Reward and correction hyperparameters used in the main experiments.

Hyperparameter	Value / Definition
Target fast prior α	0.5
Empirical fast sampling ratio q	0.7
Fast candidates N_{f}	7 per input
Slow candidates N_{s}	3 per input
BOLT temperature β	1.0
Format reward weight γ	0.1
Length penalty λ	0.05
Length cost $\ell(y)$	word length normalized by the maximum response length in the candidate group
Mode-wise $\kappa(y_{\mathrm{f}})$	0.714
Mode-wise $\kappa(y_{\mathrm{s}})$	1.667

Here r_{acc} is the task correctness reward, r_{fmt} checks valid answer or reasoning format, and $\ell(y)$ is the normalized response length. We use the same reward implementation for all BOLT-style methods. Table 11 summarizes the reward and correction hyperparameters.

For response generation and routing calibration, we use temperature $T = 0.7$. For repeated auto-mode decoding in Figure 3, each prompt is sampled 20 times. A generated response is classified as slow if it contains the reasoning marker `<think>` or follows the slow-thinking format; otherwise it is classified as fast. In the mixed benchmark tables, the slow-ratio row aggregates responses only across MathVision, DynaMath, and MMMU-Pro from the ViRL39K-trained checkpoint. The ScienceQA-trained checkpoint is separate. The 100-prompt repeated-decoding subset used for calibration and sensitivity is a distinct measurement population. For benchmark scoring, final answers are extracted with the same answer parser for all internal methods.

F.5 Optimization Details

All trainable models are optimized with PyTorch using weighted next-token likelihood. We use LoRA fine-tuning with AdamW. The optimization objective is the weighted SFT loss defined in Eq. (21). We mask prompt tokens in the label sequence so that loss is computed only on response tokens. The same optimizer, response bank, reward values, and data order are used across internal baselines whenever applicable. Table 12 summarizes the training configuration.

Table 12: Training configuration for reproduced internal experiments.

Item	Value
Framework	PyTorch
Fine-tuning method	LoRA weighted SFT
Optimizer	AdamW
Training steps	500 for white-box optimization diagnostics; full benchmark runs use the same budget across matched baselines
Per-device batch size	1
Gradient accumulation	8
Number of GPUs	4
Effective batch size	32 sequences
Prompt loss masking	Prompt tokens are masked with label value -100
Phase 0 warm-up	100 optimizer steps, unweighted 1:1 fast/slow mixture

F.6 Length Measurement

The main paper reports response *Length* measured by word count. We use word count rather than tokenizer-specific token count because compared systems may use different tokenizers or model variants. Length is computed on generated responses only, excluding the input prompt. Reasoning markers such as `<think>` and `</think>` are included because they appear in the generated output and contribute to inference cost. Tokenizer-specific token counts should only be compared within the same tokenizer and are not used as the main cross-model metric.

F.7 Compute and Resource Measurement

All reproduced resource measurements are conducted on a machine with 4 NVIDIA A100 GPUs with 80GB memory each and 32 CPU cores. All implementations are based on PyTorch. We report wall-clock training time in hours, peak GPU memory in GB, and speedup relative to the reproduced GRPO baseline. The reported time includes candidate generation, reward computation, weight

construction, and policy optimization. For GRPO-style baselines, the time includes online rollout and policy update. For DR-BOLT, the time includes offline candidate generation and weighted supervised optimization. Peak memory is measured as the maximum observed per-GPU memory during training. Speedup is computed as

$$\text{Speedup}(\text{method}) = \frac{\text{Time}(\text{GRPO})}{\text{Time}(\text{method})}.$$

Thus, values larger than 1 indicate faster training. In our reproduced setup, GRPO takes 202.4 hours with 70.2GB peak memory, while DR-BOLT takes 69.7 hours with 60.2GB peak memory, corresponding to a $2.90\times$ speedup and lower memory usage. The resource advantage comes from decoupling response generation from policy optimization, whereas GRPO-style methods couple online rollout and update.

G Additional Ablations

G.1 Downstream Ablation of Behavior-Reference Correction

The white-box analyses in Appendix E show that the mode-wise correction closely approximates the exact behavior-reference correction. We further test whether this approximation preserves downstream task performance. This ablation must use $\alpha \neq q$; otherwise $\rho = \mu$ and $\kappa(x, y) = 1$, making correction variants indistinguishable. We therefore set $\alpha = 0.5, q = 0.7$, where fast samples are over-sampled and slow samples require up-weighting. All variants use the same ViRL39K response bank, reward, optimizer, Phase 0 warm-up, and training budget.

Table 13: Downstream ablation of behavior-reference correction under $\alpha = 0.5, q = 0.7$. Each benchmark cell reports Score/Length. Slow Ratio is reported in percentage points.

Correction	MathVision	DynaMath	MMMU-Pro	Slow Ratio
No κ	$0.260/518.2$	$0.626/308.1$	$0.314/329.9$	0.13
Exact κ	$0.623/1023.2$	$0.804/325.7$	$0.594/382.0$	22.0
Mode-wise κ	$0.606/1058.1$	$0.799/370.6$	$0.581/385.7$	24.1

The results show that behavior-reference correction is necessary in the non-balanced setting. Without κ, the model is strongly biased toward the over-sampled fast mode: the slow ratio drops to 0.13%, and transfer scores fall sharply, especially on MathVision and MMMU-Pro. Exact correction restores the contribution of slow samples and gives the strongest scores. Mode-wise correction closely matches exact correction: the score gaps are 0.017, 0.005, and 0.013 on MathVision, DynaMath, and MMMU-Pro, respectively. The response Lengths and slow ratios are also close, showing that the practical mode-wise approximation preserves the downstream behavior of exact correction while avoiding expensive dual-reference likelihood evaluation.

H Controlled Offline Baselines and Further Experimental Details

H.1 Comparison with Alternative Offline Objectives

We compare DR-BOLT with multi-reference DPO (MR-DPO) and mode-conditioned reward-weighted SFT (Cond-RW). These methods use the same fast/slow response bank, scalar reward and length penalty, frozen references, Qwen3-VL-8B-Thinking initialization, and Phase-0 warm-up. Optimization and response-token budgets are matched. ScienceQA results use a ScienceQA-trained checkpoint; MathVision, DynaMath, and MMMU-Pro results use a ViRL39K-trained checkpoint. The cross-benchmark slow ratio aggregates the latter three evaluations only.

Multi-reference DPO. For each input, we construct preferences from strictly reward-ordered pairs of bank candidates, discard reward ties, and subsample pairs to match the response-token training budget. The reference is the exact mixture

$$\rho(y|x) = \alpha\pi_{\text{f}}(y|x) + (1-\alpha)\pi_{\text{s}}(y|x),$$

Table 14: One-factor ScienceQA sensitivity study. Length is mean response words. Slow ratio is measured on the 100-prompt repeated-decoding calibration subset used for Figure 3, not the full ScienceQA test set.

Change from default	Score	Length	Calibration slow ratio (%)
Default	0.967	20.4	51.4
$\alpha = 0.3$	0.968	25.5	71.1
$\alpha = 0.7$	0.964	15.2	31.2
$q = 0.3$	0.965	21.4	53.8
$q = 0.5$	0.966	20.8	52.2
$\beta = 0.5$	0.966	17.8	45.1
$\beta = 2.0$	0.963	23.7	57.8
$\lambda = 0$	0.968	28.9	59.2
$\lambda = 0.025$	0.969	24.6	55.4
$\lambda = 0.10$	0.964	15.7	43.8

computed in log space from the two frozen reference likelihoods. The loss is

$$\mathcal{L}_{\text{MR-DPO}} = -\mathbb{E}_{(x,y^+,y^-)} \log \sigma\left(\beta_{\text{DPO}} \left[\log \frac{\pi_\theta(y^+|x)}{\rho(y^+|x)} - \log \frac{\pi_\theta(y^-|x)}{\rho(y^-|x)}\right]\right), \tag{72}$$

with $\beta_{\text{DPO}} = 0.1$. Each optimizer step consumes an effective batch of 32 response sequences over four GPUs, with the same seed, LoRA configuration, and optimizer settings as DR-BOLT. This baseline tests mixture-reference preference optimization while converting scalar rewards to pairwise orderings.

Mode-conditioned reward-weighted SFT. Each candidate is prepended with its known source token $z \in \{\texttt{<FAST>}, \texttt{<SLOW>}\}$. With $\mathcal{Y}(x)$ denoting the candidate bank, this baseline uses

$$\begin{aligned} w_{\text{Cond-RW}}(x,y) &= \frac{\exp(r(x,y)/\beta)}{\sum_{y' \in \mathcal{Y}(x)} \exp(r(x,y')/\beta)}, \\ \mathcal{L}_{\text{Cond-RW}} &= -\sum_{(x,z,y)} w_{\text{Cond-RW}}(x,y) \log \pi_\theta(z,y|x). \end{aligned} \tag{73}$$

At evaluation time, no mode token is supplied externally: the model first generates the token and then the response. Thus the comparison uses one adaptive generation model without oracle mode information. Unlike DR-BOLT, it has no behavior-reference correction.

MR-DPO is the stronger alternative in Table 3, but DR-BOLT obtains higher scores and shorter responses on all four benchmarks in this setting. The comparison controls for the stronger slow teacher and candidate coverage. It supports the observed benefit of the full objective for this fixed scalar-reward bank; it does not establish universal superiority over pairwise preference learning, or isolate the causal contribution of each objective component individually.

H.2 Hyperparameter Sensitivity

We vary one factor at a time around $\alpha = 0.5, q = 0.7, \beta = 1.0, \lambda = 0.05$, keeping the remaining reward, model, decoding, and optimization settings fixed. The target prior α expresses a desired fast/slow preference, q controls candidate collection, β controls reward-tilting sharpness, and λ controls the accuracy–length preference. For the q sweep, equal-size ten-candidate banks are drawn from a common fast/slow candidate superset, so only their mode proportions change.

The prior α changes slow usage monotonically: $\alpha = 0.3$ and 0.7 yield calibration slow ratios of 71.1% and 31.2%. After correction, changing q from 0.7 to 0.3 changes score by 0.002 and calibration slow ratio by 2.4 percentage points. This is consistent with removing the systematic preference introduced by candidate collection, while finite-bank variance can remain. The λ sweep produces shorter responses as cost preference increases; the reported scores lie in $[0.964, 0.969]$. Slow labels use the generated reasoning format, however short the trace, so a high slow-format fraction can coexist with a short mean response length.

Table 15: Shared length-penalty sweep after ViRL39K training. Cells report score / mean response words; slow ratio aggregates MathVision, DynaMath, and MMMU-Pro.

λ	MathVision	MMMU-Pro	Cross-benchmark slow ratio (%)
0	0.609/1396.3	0.578/512.8	31.8
0.025	0.614/1208.6	0.585/445.2	27.7
0.05	0.606/1058.1	0.581/385.7	24.1
0.10	0.589/811.4	0.561/302.7	17.4

Table 16: Router controls. ScienceQA and transfer results use separately trained systems. Cells report score / mean response words; slow ratio aggregates the three transfer benchmarks.

Method	ScienceQA	MathVision	DynaMath	MMMU-Pro	Slow ratio (%)
Two-policy router	0.958/64.7	0.401/2018.5	0.681/827.8	0.416/1054.6	28.9
Thinking prompt router	0.953/252.4	0.366/5108.7	0.652/2396.4	0.375/2748.9	87.6
DR-BOLT	0.967/20.4	0.606/1058.1	0.799/370.6	0.581/385.7	24.1

We also train ViRL39K checkpoints for the same λ sweep and evaluate each checkpoint on all three transfer benchmarks without selecting λ separately on their test sets. Table 15 reports MathVision and MMMU-Pro; its slow ratio aggregates all three benchmarks.

Both benchmarks become shorter as λ increases, with accuracy degrading under stronger compression. Their highest reported scores in this sweep occur at $\lambda = 0.025$; the shared main value 0.05 gives shorter responses with scores of 0.606 and 0.581. Thus the results characterize a utility trade-off rather than an accuracy-maximizing choice of λ. Matched reward-based internal methods and both added offline baselines share the same reward and penalty. External adaptive-thinking reproductions preserve method-specific reward formulations and are therefore method-level comparisons rather than strictly λ-controlled ones.

H.3 Learned Routers and Deployment Controls

The two-policy learned router retains the frozen Qwen3-VL-8B-Instruct and Qwen3-VL-8B-Thinking experts. For each training input, a lightweight classifier on frozen multimodal representations predicts

$$z^*(x) = \arg\max_{z \in \{\mathrm{f},\mathrm{s}\}} U_z(x), \quad U_{\mathrm{f}}(x) = \log \alpha + \log \widehat{Z}_{\mathrm{f}}(x), \quad U_{\mathrm{s}}(x) = \log(1-\alpha) + \log \widehat{Z}_{\mathrm{s}}(x).$$

The classifier chooses one expert at inference. No evaluation-set correctness labels or rewards are used to train or operate it. A separate one-checkpoint prompt-router diagnostic uses the same classifier to choose direct-answer or reasoning prompts for the unadapted thinking checkpoint. It tests whether selecting a prompt is sufficient to realize both behaviors within an existing model.

For transfer evaluation, the router classifier is trained on ViRL39K and the expert checkpoints remain frozen; DR-BOLT trains the generation policy itself on ViRL39K. The prompt router has 18.4% fast-prompt compliance, defined as the fraction of inputs routed to the direct-answer prompt producing neither a reasoning marker nor more than 30 generated tokens. Its 87.6% slow-format rate shows that selecting a direct-answer prompt does not reliably induce fast behavior in the thinking checkpoint.

The two-policy router needs two generation checkpoints and a separate classifier, whereas DR-BOLT needs one generation policy and no external router. This is a deployment distinction; we do not infer an exact serving-memory multiplier or latency advantage without measuring the loading strategy. The router comparison also changes whether generation policies are frozen or fine-tuned, so it does not isolate routing removal alone.

H.4 Deterministic Reward Extraction

The training reward is $r = r_{\mathrm{acc}} + 0.1 r_{\mathrm{fmt}} - 0.05\ell(y)$, where correctness and format validity are binary and $\ell(y)$ is response word count divided by the longest response word count in the same input's candidate group. Correctness uses the extracted final answer; intermediate reasoning is not graded. The format term checks for a parseable final answer in the required format rather than semantic correctness or plausible reasoning. Parse failures receive zero correctness and zero format reward.

Table 17: Representative parsing cases from generated responses.

Dataset	Answer span	Parsing outcome
ScienceQA	`B. New Hampshire`	Leading-option rule; B matches ground truth B.
ScienceQA	`B` after a 168-word slow response	Reasoning is discarded; single-letter rule yields B.
ScienceQA	`Neither.`	No option is recovered; zero correctness and format reward.
MathVision	Unboxed arithmetic-sequence derivation	Last numeric literal is 60; matches ground truth 60.
MathVision	`A` for a multiple-choice item	Letter path yields A; matches the correct option.
MMMU-Pro	`E, H`	Multiple options; no unique answer, so zero correctness and format reward.

A format bonus of at most 0.1 and length penalty of at most 0.05 cannot make an incorrect answer outrank a correctly parsed correct answer solely through formatting. All internal methods share the same parsers and ground-truth annotations; no LLM judge is used for rewards or evaluation.

Answer span and option extraction. The parser uses the text after the last `</think>` marker, or the response when there is no closing marker. ScienceQA and the multiple-choice MMMU-Pro protocol upper-case the extracted letter, require it to fall within the item's valid option range, and compare it exactly with the ground-truth option letter. MMMU-Pro uses the item's supplied options, normally A–J in the ten-option setting. ScienceQA ground-truth letters are derived from the official answer index; answer-text matching is not used. The disclosed letter-extraction precedence is:

1. the last `\boxed{...}` whose first character is a valid option letter;
2. a leading `A.` or `(A)` pattern;
3. an `answer is`, `answer:`, `answer =`, or `answer should be` cue followed by an optionally bolded letter;
4. a leading `**A**`;
5. a span consisting of exactly one letter;
6. a leading letter followed by whitespace;
7. any line reducing to a single letter after stripping surrounding periods and parentheses.

Unrecoverable or out-of-range letters are incorrect. The parser also rejects a response naming multiple options, such as `E, H`, as a parse failure.

MathVision and DynaMath normalization. For MathVision items with options and a single-letter ground truth, the letter path uses the item's option range; the disclosed normalization additionally accepts the ground-truth option surface text as a case-insensitive substring. Otherwise, the numeric path uses the last boxed span when present, or the answer span, and takes its last numeric literal. The disclosed numeric pattern is

```
-?\d+(\.\d+)?([eE][+-]?\d+)?
```

A parsed numeric value v matches ground truth g when $|v - g| \leq 10^{-3} \max(1, |g|)$. For non-numeric ground truth, the disclosed fallback uses case-insensitive substring containment. DynaMath uses the same numeric extraction and tolerance, or substring matching for short non-numeric answers. Unparseable responses are incorrect. These are deterministic normalization rules and should not be interpreted as a general symbolic-equivalence verifier.

The implementation uses the MathVision evaluator's extraction and equality utilities, the DynaMath scoring routine (an accuracy sub-score above 0.5 is treated as correct), and the MMMU multiple-choice parser for MMMU-Pro.

Table 18: Model snapshot revisions.

Role	Checkpoint and revision
Fast reference	`Qwen/Qwen3-VL-8B-Instruct` `0c351dd01ed87e9c1b53cbc748cba10e6187ff3b`
Slow reference and initialization	`Qwen/Qwen3-VL-8B-Thinking` `92f3c4b4feadd3a016ef468d103bb5f58b2a2c6b`
FAST setup verification	`xiaowenyi/FAST-8B-Preview` `45aa28a69976f14f295f34d4bde25099a84cf1e5`

Table 19: Additional shared optimization and decoding configuration.

Item	Value
LoRA	Rank 8, scaling factor 64, dropout 0.05, no bias.
Target modules	`q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, `down_proj`.
AdamW	Learning rate 5×10^{-6}, weight decay 0.1, moments $(0.9, 0.95)$.
Schedule	Cosine decay with warm-up ratio 0.1.
Batch	Four GPUs, per-device batch 1, gradient accumulation 8; effective batch 32 response sequences.
Precision / attention	bf16 / SDPA; key-value cache disabled during training.
Masking	Prompt tokens and assistant header have label -100.
Checkpoint cadence	Every 25 optimizer steps for implicit-KL replay.
Decoding	Temperature 0.7, top-p 0.9; maximum new tokens 2048 for ScienceQA and 8192 for ViRL39K, MathVision, DynaMath, and MMMU-Pro.
Diagnostics	500 steps; 200 held-out samples replayed every 25 steps.

H.5 Pinned Checkpoints, Optimization, and Cost Provenance

The exact checkpoints and snapshot revisions used for internal runs are listed in Table 18. The processor and tokenizer use the corresponding model snapshot. The FAST preview checkpoint is used to verify the Qwen3-VL-8B setup; the dual-reference candidates come from the Instruct and Thinking checkpoints.

Phase 0 is 100 optimizer steps of unweighted SFT on a 1:1 fast/slow mixture. It uses the same LoRA modules, AdamW optimizer, learning rate, and batch configuration as Phase 2, without reward or behavior-reference weights. The shared configuration is summarized in Table 19.

The random seed for LoRA initialization and data order is shared across matched buckets, and the full benchmark optimization budget is matched. The 500-step budget refers to the white-box diagnostic experiment.

All reported external score/length entries are our common-data reproductions, rather than scores copied from their original papers. FAST-GRPO follows its released recipe. TON retains its thought-dropout SFT plus GRPO procedure with ViRL39K converted to its expected multimodal schema. DualMindVLM is reimplemented from its two-stage mode-labeling and GRPO specification. Their cross-benchmark training data are ViRL39K and their evaluation inputs and answer parser are shared. The matched internal suite additionally controls references, candidates, reward values, initialization, optimizer, and data order wherever applicable.

The $2.90\times$ speedup is $202.4/69.7$ and therefore includes offline response generation. Peak per-GPU memory is 60.2GB for DR-BOLT and 70.2GB for GRPO. The response bank, references, reward values, and weights are frozen before Phase-2 optimization, so no new on-policy rollouts or reference-likelihood recomputation are needed after each update.

H.6 Candidate Coverage and Scope of the Evidence

The union of seven fast and three slow candidates on ViRL39K has reported oracle coverage

$$A_{\mathrm{bank}} = \Pr[\exists y \in \mathcal{Y}_{\mathrm{f}}(x) \cup \mathcal{Y}_{\mathrm{s}}(x) : r_{\mathrm{acc}}(x, y) = 1] \approx 0.947.$$

This is the fraction of training inputs with at least one correct candidate, not a deployed model score or teacher pass@1 accuracy. Reward tilting can exploit complementary candidates, so either

Table 20: Runtime provenance on four A100-80GB GPUs. Times are wall-clock hours from our reproduced runs.

Method	Time (h)	Included stages
GRPO	202.4	Online rollout generation and policy updates.
FAST-GRPO	176.6	Online rollout generation and policy updates under the released FAST recipe.
TON	248.9	Thought-dropout SFT, online rollouts, and policy updates.
DualMindVLM	–	Runtime is not reported.
DR-BOLT	69.7	Fast/slow candidate generation, correctness and format rewards, weight construction, and policy optimization.

Table 21: 3DSRBench results on the constructed held-out test partition. Length is mean response words.

Method	Score	Length
Fast reference	0.531	13.9
Slow reference	0.548	1846.1
DR-BOLT	0.546	452.2

reference's single-decoding accuracy is not a strict ceiling on the learned policy. However, the finite bank supplies reward evidence only for sampled candidates and does not provide unconstrained online exploration.

H.7 Additional Same-Benchmark Evaluation on 3DSRBench

We additionally evaluate on 3DSRBench [16], which covers height, location, orientation, viewpoint, and multi-object 3D spatial reasoning. Its real, synthetic-common, and synthetic-uncommon subsets are evaluation groups rather than a supervised train/test split. We therefore construct a disjoint training/test partition. The response bank and DR-BOLT training use only the training partition; all three methods in Table 21 are evaluated on the identical held-out test partition. No test examples or test rewards enter candidate generation or training. This is a same-benchmark experiment, separate from ViRL39K transfer evaluation.

DR-BOLT retains nearly all of the slow reference's accuracy, with a difference of 0.002, while reducing mean length by 75.5% ($4.08\times$). It improves on the fast reference by 0.015. This illustrates that adaptive reasoning can retain most of the correctness benefit of slow reasoning while removing unnecessary length; the intended utility trade-off does not require exceeding the slow reference in raw accuracy.

I Qualitative Examples

We provide qualitative examples to illustrate how DR-BOLT allocates fast and slow thinking across different inputs. Case 1 is a ViRL39K training example comparing fast and slow teacher responses. Cases 2–4 are held-out transfer examples from MathVision, DynaMath, and MMMU-Pro. The examples show that DR-BOLT uses reasoning when visual pattern or chart interpretation is needed, while producing direct answers for visually obvious symbolic recognition questions.

Case 1: ViRL39K Training Example



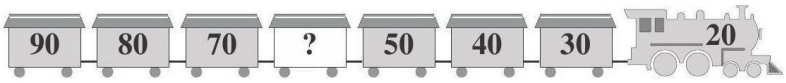
Question: The clocks show two things Lily did Tuesday evening. Which did Lily do earlier? Choices: (A) ate dinner; (B) watched a movie. Answer the question directly in one \boxed{}.

Ground Truth: $\boxed{B}$

Fast teacher output *(169 words):* The fast teacher reads the clocks as 7:00 for "watched a movie" and 9:00 for "ate dinner", compares the two times, and correctly answers $\boxed{B}$.

Slow teacher output *(278 words):* The slow teacher gives the same correct answer, but produces a longer reasoning trace that repeats the clock-reading process. This case shows that for easy visual reasoning examples, slow reasoning is correct but unnecessary.

Case 2: MathVision Transfer Example



Question: Which number should be written in place of the question mark?
Ground Truth: 60

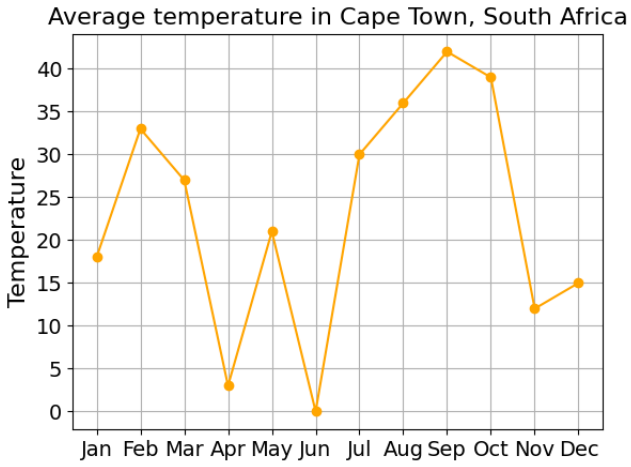
DR-BOLT output *(180 tokens):* The model identifies the sequence in the train cars:

$$90,\ 80,\ 70,\ ?,\ 50,\ 40,\ 30,\ 20.$$

It recognizes that the numbers decrease by 10 at each step, infers that the missing value is $70 - 10 = 60$, and outputs 60.

Observation: DR-BOLT activates a short reasoning process because solving the problem requires recognizing a visual numerical pattern.

Case 3: DynaMath Transfer Example



Question: Use the graph to answer the question below. Which month is the hottest on average in Cape Town?
Ground Truth: Sep

DR-BOLT output *(138 tokens):* The model inspects the line graph, compares the monthly temperature values, and identifies the highest point on the curve. The peak occurs in September, where the temperature is slightly above $40°$C. It therefore answers September.

Observation: The model uses reasoning to inspect the chart and compare values, rather than relying on a direct recognition shortcut.

Case 4: MMMU-Pro Transfer Example



Question: The image is a symbol of which option? Options include: A. A memory; B. A transition; C. An opportunity; D. A common experience; E. A journey; F. A warning; G. A clock; H. Time waits for no man; I. An obstacle; J. A mystery. Answer with the option letter only.
Ground Truth: H

DR-BOLT output *(2 tokens):* H

Observation: The image directly depicts Father Time, which corresponds to the idiom "Time waits for no man." DR-BOLT answers directly without unnecessary reasoning, showing that the model can preserve fast behavior on visually obvious symbolic questions.

NeurIPS Paper Checklist

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 and introduction state the main contributions: dual-reference offline optimization, the DR-BOLT objective, router-free fast/slow switching, and empirical improvements on ScienceQA and OOD multimodal reasoning benchmarks. The claims are supported by theoretical derivations in Sections 3–4 and experiments in Section 5.

2. **Limitations**

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

   Answer: [Yes].

   Justification: We include a Limitations section discussing the two-mode setting, the approximate mode-format separation assumption, and the fact that runtime may vary with hardware, inference engine, rollout length, and implementation details.

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: The assumptions are stated in the main text, including the approximate mode-format separation assumption. Proofs for single-reference BOLT, dual-reference optimality, dual-reference weighted SFT, routing by partition competition, and mode-wise correction are provided in the appendix.

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: The paper describes the datasets, IID/OOD split protocol, baselines, length measurement, resource measurement, and evaluation metrics. All datasets used in the paper are publicly available. We will release the code and model weights with instructions to reproduce the main 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 are public benchmarks. Our code is available at https://github.com/HongbinLin3589/DR-BOLT.

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: The paper specifies that ScienceQA is used for IID training and evaluation, while ViRL39K is used for training and MathVision, DynaMath, and MMMU-Pro are used for OOD evaluation. The appendix describes response-bank construction, reward components, correction settings, length measurement, and resource measurement. Additional implementation details will be included in the released code.

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: [No].

Justification: We report confidence intervals for the partition-based routing calibration and provide repeated-decoding statistics for routing behavior. However, due to the computational cost of training large VLMs, we do not report multi-seed error bars for all benchmark scores. We provide detailed ablations and controlled comparisons under the same hardware and data scale.

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: The appendix reports the compute setting: 4 NVIDIA A100 80GB GPUs and 32 CPU cores. The main tables report wall-clock training time, peak GPU memory, and speedup relative to reproduced GRPO. The reported time includes candidate generation, reward computation, weight construction, and policy optimization.

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: The work uses public datasets and standard model training/evaluation procedures. We are not aware of any deviation from the NeurIPS Code of Ethics.

10. **Broader impacts**

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

    Answer: [Yes].

    Justification: The method may reduce unnecessary inference cost and energy use by avoiding overthinking on simple inputs. As with other VLM reasoning methods, incorrect model outputs or overconfidence may still cause downstream risks if deployed in high-stakes settings; our work focuses on research benchmarks rather than deployment.

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 does not introduce a new scraped dataset or a general-purpose foundation model from scratch. We fine-tune existing VLMs for research on adaptive reasoning using public benchmarks. Released weights will follow the license and usage terms of the base models.

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: We cite all datasets and baseline methods used in the experiments, including ScienceQA, ViRL39K, MathVision, DynaMath, MMMU-Pro, Qwen models, and external adaptive-thinking baselines. We will document dataset and model licenses in the code/model release.

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 will release code and model weights as new research artifacts. The release will include documentation describing training data, evaluation scripts, model checkpoints, expected usage, and limitations.

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: The paper does not involve crowdsourcing or human-subject experiments. All experiments use public datasets and automated evaluation protocols.

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: The paper does not involve human-subject research, so IRB approval is not applicable.

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: [Yes].

Justification: VLMs are central to the method and experiments. The paper describes the fast and slow VLM policies, the trainable policy, and the use of public VLM checkpoints for response generation, training, and evaluation.