REINFORCING INTERNAL COMPUTATIONS FOR LARGE LANGUAGE MODEL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Mixture-of-Experts (MoE) models can solve a problem through many different combinations of experts, yet standard training largely treats expert selection as fixed internal computation. We ask whether reasoning can improve by learning not only what the model outputs, but also how it routes computation internally. We introduce a reinforcement learning approach that jointly optimizes output tokens and expert selection, allowing the model to explore alternative computation paths and reinforce those that lead to better solutions. On Moonlight-16B-A3B, our method improves over token-only GRPO by 2.06 percentage points on in-domain mathematical reasoning, 7.49 points on out-of-domain academic reasoning, and 3.83 points on out-of-domain commonsense reasoning. It also reduces mean evaluation response length by 8.3% under the same training schedule, while retaining deterministic top-K inference and the original active-expert budget per token. Our results suggest that learning how a model computes can complement learning what it generates.

1 INTRODUCTION

Mixture-of-Experts (MoE) models achieve strong scaling performance on frontier language tasks, yet their internal computation during reasoning is largely determined by a fixed routing rule (Shazeer et al., 2017; Lepikhin et al., 2020; Du et al., 2022; Fedus et al., 2022; Zoph et al., 2022; Clark et al., 2022). In standard sparse MoE inference, deterministic top-K routing collapses the router distribution into a small set of highest-scoring experts at each token and layer (Clark et al., 2022; Zhou et al., 2022; Jiang et al., 2024; DeepSeek-AI, 2024; Dai et al., 2024). This creates a deterministic sparse computation graph, potentially discarding alternative expert paths that receive lower router scores but may still support successful reasoning.

Recent reinforcement learning methods have substantially improved language models by optimizing output-token trajectories with outcome rewards (Shao et al., 2024; Guo et al., 2025; Yu et al., 2025; Liu et al., 2025b). However, these methods typically retain deterministic top-K expert selection (Ma et al., 2025; Zhang et al., 2025). The token loss can update router parameters, but does not explicitly explore alternative expert subsets at a given state. Token-level RL therefore explores answer trajectories through the expert paths selected by the current router. Successful reasoning may therefore depend not only on what tokens the model generates, but also on where the model computes internally.

In this paper, we introduce an approach for learning the *internal* computation policy of MoE language models. Instead of treating the router as a fixed top-K selector, we view it as a policy over latent computation paths. Just as standard RL explores alternative output-token sequences to discover better answers, our approach explores alternative expert pathways to discover better internal computation paths for reasoning.

Our key insight is that MoE models contain intrinsic computational diversity that can be explored and learned from. Although deterministic top-K routing selects the highest-scoring experts at each token and layer, it can also suppress nearby non-top-K expert paths that may lead to successful reasoning. To leverage these latent paths, we introduce Reinforcement Learning for Routing Policy (RLRP). RLRP jointly optimizes the token policy and router policy with reinforcement learning. It uses stochastic K-expert sampling and adds a reward-guided, expert-level objective to token-policy training by directly optimizing the advantage-weighted log likelihood of replayed, ordered expert actions. Our analysis establishes its on-policy reward-gradient interpretation and explains the effects of group-mean advantages, token clipping, and replay.

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

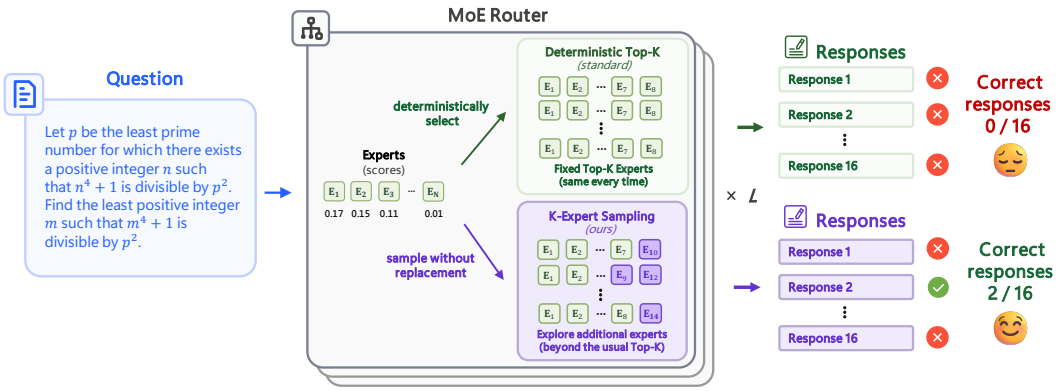

Figure 1: **K-expert sampling reveals useful computation paths hidden by deterministic top-K.** Standard deterministic top-K routing selects a fixed expert subset for each given token-layer state. In this example, all sampled responses fail under the sampling budget. In contrast, K-expert sampling draws exactly K experts without replacement from the router distribution, allowing the model to explore additional expert paths beyond the usual top-K set. Some of these alternative paths lead to correct solutions while keeping the same number of activated experts per token. Correct responses: $0/16$ with deterministic top-K routing and $2/16$ with K-expert sampling. This suggests that MoE reasoning failures can arise not only from token generation, but also from committing to a suboptimal internal computation path.

RLRP complements standard token-level RL by optimizing where the model computes while token-level RL optimizes what the model generates. Our framework is lightweight and modular: RLRP updates only the router projections for the routing-policy objective, while the token-level RL objective updates the model as usual. Since sampled routing still activates exactly K experts, the method changes which experts are selected rather than how many are activated. The routing objective can be combined with advantage-based token-policy objectives without changing the active-expert budget.

Empirically, we evaluate RLRP on Moonlight-16B-A3B (Liu et al., 2025a). With RL training on mathematical reasoning data, GRPO+RLRP improves average accuracy over token-only GRPO, with absolute gains of 2.06, 7.49, and 3.83 percentage points on in-domain mathematical, out-of-domain academic, and out-of-domain commonsense reasoning, respectively. Adding the routing objective also improves the seven-benchmark average by 4.11 percentage points over GRPO+K-expert sampling, which uses the same sampler, routing replay, and token objective. Under the same training schedule, RLRP also reduces mean evaluation response length by 8.3% relative to GRPO. This combination of higher accuracy and shorter responses supports a key advantage of router learning: using the internal computation graph more effectively to reason with fewer generated tokens. These gains come with only a 2.86% increase in total training time over GRPO, while expert load balance remains nearly unchanged.

2 Related Work

Sparse routing and expert allocation. Sparse MoE routers allocate computation and shape expert specialization (Shazeer et al., 2017; Fedus et al., 2022). Prior work studies load balancing (Lewis et al., 2021; Zhou et al., 2022) and differentiable or probabilistic routing (Wang et al., 2025; Vahidi et al., 2026). RLRP focuses on reward-guided, task-specific adaptation while retaining the pretrained active-expert budget.

Learning routing policies. ERL (Kim & Kang, 2025) uses ground-truth next-token likelihoods for routing advantages, so its original advantage construction does not directly apply to reasoning RL with only outcome rewards. RLRP jointly optimizes token and routing policies from response-level rewards.

MoE-GRPO (Ko et al., 2026) treats expert selection as an RL action and jointly optimizes token and expert policies with GRPO, using stochastic expert selection but greedy token decoding during

rollout. RLRP samples both tokens and experts. Unlike MoE-GRPO's clipped expert-policy ratio objective, RLRP optimizes the advantage-weighted log likelihood of ordered routing actions with exact conditional probabilities under sampling without replacement.

Most recently, ESRL (He et al., 2026) adds stochastic expert selection to token-level GRPO without a separate objective for discrete routing actions. RLRP directly optimizes their log likelihood with reward-derived advantages, turning exploration into reward-guided routing-policy learning.

Routing consistency. We use Rollout Routing Replay (Ma et al., 2025) to reduce rollout–training routing mismatch. RLRP complements replay by learning routing probabilities from task rewards. Additional routing and RL work is discussed in Appendix A.2.

3 METHOD

We study how to improve MoE reasoning by learning both what the model generates and how it computes internally. Pretraining provides useful expert-allocation preferences, but these need not be optimal for reasoning or other downstream tasks. During RL post-training, our method explores token continuations and alternative expert paths while jointly optimizing the token and router policies. Reinforcement Learning for Routing Policy (RLRP) adds a reward-guided, expert-level auxiliary objective to token-policy training, reinforcing sampled internal computation paths associated with successful reasoning while preserving the active-expert budget per token and MoE layer.

3.1 MOE ROUTING AS A COMPUTATION GRAPH

Consider a decoder-only MoE language model. At token position t and MoE layer l, let $h_{t,l}$ denote the hidden state after self-attention. The router produces logits $o_{t,l}$ over N routed experts. We define the selection scores $a_{t,l}$ according to the model's routing rule:

$$o_{t,l} = W_l h_{t,l}, \qquad a_{t,l,i} = \begin{cases} o_{t,l,i}, & \text{softmax router,} \\ \sigma(o_{t,l,i}) + b_{l,i}, & \text{sigmoid router.} \end{cases} \tag{1}$$

Here σ is the sigmoid function, and b_l is the expert correction bias. Standard routing selects

$$S_{t,l}^{\text{top-}K} = \text{TopK}(a_{t,l}, K). \tag{2}$$

For a selected subset S, the routed-expert output is

$$\text{MoE}_{\text{routed}}(h_{t,l}; S) = c_l \sum_{i \in S} \bar{w}_{t,l,i}(S) E_{i,l}(h_{t,l}), \qquad \bar{w}_{t,l,i}(S) = \frac{w_{t,l,i}}{\sum_{j \in S} w_{t,l,j}}. \tag{3}$$

Here $w_{t,l,i} = \exp(o_{t,l,i})$ for softmax routers and $w_{t,l,i} = \sigma(o_{t,l,i})$ for sigmoid routers, and c_l retains the model's original routed-output scaling factor. The correction bias affects selection only and is excluded from the clean mixing weights. These token- and layer-dependent expert choices define a sparse computation graph. Deterministic top-K selects a single subset at each given routing state.

From pretraining to task-specific routing. The router parameters $\theta = \{W_l\}_l$ are learned jointly with the experts through the language-modeling loss and model-specific auxiliary objectives (Shazeer et al., 2017). The resulting routing preferences are shaped by the pretraining distribution and objectives, and need not be optimal for reasoning or other downstream tasks. This motivates adapting the model's internal computation policy alongside its token policy during RL post-training. To discover and learn better task-specific routing, we first define expert-selection actions and a stochastic policy for exploring alternative internal computation paths.

3.2 K-EXPERT SAMPLING

For either router type, given selection scores $a_{t,l}$ and temperature $\tau > 0$, we define

$$p_{t,l,i}^{(\tau)} = \frac{\exp(a_{t,l,i}/\tau)}{\sum_{j=1}^{N} \exp(a_{t,l,j}/\tau)}, \qquad S_{t,l}^{\text{sample-}K} \sim \text{SampleWithoutReplacement}(p_{t,l}^{(\tau)}, K). \tag{4}$$

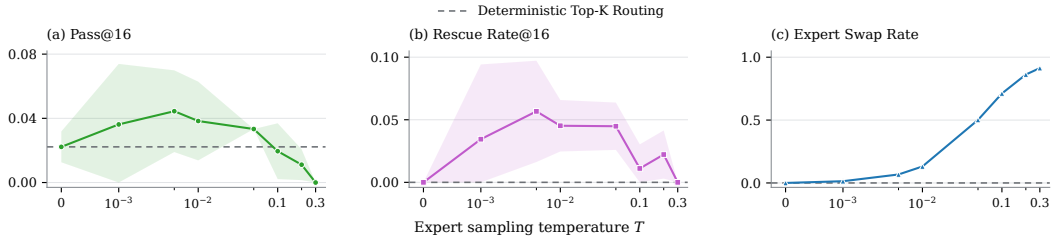

Figure 2: **Moderate expert sampling improves Pass@16.** We evaluate Moonlight-16B-A3B Base on the AIME25 dataset. Expert sampling uses a shallow-to-deep temperature schedule across the MoE layers, increasing linearly from 0 to the endpoint T shown on the horizontal axis. The $T = 0$ point uses deterministic top-K routing. **(Left)** Pass@16. **(Middle)** Rescue rate@16: among questions missed by the first 16 top-K responses, the fraction solved by the first 16 sampled-routing responses, paired by question and seed. **(Right)** Expert swap rate: the fraction of decoding token-layer decisions whose sampled expert set differs from deterministic top-K on the same hidden state. All rates are plotted as fractions. Lines and shaded bands show the mean and ± 1 sample standard deviation across three seeds.

Sampling proceeds sequentially: each selected expert is removed, and the remaining probabilities are renormalized before the next draw. After selecting K distinct experts, we combine their outputs using the original, untempered weights in Eq. (3). Temperature therefore controls expert selection while preserving the mixing rule conditional on the selected subset. Each token activates exactly K routed experts per MoE layer.

Exploration around learned routing preferences. The Gumbel-Top-K representation of sampling without replacement (Kool et al., 2019) gives

$$S_{t,l}^{\text{sample-}K} \overset{d}{=} \text{TopK}(a_{t,l} + \tau\xi, K), \qquad \xi_i \overset{\text{i.i.d.}}{\sim} \text{Gumbel}(0, 1). \tag{5}$$

Thus, exploration perturbs the current routing scores, with τ controlling the perturbation scale. The initial scores come from pretraining, and the sampling distribution subsequently follows the updated router. For a fixed score vector a, write $P_a^\tau(S)$ for the probability that Eq. (5) selects the unordered subset S.

At a fixed routing state, a large top-K boundary margin relative to τ concentrates sampling on the default subset. The controlled-deviation bound in Appendix C.1 applies to both router types with the margin measured in the selection scores a.

Motivating probe. Our fixed-model inference probe in Figure 2 measures expert swap rate, pass@16, and rescue rate@16. Expert swap rate is the fraction of token-layer decisions whose sampled expert subset differs from deterministic top-K. Rescue rate is the fraction of questions missed by the deterministic-routing sample that sampled routing solves within the same response budget. Moderate exploration improves pass@16 and rescues some questions missed by deterministic routing. Excessive exploration reduces both pass@16 and rescue rate, indicating that overly large routing perturbations can impair reasoning performance. This trade-off motivates retaining the learned routing preferences during exploration. We next use outcome feedback to learn from the executed routes, turning training-time exploration into a source of routing supervision.

3.3 Reinforcement Learning for Routing Policy

For each prompt x, let $y^{(g)}$ denote a response generated by the rollout policy with stochastic K-expert sampling, and let $A_t^{(g)}$ be the advantage estimate supplied by the token-level RL algorithm at token t, held fixed during optimization. This estimate can come from GRPO, PPO, or another advantage-based RL algorithm.

Replay of ordered routing actions. Although the expert mixture depends only on the selected subset, the sampler generates an ordered sequence of expert choices. Retaining this sampled order provides an exact action likelihood without explicitly summing over all $K!$ permutations of the subset.

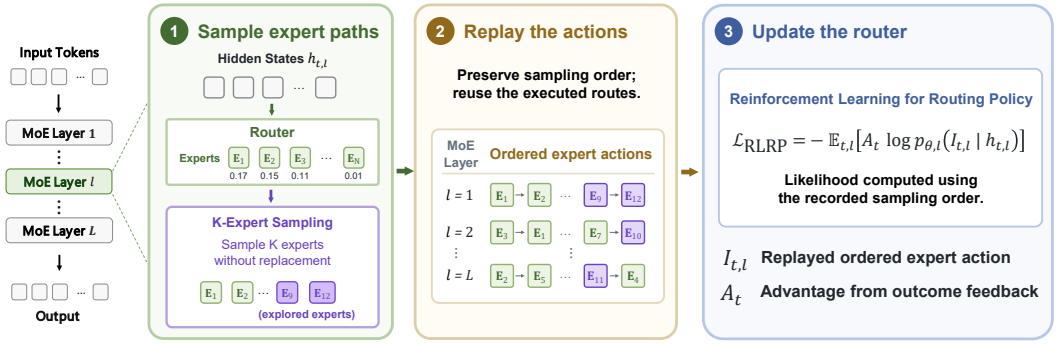

Figure 3: **Reinforcement Learning for Routing Policy.** Rollout records the ordered expert actions that produce each response. Training maximizes their advantage-weighted log likelihood, retaining only tokens whose token-policy surrogate is not clipped. The token objective is optimized jointly using the same replayed expert paths.

During rollout, we record the ordered expert action $I_{t,l}^{(g)} = (i_1, \ldots, i_K)$, preserving the descending order of the Gumbel-perturbed scores, and let $S_{t,l}^{(g)}$ denote its underlying subset. During optimization, we replay these expert identities for both the routing and token-policy updates, following Rollout Routing Replay (Ma et al., 2025). This keeps expert selections consistent between the inference and training engines despite numerical differences and execution nondeterminism. Sorting the recorded experts by their IDs or unperturbed scores would change the action likelihood.

The routing-loss branch uses detached hidden states, $o_{t,l}^{\text{policy}} = W_l \,\text{stopgrad}(h_{t,l})$, and computes $a_{t,l}^{\text{policy}}$ using Eq. (1) with the same fixed correction bias. For $\tau_l > 0$, its ordered-action probability is

$$p_{\theta,l}(I_{t,l} \mid h_{t,l}) = \prod_{k=1}^{K} \frac{\exp(a_{t,l,i_k}^{\text{policy}}/\tau_l)}{\sum_{j \notin I_{t,l,<k}} \exp(a_{t,l,j}^{\text{policy}}/\tau_l)}, \qquad I_{t,l,<k} = \{i_1, \ldots, i_{k-1}\}. \tag{6}$$

Each denominator includes every remaining expert, matching sampling without replacement at the rollout temperature. The loss sums the K conditional log probabilities. Layers with $\tau_l = 0$ use deterministic routing and contribute zero to the routing loss.

Token clipping and the routing objective. We alternate between collecting fresh rollouts and taking multiple minibatch updates on each rollout batch. The recorded actions and advantages remain fixed while the token and routing policies change, so later updates use feedback from an earlier policy. GRPO clipping removes the token surrogate's incentive to move a token's probability further in the direction favored by its advantage once the current-to-rollout ratio crosses the corresponding threshold (Schulman et al., 2017; Shao et al., 2024). An unclipped routing loss can continue reinforcing or suppressing an expert action even after the corresponding token-policy surrogate has saturated. We share the token-policy clipping criterion to coordinate the two objectives. Let $r_t^{(g)}$ denote this token-probability ratio, evaluated on the replayed expert path, and define the active-token mask

$$m_t^{(g)} = \begin{cases} 0, & A_t^{(g)} > 0 \text{ and } r_t^{(g)} > 1 + \epsilon_{\text{high}}, \\ 0, & A_t^{(g)} < 0 \text{ and } r_t^{(g)} < 1 - \epsilon_{\text{low}}, \\ 1, & \text{otherwise.} \end{cases} \tag{7}$$

The mask is recomputed at each update and applies to the entire ordered routing action at every MoE layer. The advantage-weighted routing loss is

$$\mathcal{L}_{\text{RLRP}} = -\mathbb{E}_{x,g,t,l}\left[m_t^{(g)} A_t^{(g)} \log p_{\theta,l}\left(I_{t,l}^{(g)} \mid h_{t,l}^{(g)}\right)\right]. \tag{8}$$

Here $m_t^{(g)}$ and $A_t^{(g)}$ are held fixed during differentiation, and $\mathbb{E}_{x,g,t,l}$ summarizes rollout aggregation under the chosen RL recipe. Clipping thresholds and loss normalization are specified in Appendix E.1. The shared mask does not bound the routing-action probability ratio. Section 4.4 evaluates this design against unclipped routing updates and expert-ratio clipping.

Policy-gradient justification. The exact ordered likelihood gives a score-function estimator of routing reward (Williams, 1992). At a fixed routing state, let $J(\theta) = \mathbb{E}_{I\sim p_\theta}[Q(S(I))]$ for a fixed subset-conditioned reward function Q. For $G > 1$ independent on-policy samples and $\bar{R} = G^{-1}\sum_g R^{(g)}$, group-mean advantages give the expected update

$$\mathbb{E}\left[\frac{1}{G}\sum_{g=1}^{G}(R^{(g)} - \bar{R})\nabla_\theta \log p_\theta(I^{(g)})\right] = \left(1 - \frac{1}{G}\right)\nabla_\theta J(\theta). \tag{9}$$

Thus group centering preserves the reward-gradient direction. Appendix C proves these identities and extends them to a trajectory objective with the model's computation held fixed. They justify the unmasked update at the rollout policy; clipping and subsequent replay updates introduce deviations analyzed in Appendix C.4.

Joint optimization and deployment. We combine the routing loss with a token-level RL objective $\mathcal{L}_{\text{token}}$, such as GRPO (Shao et al., 2024) or PPO (Schulman et al., 2017):

$$\mathcal{L} = \mathcal{L}_{\text{token}} + \mathcal{L}_{\text{RLRP}}. \tag{10}$$

We set the coefficient of $\mathcal{L}_{\text{RLRP}}$ to 1 in all experiments, avoiding an additional tuning hyperparameter. Detached routing inputs restrict each routing-loss term to its corresponding router projection. The token objective retains its usual gradient paths through the model, including the router. Figure 3 illustrates the workflow, and Algorithm 1 summarizes the routing update.

Shared experts remain active and are excluded from stochastic selection and the routing loss. Appendix D details the Moonlight specialization and its logit gradient. For the main checkpoint evaluations, we use deterministic top-K with the learned router, retaining the original active-expert budget.

4 Experiments

We compare GRPO+RLRP with GRPO, MoE-GRPO, and GRPO+K-expert sampling on seven mathematical, academic, and commonsense reasoning benchmarks. We examine reasoning accuracy and response length (Section 4.2), analyze routing preferences across domains, total training cost, and expert load balance (Section 4.3), and compare three clipping rules for the routing objective (Section 4.4).

4.1 Setup

Model. We fine-tune Moonlight-16B-A3B (Liu et al., 2025a), a 16B-parameter MoE model with approximately 3B activated parameters. All methods retain its original budget of $K = 6$ routed experts out of $N = 64$ per token.

Training data and reward. We train on the original DAPO-Math dataset (Yu et al., 2025), containing approximately 17k examples. The reward is one if the normalized final answer matches the ground truth and zero otherwise. All methods share the prompts, verifier, and reward function.

Evaluation. The main evaluation uses Olympiad (He et al., 2024), AMC23 (Mathematical Association of America, 2023), and Minerva Math (Lewkowycz et al., 2022) for mathematical reasoning; MMLU (Hendrycks et al., 2021) and MMLU-Pro (Wang et al., 2024) for academic reasoning; and SIQA (Sap et al., 2019) and WinoGrande (Sakaguchi et al., 2020) for commonsense reasoning. We treat mathematical reasoning as in-domain and academic and commonsense reasoning as out-of-domain, relative to the math-only RL training data. We use the MMLU development split, the MMLU-Pro validation split, and fixed 100-example validation subsets for SIQA and WinoGrande. All main results use deterministic top-K expert routing. We report avg@16 for AMC23 and avg@3 for Olympiad, Minerva Math, MMLU, MMLU-Pro, SIQA, and WinoGrande. The overall score is the unweighted mean of the seven benchmark accuracies. Dataset preparation and decoding settings are given in Appendix E.2.

Table 1: **In-domain and out-of-domain evaluation on Moonlight-16B-A3B.** We compare GRPO, MoE-GRPO, GRPO+K-expert sampling, and GRPO+RLRP under the same active-expert budget. All models use default deterministic top-K expert routing at evaluation; inference-time expert sampling is disabled. We report step-100 results unless marked with †. † marks results from the best checkpoint for runs affected by training collapse. All scores are mean accuracies (%), and Avg. is the unweighted mean across the seven datasets. The best result in each column is bolded.

Method	In-domain			Out-of-domain				Avg.
	Mathematical Reasoning			Academic Reasoning		Commonsense Reasoning		
	Olympiad	AMC23	Minerva	MMLU	MMLU-Pro	SIQA	WinoGrande	
GRPO	34.39	46.88	24.26	63.95	39.22	60.67	60.33	47.10
MoE-GRPO†	9.52	14.84	8.21	9.81	3.43	11.00	16.33	10.45
GRPO+K-expert sampling	33.28	48.13	22.92	62.77	41.67	62.00	59.00	47.11
GRPO+RLRP	**35.54**	**50.31**	**25.86**	**66.67**	**51.47**	**63.67**	**65.00**	**51.22**

Compared methods. **GRPO** uses a token-level RL objective with deterministic expert routing. **MoE-GRPO** (Ko et al., 2026) adds a router-level GRPO objective to token-level GRPO, with greedy token decoding and stochastic expert selection during rollout. **GRPO+K-expert sampling** samples both tokens and expert subsets during rollout, using the same expert sampler, routing replay, and token-level GRPO objective as GRPO+RLRP. **GRPO+RLRP** additionally optimizes the advantage-weighted ordered-action routing loss, isolating the benefit of directly training the routing policy.

Training protocol. At each training step, we generate 16 responses for each of 256 prompts, with a maximum response length of 4,096 tokens. We then perform eight gradient steps on the collected batch, using minibatches of 32 prompts. Training lasts 100 rollout steps, totaling 800 gradient steps. The token and routing objectives share unstandardized group-mean advantages and fixed maximum-response-length normalization (Liu et al., 2025b). For a fair comparison, all training methods use Rollout Routing Replay (R3) (Ma et al., 2025) to replay the expert choices recorded during rollout. Appendix E.1 provides the full training recipe.

4.2 Main Results

RLRP improves in-domain and out-of-domain reasoning. Table 1 compares checkpoints under deterministic top-K expert routing. GRPO+K-expert sampling achieves an average accuracy of 47.11%, close to GRPO's 47.10%. With the sampler, routing replay, and token objective held fixed, adding the routing objective improves average accuracy from 47.11% to 51.22%.

GRPO+RLRP also achieves higher accuracy than MoE-GRPO on all seven benchmarks. Table 1 reports MoE-GRPO's best checkpoint (marked with †), as its training run was affected by collapse.

On the in-domain mathematical reasoning benchmarks, GRPO+RLRP achieves the highest accuracy on Olympiad, AMC23, and Minerva, with an absolute mean accuracy gain of 2.06 percentage points over GRPO.

The gains extend beyond the math training domain. The largest domain-level gain is on the out-of-domain academic benchmarks MMLU and MMLU-Pro, where GRPO+RLRP improves mean accuracy over GRPO by 7.49 percentage points. MMLU-Pro shows the largest single-benchmark improvement, at 12.25 percentage points. On out-of-domain commonsense reasoning, the mean gain across SIQA and WinoGrande is 3.83 percentage points. These gains show that the benefits of routing-policy training extend beyond the mathematical domain used for RL.

Accuracy gains accompany shorter responses. Figure 4 compares evaluation accuracy and response length across runs with the same numbers of training steps. GRPO+RLRP achieves the highest average evaluation accuracy and the shortest average responses on evaluation benchmarks and in training rollouts. These gains highlight the benefit of explicitly training the router: higher accuracy and more concise responses with the same number of active experts. They support the view that router learning makes more effective use of the model's internal computation graph, enabling better reasoning with fewer generated tokens.

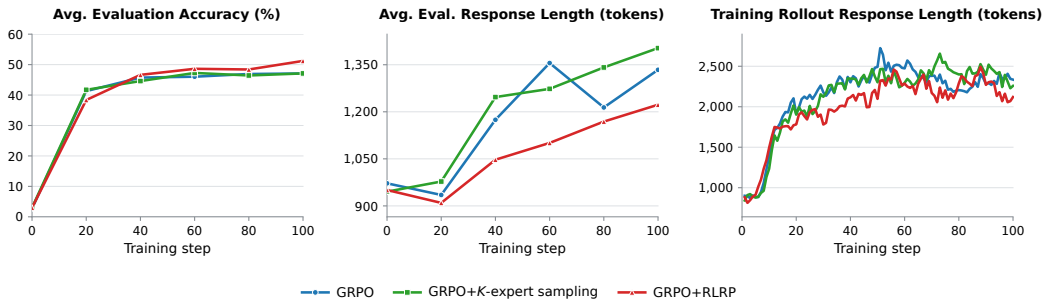

Figure 4: **Accuracy and response length during training.** We compare training runs of Moonlight-16B-A3B. **(Left)** Average accuracy across the seven evaluation benchmarks. **(Middle)** Average evaluation response length across the same seven benchmarks. Both evaluation averages weight each benchmark equally. **(Right)** Mean rollout response length on the DAPO-Math training data, measured on responses generated from training prompts during RL.

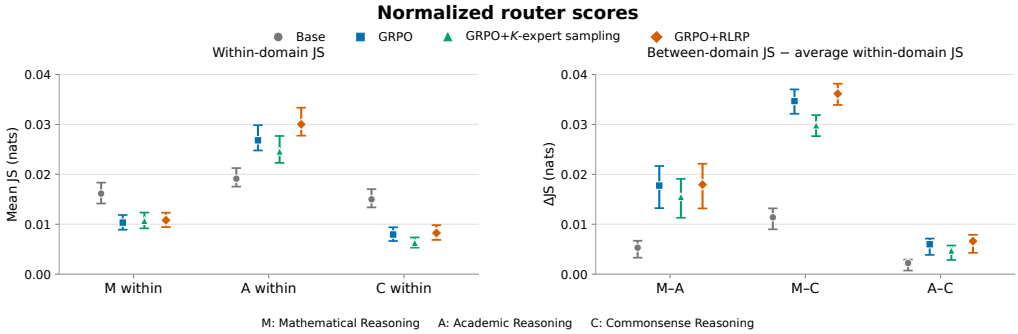

Figure 5: **Domain separation in routing.** We compare Moonlight-16B-A3B Base and the final trained checkpoints at layer 24 under deterministic top-K inference. Each prompt is represented by its normalized sigmoid router scores, averaged over generated tokens and then across responses. **(Left)** Mean within-domain Jensen–Shannon (JS) divergence. **(Right)** Mean between-domain JS minus the average within-domain JS of the two domains. Points show means; intervals show pointwise 95% prompt-bootstrap confidence intervals.

4.3 Analysis

RLRP exhibits domain-dependent routing preferences. Figure 5 shows two distinct patterns of adaptation from Base. For academic reasoning, fine-tuning produces more differentiated expert preferences across prompts, reflected in higher within-domain JS. Routing also becomes more separated between commonsense reasoning and both mathematical and academic reasoning. These changes occur under GRPO and GRPO+K-expert sampling, but RLRP yields larger mean increases in all three comparisons. RLRP thus shows greater differentiation where the baselines already move toward more distinct expert preferences.

Within mathematical reasoning and within commonsense reasoning, fine-tuning instead produces more consistent expert preferences across prompts. RLRP retains this pattern, with mean within-domain JS below Base and close to GRPO. Taken together, these patterns suggest that RLRP is associated with more differentiated expert preferences in some comparisons while retaining consistency within other domains. Alongside the accuracy gains, these observations are consistent with task-specific routing adaptation.

Computational cost analysis. Although optimizing the router loss adds a small computational overhead, shorter responses help limit GRPO+RLRP's increase in total training time to 2.86% over GRPO (Table 2). To measure expert load balance, we count the tokens routed to each routed expert during deterministic top-K inference on AIME25, divide the standard deviation of these counts by

Table 2: **Computational cost and expert load balance.** We report total GPU-hours for 100 training steps on four GPUs, including setup and evaluation. Expert load CV is the coefficient of variation of per-expert token counts, averaged over MoE layers; lower values indicate better balance.

Method	Training time (GPU-hours)	Expert load CV
GRPO	235.14	1.5593
GRPO+RLRP	241.87	1.5745

Table 3: **Clipping ablation for RLRP on Moonlight-16B-A3B.** We compare three clipping rules for the routing objective after 100 training steps, using the same seven benchmarks and evaluation protocol as Table 1. All scores are mean accuracies (%). Avg. is the unweighted mean across the seven datasets, and the best result in each column is bolded. GRPO clip is the configuration used for GRPO+RLRP in Table 1.

Method	In-domain			Out-of-domain				Avg.
	Mathematical Reasoning			Academic Reasoning		Commonsense Reasoning		
	Olympiad	AMC23	Minerva	MMLU	MMLU-Pro	SIQA	WinoGrande	
Unclipped	33.83	47.50	**26.35**	65.13	46.08	**63.67**	60.33	48.98
Expert-ratio clip	34.54	47.19	24.88	**68.44**	49.51	62.67	62.00	49.89
GRPO clip	**35.54**	**50.31**	25.86	66.67	**51.47**	**63.67**	**65.00**	**51.22**

their mean at each MoE layer, and average this coefficient of variation (CV) over layers. RLRP's CV is only 1.0% higher than GRPO's, indicating comparable expert load balance.

4.4 Clipping Ablation

During minibatch optimization, the policy changes while rollout actions and advantages remain fixed. To evaluate how routing updates should be controlled in this setting, we compare the shared GRPO token mask with a separate expert-level criterion and an unclipped routing loss. All variants use eight minibatch updates per rollout batch and the same clipped GRPO token objective, with all other training and evaluation settings held fixed.

Unclipped retains every ordered routing log-likelihood term, equivalently setting $m_t^{(g)} = 1$ in Eq. (8). **GRPO clip** uses the token-policy mask in Eq. (7), with $\epsilon_{\text{low}} = 0.2$ and $\epsilon_{\text{high}} = 0.28$. A clipped token contributes no routing loss at any MoE layer. **Expert-ratio clip** instead masks individual selected-expert terms using their current-to-rollout clean router-probability ratios. For Moonlight, define

$$p_{t,l,i}^{\text{clean}} = \frac{\sigma(o_{t,l,i})}{\sum_{j=1}^{N} \sigma(o_{t,l,j})}, \qquad r_{t,l,k}^{\text{expert}} = \frac{p_{t,l,i_k}^{\text{clean,policy}}}{p_{t,l,i_k}^{\text{clean,rollout}}}. \tag{11}$$

These clean probabilities are normalized over all experts and exclude the sampling temperature, Gumbel noise, and correction bias. The k-th ordered log-likelihood term is masked out when $A_t^{(g)} > 0$ and $r_{t,l,k}^{\text{expert}} > 1.1$, or when $A_t^{(g)} < 0$ and $r_{t,l,k}^{\text{expert}} < 0.9$. The mask is held fixed during differentiation and leaves the conditional sampling denominators unchanged. This variant uses no additional token-policy mask for the routing loss.

GRPO clip achieves the highest average accuracy (Table 3). Sharing the token mask may better coordinate token and router learning: it suppresses auxiliary routing updates when the token surrogate saturates and preserves the complete ordered-action loss at active tokens.

5 Conclusion

We introduced RLRP, which learns MoE routing decisions from outcome rewards through stochastic expert sampling and an advantage-weighted ordered-action objective. Combined with GRPO, RLRP improves accuracy on in-domain and out-of-domain reasoning benchmarks while producing shorter responses. Our findings suggest reinforcing the internal computation policy is a promising direction to improve the reasoning ability of MoE models.

AI USE STATEMENT

We used generative AI tools to identify typographical errors, search for related work, assist with code implementation and debugging, and help complete parts of the mathematical proofs. All AI-assisted code and written content were reviewed and verified by the authors. The authors take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

The routing policy and learning objective are specified in Section 3, with algorithm pseudocode in Appendix B. Appendix C provides the assumptions and proofs for the theoretical results. Section 4.1 and Appendices D and E document the Moonlight routing implementation, training and evaluation protocols, hyperparameters, reward computation, prompt template, hardware, and dataset preparation.

REFERENCES

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Aidan Clark, Diego de Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake Hechtman, Trevor Cai, Sebastian Borgeaud, et al. Unified scaling laws for routed language models. In *International conference on machine learning*, pp. 4057–4086. PMLR, 2022.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1280–1297, 2024.

DeepSeek-AI. Deepseek-v3 technical report, 2024. URL https://arxiv.org/abs/2412.19437.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. Glam: Efficient scaling of language models with mixture-of-experts. In *International conference on machine learning*, pp. 5547–5569. PMLR, 2022.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Gökdeniz Gülmez. Dynamoe: Dynamic token-level expert activation with layer-wise adaptive capacity for mixture-of-experts neural networks. *arXiv preprint arXiv:2603.01697*, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.211. URL https://aclanthology.org/2024.acl-long.211/.

Hongyi He, Zhenghao Lin, Xiao Liu, Peng Cheng, Yan Lu, and Yeyun Gong. Expert-space exploration in MoE reinforcement learning. *arXiv preprint arXiv:2609.13058*, 2026. URL https://arxiv.org/abs/2609.13058v1.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2009.03300.

Ailin Huang, Ang Li, Aobo Kong, Bin Wang, Binxing Jiao, Bo Dong, Bojun Wang, Boyu Chen, Brian Li, Buyun Ma, et al. Step 3.5 flash: Open frontier-level intelligence with 11b active parameters. *arXiv preprint arXiv:2602.10604*, 2026.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Can Jin, Hongwu Peng, Mingcan Xiang, Qixin Zhang, Xiangchi Yuan, Amit Hasan, Ohiremen Dibua, Yifan Gong, Yan Kang, and Dimitris N Metaxas. Sparsity-controllable dynamic top-p moe for large foundation model pre-training. *arXiv preprint arXiv:2512.13996*, 2025.

Gyunyeop Kim and Sangwoo Kang. Exploration-driven reinforcement learning for expert routing improvement in mixture-of-experts language models. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 23592–23605. Association for Computational Linguistics, 2025.

Dohwan Ko, Jinyoung Park, Seoung Choi, Sanghyeok Lee, Seohyun Lee, and Hyunwoo J Kim. Moe-grpo: Optimizing mixture-of-experts via reinforcement learning in vision-language models. *arXiv preprint arXiv:2603.24984*, 2026.

Wouter Kool, Herke Van Hoof, and Max Welling. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In *International conference on machine learning*, pp. 3499–3508. PMLR, 2019.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*, 2020.

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. Base layers: Simplifying training of large, sparse models. In *International Conference on Machine Learning*, pp. 6265–6274. PMLR, 2021.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35: 3843–3857, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/18abbeef8cfe9203fdf9053c9c4fe191-Abstract-Conference.html.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, et al. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*, 2025a.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025b.

Chiyu Ma, Shuo Yang, Kexin Huang, Jinda Lu, Haoming Meng, Shangshang Wang, Bolin Ding, Soroush Vosoughi, Guoyin Wang, and Jingren Zhou. Fipo: Eliciting deep reasoning with future-kl influenced policy optimization. *arXiv preprint arXiv:2603.19835*, 2026.

Wenhan Ma, Hailin Zhang, Liang Zhao, Yifan Song, Yudong Wang, Zhifang Sui, and Fuli Luo. Stabilizing moe reinforcement learning by aligning training and inference routers. *arXiv preprint arXiv:2510.11370*, 2025.

Mathematical Association of America. American Mathematics Competitions (AMC 10/12). Mathematics Competition Series, 2023. URL https://maa.org/math-competitions/amc.

Mathematical Association of America. American Invitational Mathematics Examination (AIME). Mathematics Competition Series, 2025. URL https://maa.org/math-competitions/aime.

Penghui Qi, Xiangxin Zhou, Zichen Liu, Tianyu Pang, Chao Du, Min Lin, and Wee Sun Lee. Rethinking the trust region in llm reinforcement learning. *arXiv preprint arXiv:2602.04879*, 2026.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020. doi: 10.1609/aaai.v34i05.6399. URL https://ojs.aaai.org/index.php/AAAI/article/view/6399.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4463–4473. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1454. URL https://aclanthology.org/D19-1454/.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

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

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL https://jmlr.org/papers/v15/srivastava14a.html.

Ling Team, Anqi Shen, Baihui Li, Bin Hu, Bin Jing, Cai Chen, Chao Huang, Chao Zhang, Chaokun Yang, Cheng Lin, et al. Every step evolves: Scaling reinforcement learning for trillion-scale thinking model. *arXiv preprint arXiv:2510.18855*, 2025.

Amirhossein Vahidi, Hesam Asadollahzadeh, Navid Akhavan Attar, Marie Moullet, Kevin Ly, Xingyi Yang, and Mohammad Lotfollahi. Dirmoe: Dirichlet-routed mixture of experts. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 155806–155824, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/fc0e3f908a2116ba529ad0a1530a3675-Paper-Conference.pdf.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ad236edc564f3e3156e1b2feafb99a24-Abstract.html.

Ziteng Wang, Jun Zhu, and Jianfei Chen. Remoe: Fully differentiable mixture-of-experts with relu routing. In *The Thirteenth International Conference on Learning Representations*, 2025.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

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

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

Di Zhang, Xun Wu, Shaohan Huang, Lingjie Jiang, Yaru Hao, Li Dong, Zewen Chi, Zhifang Sui, and Furu Wei. Towards stable and effective reinforcement learning for mixture-of-experts. *arXiv preprint arXiv:2510.23027*, 2025.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Chujie Zheng, Kai Dang, Bowen Yu, Mingze Li, Huiqiang Jiang, Junrong Lin, Yuqiong Liu, Hao Lin, Chencan Wu, Feng Hu, et al. Stabilizing reinforcement learning with llms: Formulation and practices. *arXiv preprint arXiv:2512.01374*, 2025a.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025b.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody H Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Sglang: Efficient execution of structured language model programs. *Advances in neural information processing systems*, 37:62557–62583, 2024.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Zhifeng Chen, Quoc V Le, and James Laudon. Mixture-of-experts with expert choice routing. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 7103–7114. Curran Associates, Inc., 2022. doi: 10.52202/068431-0515. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/2f00ecd787b432c1d36f3de9800728eb-Paper-Conference.pdf.

Zilin Zhu, Chengxing Xie, Xin Lv, and slime Contributors. slime: An llm post-training framework for rl scaling. https://github.com/THUDM/slime, 2025. GitHub repository. Corresponding author: Xin Lv.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. St-moe: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*, 2022.

A Discussions

A.1 Limitations and Future Work

RL training is restricted to mathematical reasoning, although evaluation includes academic and commonsense tasks. This setting allows us to use reliable rule-based rewards and obtain a clean analysis of routing behavior, but it does not fully characterize how routing-policy learning transfers to broader domains such as coding, tool use, multi-turn interaction, open-ended instruction following, or multimodal reasoning in vision-language models. Future work should evaluate whether reward-driven routing can improve more diverse reasoning abilities and whether the learned routing policy remains robust under different task distributions and model families.

A second limitation is that our method optimizes routing from sequence-level rewards. While this design keeps the training signal aligned with final task success, it also makes credit assignment challenging: a successful response may depend on many token-level and layer-level routing decisions, and the current objective does not explicitly identify which expert choices are causally responsible for the reward. More fine-grained training signals, such as token-level credit assignment, process-level feedback, and on-policy self-distillation, may help assign denser supervision to intermediate reasoning steps and internal computation paths, thereby improving the efficiency and stability of routing-policy optimization (Ma et al., 2026; Zhao et al., 2026; Yang et al., 2026).

Finally, our current formulation directly combines routing-policy optimization with token-policy optimization in a single RL training objective. This simple joint optimization makes the approach easy to integrate into existing RL post-training pipelines, but the interaction between these two learning processes is still not well understood. In particular, token-policy updates change the states and trajectories on which routing decisions are made, while router updates change the internal computation paths that support token generation. Future work should study this coupled optimization problem more systematically, including how to balance the two objectives, whether they should be updated at different timescales, and when their interaction leads to either synergy or instability.

A.2 Extended Related Work

Discrete routing and adaptive expert allocation. Discrete top-K selection connects router learning to gradient estimation for stochastic and non-differentiable decisions (Bengio et al., 2013). Work on scalable sparse activation (Lepikhin et al., 2020; Clark et al., 2022) and adaptive or exploration-based routing (Jin et al., 2025; Gülmez, 2026; Kim & Kang, 2025) further studies how experts are allocated. These directions complement the task-specific routing adaptation considered in this work.

RL objectives and training stability. Routing consistency is important for stable MoE RL (Ma et al., 2025; Zhang et al., 2025). Beyond routing replay, work on RL optimization includes sequence-level objectives (Zheng et al., 2025b), trust-region and optimization variants (Zheng et al., 2025a; Qi et al., 2026), and practical large-scale training recipes (Team et al., 2025; Huang et al., 2026). These studies address the stability and scalability of policy updates. Our routing objective introduces an additional discrete action space within the model, while sharing the token-policy clipping criterion to coordinate the two updates. Section 4.4 examines this choice against unclipped routing updates and expert-ratio clipping.

B RLRP: Algorithm

Algorithm 1 gives the direct routing-likelihood update with the current GRPO token mask. Rollout records the ordered expert actions, and training replays the same expert paths for both objectives. Selection scores follow Eq. (1), so the update applies to both router types. Index t denotes the forward position predicting response token y_t, so routing actions, token advantages, and clip masks are aligned to the same prediction. Prompt-only and padding positions do not contribute to the loss.

Algorithm 1 Reinforcement Learning for Routing Policy (`RLRP`).

Require: Prompt batch $\mathcal{B}$, rollout model ϕ_{old}, router projections $\{W_l\}_{l=1}^{L}$, layer temperatures $\{\tau_l\}$, expert budget K, group size G, maximum response length M

Collect rollouts
for each prompt $x \in \mathcal{B}$ **do**
  for $g = 1, \ldots, G$ **do**
    Generate $y^{(g)}$ with ϕ_{old} and K-expert sampling.
    Record every ordered action $I_{t,l}^{(g)}$ in descending Gumbel-score order.
    Record the token log probabilities and evaluate reward $R^{(g)}$.
  end for
  Set $A^{(g)} = R^{(g)} - G^{-1} \sum_j R^{(j)}$; detach all advantages.
end for
Optimize on replayed routes
for each optimization minibatch **do**
  Replay the recorded expert choices and compute the current token log probabilities.
  Form $\mathcal{L}_{\text{token}}$ and recompute the detached active-token mask $m_t^{(g)}$ using Eq. (7).
  Initialize $\mathcal{L}_{\text{RLRP}} \leftarrow 0$.
  for each response-predicting token t in each replayed response g **do**
    for each MoE layer l with $\tau_l > 0$ **do**
      $o \leftarrow W_l \,\text{stopgrad}(h_{t,l}^{(g)})$; $I \leftarrow I_{t,l}^{(g)}$.
      Compute selection scores a from o using Eq. (1).
      Evaluate the ordered log likelihood:

$$\ell_{t,l}^{(g)} \leftarrow \sum_{k=1}^{K} \left[\frac{a_{i_k}}{\tau_l} - \text{logsumexp}_{j \notin I_{<k}} \left(\frac{a_j}{\tau_l} \right) \right].$$

      Accumulate $-m_t^{(g)} A^{(g)} \ell_{t,l}^{(g)}$.
    end for
  end for
  Average over responses and divide by ML, retaining these denominators for clipped tokens and zero-temperature layers.
  Update with $\mathcal{L}_{\text{token}} + \mathcal{L}_{\text{RLRP}}$.
end for

C Analysis of Sampling and Direct Routing Policy Gradients

This appendix analyzes the sampler and the direct ordered-action objective in Section 3. We first fix the routing state, use finite selection scores $a \in \mathbb{R}^N$ from Eq. (1), and assume $1 \le K < N$ and $\tau > 0$. Thus $a = o$ for a softmax router and $a = \sigma(o) + b_l$ for a sigmoid router. Temperatures and correction biases are held fixed throughout differentiation. The sampler is

$$S = \mathrm{TopK}(a + \tau\xi, K), \qquad \xi_i \overset{\text{i.i.d.}}{\sim} \mathrm{Gumbel}(0, 1), \tag{12}$$

and $P_a^\tau(S)$ is its probability of selecting the unordered subset S. We then establish policy-gradient identities for subset rewards, group-mean advantages, and a trajectory objective with frozen model computation. Finally, Appendix C.4 separates these identities from the effects of token clipping and replay. Appendix D gives Moonlight-specific sampling details and the corresponding logit gradient.

C.1 Sampling distribution and exploration range

The Gumbel-Top-K representation is equivalent to sequential probability-proportional sampling without replacement (Kool et al., 2019). In particular, for an ordered sequence of distinct experts $I = (i_1, \ldots, i_K)$,

$$\overset{\tau}{\Pr_a}(I) = \prod_{k=1}^{K} \frac{\exp(a_{i_k}/\tau)}{\sum_{j \notin \{i_1, \ldots, i_{k-1}\}} \exp(a_j/\tau)}. \tag{13}$$

Let $\mathrm{Perm}(S)$ denote the permutations of a K-element subset S. Its selection probability is

$$P_a^\tau(S) = \sum_{I \in \mathrm{Perm}(S)} \overset{\tau}{\Pr_a}(I). \tag{14}$$

The mixture depends on the unordered subset. RLRP records the sampled order and uses $p_a(I) = \Pr_a^\tau(I)$ as its action likelihood.

Proposition C.1 (Controlled deviation from top-K)**.** *Fix a routing state, finite scores $a \in \mathbb{R}^N$, $1 \le K < N$, and $\tau > 0$. Let $S_0 = \mathrm{TopK}(a, K)$ and suppose the boundary margin is $\gamma = a_{(K)} - a_{(K+1)} > 0$, where $a_{(j)}$ is the j-th largest score. Then*

$$1 - P_a^\tau(S_0) \le \min\{1,\ K(N-K)\exp(-\gamma/\tau)\}. \tag{15}$$

Moreover, $P_a^\tau(S) > 0$ for every K-element subset S.

Proof. Let $S_0 = \mathrm{TopK}(a, K)$ and $\gamma = a_{(K)} - a_{(K+1)} > 0$. If the perturbed top-$K$ subset differs from S_0, at least one excluded expert must overtake an included expert. Hence,

$$\{\mathrm{TopK}(a + \tau\xi, K) \ne S_0\} \subseteq \bigcup_{i \in S_0,\, j \notin S_0} \{a_j + \tau\xi_j > a_i + \tau\xi_i\}, \tag{16}$$

up to the probability-zero event of a perturbed-score tie.

For independent standard Gumbel variables, their difference has the standard logistic distribution. To obtain its cumulative distribution directly, write $F(u) = \exp(-e^{-u})$ and $f(u) = e^{-u}\exp(-e^{-u})$. Then

$$\begin{aligned} \Pr(\xi_j - \xi_i \le z) &= \int_{-\infty}^{\infty} F(u+z) f(u)\, du \\ &= \int_0^{\infty} \exp[-(1+e^{-z})v]\, dv = \frac{1}{1+e^{-z}}, \end{aligned} \tag{17}$$

where the second line uses $v = e^{-u}$. Consequently, for $i \in S_0$ and $j \notin S_0$,

$$\Pr(a_j + \tau\xi_j > a_i + \tau\xi_i) = \frac{1}{1 + \exp((a_i - a_j)/\tau)} \le \exp(-\gamma/\tau), \tag{18}$$

since $a_i - a_j \ge \gamma$. Applying a union bound to Eq. (16) yields

$$1 - P_a^\tau(S_0) \le \sum_{i \in S_0,\, j \notin S_0} \frac{1}{1 + \exp((a_i - a_j)/\tau)} \le K(N-K)\exp(-\gamma/\tau). \tag{19}$$

Combining this with the upper bound of one proves Eq. (15).

For full support, fix any K-element subset S. Every factor in Eq. (13) is strictly positive because the scores are finite and $\tau > 0$. Thus every ordering $I \in \mathrm{Perm}(S)$ has positive probability, and summing over these orders gives $P_a^\tau(S) > 0$. □

For a fixed positive boundary margin, Eq. (15) tends to zero as $\tau \to 0$. At larger temperatures, access to alternatives is governed by the individual inside–outside score differences in Eq. (19). The bound is local to a routing state; changing a route during generation can also change subsequent states.

C.2 Exact likelihood and latent sampling order

At a fixed state, write $p_\theta(I)$ for the probability of the recorded ordered action and $P_\theta(S) = \sum_{I \in \mathrm{Perm}(S)} p_\theta(I)$ for the subset probability. Let $s_\theta(I) = \nabla_\theta \log p_\theta(I)$. Assume the selection scores are differentiable in θ. Finite scores and positive temperatures make these probabilities strictly positive, and differentiation commutes with the finite sums.

Proposition C.2 (Ordered actions recover the subset-policy gradient)**.** *For every subset S,*

$$\mathbb{E}_{I \sim p_\theta(\cdot|S)}[s_\theta(I)] = \nabla_\theta \log P_\theta(S). \tag{20}$$

Consequently, for a fixed subset reward $Q(S)$, any action-independent baseline b, and $J(\theta) = \sum_S P_\theta(S)Q(S)$,

$$\mathbb{E}_{I \sim p_\theta}[(Q(S(I)) - b)s_\theta(I)] = \nabla_\theta J(\theta). \tag{21}$$

The same identity holds with a stochastic reward R satisfying $\mathbb{E}[R \mid I] = Q(S(I))$, with the conditional reward law held fixed.

Proof. Conditioning on the subset gives

$$\begin{aligned}\mathbb{E}[s_\theta(I) \mid S] &= \sum_{I \in \mathrm{Perm}(S)} \frac{p_\theta(I)}{P_\theta(S)} \frac{\nabla_\theta p_\theta(I)}{p_\theta(I)} \\ &= \frac{\nabla_\theta P_\theta(S)}{P_\theta(S)}.\end{aligned} \tag{22}$$

Multiplying by $P_\theta(S)(Q(S) - b)$ and summing over S proves Eq. (21), since $\sum_S \nabla_\theta P_\theta(S) = 0$. Replacing $Q(S(I))$ by R follows by conditional expectation. □

This identity explains why order can be retained as a latent action even though the expert mixture is invariant to it. Evaluating the recorded ordered likelihood needs only K conditional log probabilities, each normalized over the remaining experts, rather than explicit enumeration of $K!$ orders. The order must come from sampling; a deterministic reordering after sampling does not have distribution p_θ.

The identity is an exactness result, not a variance-reduction claim. For $A(S) = Q(S) - b$, let $Z = A(S(I))s_\theta(I)$. Its conditional mean is $\mathbb{E}[Z \mid S] = A(S)\nabla_\theta \log P_\theta(S)$, and the law of total covariance gives

$$\mathrm{Cov}(Z) = \mathrm{Cov}(\mathbb{E}[Z \mid S]) + \mathbb{E}[\mathrm{Cov}(Z \mid S)]. \tag{23}$$

Explicitly marginalizing the order would remove the last, positive-semidefinite term. Recording the order instead makes the likelihood inexpensive to evaluate.

Selection-score gradient. For either router type, define the conditional distribution over remaining experts at draw k by

$$p_{a,j}^{(k)} = \frac{\mathbf{1}[j \notin I_{<k}] \exp(a_j/\tau)}{\sum_{u \notin I_{<k}} \exp(a_u/\tau)}. \tag{24}$$

The gradient with respect to these selection scores is

$$\nabla_a \log p_a(I) = \frac{1}{\tau} \sum_{k=1}^{K} (e_{i_k} - p_a^{(k)}), \qquad \|\nabla_a \log p_a(I)\|_2 \le \frac{\sqrt{2}K}{\tau}, \tag{25}$$

where e_i is a coordinate vector. The bound follows from $\|e_i - p\|_2 \le \sqrt{2}$ for a categorical distribution p and the triangle inequality. Thus each draw accounts for the competition among the experts still available. The factor $1/\tau$ is part of the sampling likelihood; low temperature does not itself imply small routing gradients. For model parameters, the implementation applies the chain rule,

$$s_\theta(I) = (D_\theta a_\theta)^\top \nabla_a \log p_{a_\theta}(I), \tag{26}$$

where $D_\theta a_\theta$ is the selection-score Jacobian. The selection-score gradient equals the raw-logit gradient for softmax routers; sigmoid routers also require the sigmoid derivative, as given in Appendix D.

C.3 Group advantages and a trajectory interpretation

Proposition C.3 (Group centering preserves the on-policy direction)**.** *Fix a routing state and a reward law as in Proposition C.2. Draw $G > 1$ independent pairs $(I^{(g)}, R^{(g)})$ under p_θ and the fixed reward law, and let $\bar{R} = G^{-1} \sum_g R^{(g)}$. Then the estimator*

$$\widehat{g} = \frac{1}{G} \sum_{g=1}^{G} (R^{(g)} - \bar{R}) s_\theta(I^{(g)}) \tag{27}$$

satisfies Eq. (9)*.*

Proof. Normalization gives $\mathbb{E}[s_\theta(I)] = 0$, and Proposition C.2 gives $\mathbb{E}[R s_\theta(I)] = \nabla_\theta J$. For $j \neq g$, independence yields $\mathbb{E}[R^{(j)} s_\theta(I^{(g)})] = 0$. The self term in $\bar{R}$ is the only nonzero subtraction:

$$\mathbb{E}[(R^{(g)} - \bar{R}) s_\theta(I^{(g)})] = \nabla_\theta J - \frac{1}{G} \nabla_\theta J. \tag{28}$$

Averaging over g proves the result. □

For $G = 16$, the factor is $15/16$. This result uses the unstandardized group-mean advantage in Eq. (39); division by the sample reward standard deviation would introduce another random, reward-dependent factor. With $c = 1 - 1/G$ and $\bar{g} = \mathbb{E}[\widehat{g}]$, and assuming J is twice continuously differentiable near θ, the ordinary update along the expected direction satisfies

$$J(\theta + \eta \bar{g}) = J(\theta) + \eta c \|\nabla_\theta J(\theta)\|_2^2 + O(\eta^2). \tag{29}$$

Hence it is a local ascent direction whenever $\nabla_\theta J \neq 0$. This concerns the expected on-policy direction at a fixed state, not each sample or a finite Adam update.

Routing over an entire response. To account for dependent routing states without treating them as independent bandits, fix a reference model ϕ_0 and define a routing intervention. Its expert, attention, clean mixing, and token-generation functions remain those of ϕ_0: their parameters are fixed, but their outputs can change with the sampled history. Only the positive-temperature routing-selection kernels are parameterized by θ; let θ_0 denote their reference parameters in ϕ_0. Prompt-only routing and zero-temperature routing remain part of the fixed computation. Given a history H of tokens and executed routes, the hidden state $h_0(H)$ is computed by this reference model; it can change when the sampled history changes, but its computation has no explicit dependence on θ.

Assume a finite vocabulary and bounded response length. A response trajectory ζ has the factorization

$$\mathcal{P}_\theta(\zeta \mid x) = C_0(\zeta, x) \prod_{(t,l):\, \tau_l > 0} p_{\theta,l}(I_{t,l} \mid h_0(H_{t,l})), \tag{30}$$

where C_0 contains the fixed token and other transition probabilities, and the product ranges over response-predicting routing decisions. Let $\mathcal{D}$ be a fixed distribution over a finite set of prompts and $R(\zeta, x)$ a bounded reward independent of θ. Define

$$J_{\mathrm{route}}(\theta \mid \phi_0) = \mathbb{E}_{x \sim \mathcal{D}} \mathbb{E}_{\zeta \sim \mathcal{P}_\theta(\cdot \mid x)} [R(\zeta, x)].$$

Proposition C.4 (Policy gradient with frozen model computation)**.** *For the routing intervention above and any prompt-dependent baseline $b(x)$,*

$$\nabla_\theta J_{\mathrm{route}}(\theta \mid \phi_0) = \mathbb{E}_{\mathcal{D}, \mathcal{P}_\theta} \left[(R - b(x)) \sum_{t,l:\, \tau_l > 0} \nabla_\theta \log p_{\theta,l}(I_{t,l} \mid h_0(H_{t,l})) \right]. \tag{31}$$

For each prompt, draw G independent responses from $\mathcal{P}_{\theta_0}(\cdot \mid x)$. Under the experimental normalization in Appendix E.1, let M be the maximum response length and L the number of MoE layers. The unmasked routing update at θ_0 has expectation

$$-\mathbb{E}[\nabla_\theta \mathcal{L}_{\mathrm{RLRP}}]|_{\theta=\theta_0,\, m=1} = \frac{1-1/G}{ML}\, \nabla_\theta J_{\mathrm{route}}(\theta \mid \phi_0)|_{\theta=\theta_0}\,. \tag{32}$$

Here replay must reproduce the reference model's routing states, and the sampled actions, advantages, and hidden states are held fixed when differentiating the surrogate loss.

Proof. Differentiate the trajectory expectation and use $\nabla_\theta \mathcal{P}_\theta = \mathcal{P}_\theta \nabla_\theta \log \mathcal{P}_\theta$; the finite trajectory space justifies exchanging differentiation and expectation. The log of Eq. (30) has derivative equal to the sum of routing scores because C_0 is fixed. The baseline term vanishes since the conditional trajectory score has zero expectation for every prompt. The group-mean calculation in Proposition C.3 applies to each complete trajectory score, and the constant $1/(ML)$ gives Eq. (32). □

This construction matches the detached router branch at the rollout point: $\nabla_{W_l} \log p_{\theta,l} = (\nabla_o \log p_{\theta,l}) h_{t,l}^\top$ with $h_{t,l}$ held fixed. Here the raw-logit derivative includes the chain rule in Eq. (26) for the corresponding router type. The actual model shares parameters between routing selection, mixing, and hidden-state computation. Its full expected-reward derivative can also contain derivatives through earlier hidden states and other model components. The proposition characterizes the routing intervention, not that full shared-parameter derivative.

C.4 Token clipping, replay, and the scope of the guarantees

What the GRPO mask controls. For a replayed token with advantage A and token-probability ratio r, away from clipping boundaries the ascent gradient of the PPO/GRPO clipped surrogate is $mAr\,\nabla \log \pi_\phi$, with m given by Eq. (7). The routing branch instead uses $mA\,\nabla \log p_\theta$. It shares the active-token decision, but contains neither the token-ratio multiplier nor an ordered-action importance ratio. The mask is detached and recomputed from the current forward pass. If rollout and training probabilities agree at the reference policy, $r = 1$ and every nonzero-advantage token is active. This recovers the unmasked on-policy identities above.

For later updates, token clipping does not bound the routing-action ratio $p_\theta(I)/p_{\mathrm{old}}(I)$ or a router KL divergence. For example, if the experts produce identical outputs, the token ratio can remain one while the routing distribution changes. Thus sharing the token mask supplies no router trust-region or monotonic-improvement guarantee.

A bound separating clipping from distribution shift. A fixed-state calculation makes the two sources of deviation explicit. Let $\mu(I) = p_{\mathrm{old}}(I)$, keep $Q(S)$ and a baseline b fixed, and let $m(I) \in [0,1]$ be a detached mask. Define

$$g_{\mu,m}(\theta) = \mathbb{E}_\mu[m(I)(Q(S(I)) - b)s_\theta(I)], \qquad \delta_m = \mathbb{E}_\mu[1 - m(I)]. \tag{33}$$

Suppose $|Q(S) - b| \le A_{\max}$ and $\sup_I \|s_\theta(I)\|_2 \le B_\theta$. Then

$$\begin{aligned} \|g_{\mu,m}(\theta) - \nabla_\theta J(\theta)\|_2 &\le A_{\max} B_\theta \,(2\,\mathrm{TV}(\mu, p_\theta) + \delta_m) \\ &\le A_{\max} B_\theta \left(\sqrt{2 D_{\mathrm{KL}}(\mu \| p_\theta)} + \delta_m\right). \end{aligned} \tag{34}$$

Here $\mathrm{TV}(\mu,\nu) = \frac{1}{2}\sum_I |\mu(I) - \nu(I)|$. To prove the first inequality, set $f_\theta(I) = (Q(S(I)) - b)s_\theta(I)$, use Proposition C.2, and write

$$g_{\mu,m} - \nabla_\theta J = (\mathbb{E}_\mu - \mathbb{E}_{p_\theta}) f_\theta - \mathbb{E}_\mu[(1-m) f_\theta]. \tag{35}$$

The triangle inequality bounds the two terms by $2A_{\max}B_\theta \mathrm{TV}$ and $A_{\max}B_\theta \delta_m$. The second line of Eq. (34) follows from Pinsker's inequality. In selection-score coordinates, Eq. (25) permits $B_a = \sqrt{2}K/\tau$; for general parameters, a valid bound is $B_\theta = \|D_\theta a_\theta\|_{\mathrm{op}} \sqrt{2}K/\tau$. The same argument covers stochastic rewards and masks using their joint distribution with I, provided the conditional reward law is fixed, $|R - b| \le A_{\max}$, and b remains a fixed baseline.

The group-mean baseline. The sample mean used in our experiments depends on the sampled actions, so it requires a separate calculation. At the same fixed routing state, draw G independent action–reward pairs under μ and a fixed conditional reward law, with $R \in [r_{\min}, r_{\max}]$. Set $D_R = r_{\max} - r_{\min}$, $c = 1 - 1/G$, and

$$g_{\mu,m}^{(G)} = \mathbb{E}\left[\frac{1}{G}\sum_{g=1}^{G} m_g(R^{(g)} - \bar{R})s_\theta(I^{(g)})\right], \qquad \delta_G = \mathbb{E}\left[\frac{1}{G}\sum_{g=1}^{G}(1 - m_g)\right]. \tag{36}$$

The detached masks $m_g \in [0, 1]$ may depend on the whole group. Then

$$\begin{aligned}\|g_{\mu,m}^{(G)} - c\nabla_\theta J\|_2 &\le D_R B_\theta \left(2c\,\mathrm{TV}(\mu, p_\theta) + \delta_G\right) \\ &\le D_R B_\theta \left(c\sqrt{2D_{\mathrm{KL}}(\mu\|p_\theta)} + \delta_G\right).\end{aligned} \tag{37}$$

Indeed, independence gives $g_{\mu,1}^{(G)} = c\,\mathbb{E}_\mu[(Q(S(I)) - J_\mu)s_\theta(I)]$, where $J_\mu = \mathbb{E}_\mu[R]$. Apply the preceding distribution-shift argument with the fixed baseline J_μ, using $|Q(S) - J_\mu| \le D_R$. Removing the masks changes the direction by at most $D_R B_\theta \delta_G$, since $|R^{(g)} - \bar{R}| \le D_R$. This recovers the factor c when $\mu = p_\theta$ and all masks are one.

These bounds are local to a fixed routing state and a fixed reward law. They do not control hidden-state drift or simultaneous token-policy changes, and the token mask does not enforce a small routing-distribution shift. They identify when the ideal score-function direction is recovered, rather than proving that practical replay updates satisfy a trust-region condition.

Zero temperature and deployment. The positive-temperature likelihood does not extend to a differentiable stochastic policy at $\tau_l = 0$. The implementation sets that layer's auxiliary loss to zero while retaining its token-loss gradients; it still counts the layer in the fixed normalization L. The preceding reward identities concern stochastic routing during training. They do not imply monotonic improvement after switching to deterministic top-K deployment.

D Moonlight Routing Details

Sampling implementation. Moonlight samples from all $N = 64$ routed experts in a single group, without candidate truncation. The Gumbel noise in Eq. (5) is added to the corrected selection scores, after sigmoid. Perturbing raw logits before sigmoid would generally induce a different sampling law and would not match the ordered likelihood in Eq. (6).

Logit gradient. At a fixed routing state and positive temperature τ, let $p_\theta(I)$ denote the ordered likelihood in Eq. (6). Differentiating through $a = \sigma(o) + b$ with fixed correction bias b gives

$$\nabla_o \log p_\theta(I) = \mathrm{diag}(\sigma'(o))\frac{1}{\tau}\sum_{k=1}^{K}(e_{i_k} - p_a^{(k)}), \tag{38}$$

where $p_a^{(k)}$ normalizes $\exp(a/\tau)$ over the experts remaining at draw k, and e_{i_k} is the selected expert's one-hot vector. The ordered-action and group-mean identities in Appendix C remain applicable, while the controlled-deviation bound uses the boundary margin in the selection scores a.

E Implementation Details

E.1 RL Fine-tuning Details

We implement RL training in Slime (Zhu et al., 2025), using Megatron (Shoeybi et al., 2019) as the training engine and SGLang (Zheng et al., 2024) as the inference engine. Table 4 summarizes the Moonlight hyperparameters used for RL training. For reward calculation, we employ a rule-based correctness reward that depends only on whether the final answer matches the reference solution. Concretely, the model receives a reward of 1 for an exact match and 0 otherwise, with no additional

format-based reward components. Answer matching is implemented using the Math-Verify library[1]. We use a standard chain-of-thought (CoT) prompting template, shown in Table 5. The prompt content is kept the same across methods, and we preserve Moonlight's default chat template when formatting the final input sequence.

Objectives and loss normalization. We remove KL regularization and apply DAPO's Clip-Higher technique (Yu et al., 2025), with $\epsilon_{\text{low}} = 0.2$ and $\epsilon_{\text{high}} = 0.28$. GRPO supplies the token-policy objective, and RLRP shares its response-level advantages. For each prompt x, the rollout policy generates $G = 16$ responses $\{y^{(g)}\}_{g=1}^{G}$ with rewards $R^{(g)}$. Following Dr. GRPO (Liu et al., 2025b), we omit standard-deviation normalization and compute

$$A^{(g)} = R^{(g)} - \frac{1}{G}\sum_{j=1}^{G} R^{(j)}. \tag{39}$$

We set $A_t^{(g)} = A^{(g)}$ for all generated tokens and routing decisions in response $y^{(g)}$. Both objectives divide each response's summed token loss by the fixed maximum response length M, rather than its realized length, and average over prompts and their G responses. For RLRP, we additionally average over all L MoE layers. Thus the shorthand aggregation in Eq. (8) uses a denominator of GML per prompt. Clipped tokens and zero-temperature layers contribute zero without changing this denominator.

Expert sampling. Moonlight's two shared experts are always activated. Stochastic expert sampling and the routing-policy objective apply only to its routed experts. Inspired by dropout (Srivastava et al., 2014), we use a shallow-to-deep expert-temperature schedule during Moonlight training. Across its L MoE layers, we set $\tau_\ell = 0.01(\ell - 1)/(L - 1)$ for $\ell = 1, \ldots, L$, increasing linearly from 0 at the shallowest MoE layer to 0.01 at the deepest.

Table 4: **RL fine-tuning recipe.** The three methods in Figure 4 use the same training schedule of 100 rollout steps and 800 gradient steps. Batch sizes are measured in prompts.

Hyperparameter	**Value**
Optimization	
Optimizer	AdamW
Learning Rate	1×10^{-6}
LR Schedule	Constant
Weight Decay	0.0
Warm-up Training Steps	0
Batching and training	
Rollout Batch Size	256
Mini-batch Size	32
Number of Rollout Steps	100
Number of Gradient Steps	800
Rollout generation	
Max Rollout Length	4096
Number of Rollouts Per Prompt	16
Rollout Sampling Temperature	1.0
RLRP	
Expert Sampling Temperature (shallow to deep)	$0 \to 0.01$
Hardware	
GPU Model	NVIDIA RTX PRO 6000 Blackwell

E.2 Dataset and Evaluation Details

Training dataset. We train on the original DAPO-Math dataset (Yu et al., 2025), containing approximately 17k examples. DAPO-Math is constructed for reinforcement learning on mathematical

[1] https://github.com/huggingface/Math-Verify

Table 5: **Prompt template used for RL fine-tuning and evaluation.** The placeholder <question> is replaced with the actual question during fine-tuning and evaluation. Special tokens "<|im_start|>" and "<|im_end|>" are omitted for clarity.

user
<question>
Let's think step by step and output the final answer within \boxed{}.
assistant

reasoning tasks, where each problem is paired with an integer-form final answer, enabling rule-based reward computation.

Evaluation datasets. Table 1 and Figures 4 and 6 use the same seven benchmarks. Relative to the math-only RL training data, we treat mathematical reasoning as in-domain and academic reasoning (MMLU and MMLU-Pro) and commonsense reasoning (SIQA and WinoGrande) as out-of-domain. The sizes below refer to our evaluation files before prompt-length filtering.

- **Olympiad (in-domain).** We use the English, text-only mathematics problems from OlympiadBench (He et al., 2024). Our evaluation file contains 674 problems requiring advanced mathematical problem solving; it excludes the benchmark's physics and multimodal tasks.
- **AMC23 (in-domain).** AMC23 (Mathematical Association of America, 2023) is a 40-problem, answer-based evaluation set drawn from the 2023 AMC 12 competitions. It tests high-school mathematics, including algebra, geometry, and counting.
- **Minerva Math (in-domain).** We use the 272-question OpenCourseWare evaluation set introduced with Minerva (Lewkowycz et al., 2022). It contains undergraduate science and mathematics problems from MIT OpenCourseWare, requiring quantitative and symbolic reasoning.
- **MMLU (out-of-domain academic reasoning).** MMLU (Hendrycks et al., 2021) tests knowledge and problem solving across 57 academic and professional subjects, with four answer choices per question. We use its 285-question development split.
- **MMLU-Pro (out-of-domain academic reasoning).** MMLU-Pro (Wang et al., 2024) emphasizes harder, reasoning-focused questions across 14 subject areas, with up to ten answer choices. We use its 70-question validation split.
- **SIQA (out-of-domain commonsense reasoning).** Social IQa (Sap et al., 2019) tests commonsense understanding of everyday social interactions, including people's intentions and likely reactions. Each question has three answer choices. We use a fixed 100-example validation subset.
- **WinoGrande (out-of-domain commonsense reasoning).** WinoGrande (Sakaguchi et al., 2020) tests commonsense reasoning through sentence completion with two candidate referents. We use a fixed 100-example validation subset of the debiased release.

The SIQA and WinoGrande subsets are selected with seed 1234 and shared across methods.

Evaluation protocol. For the main evaluation, we report avg@16 for AMC23 and avg@3 for Olympiad, Minerva Math, MMLU, MMLU-Pro, SIQA, and WinoGrande. Here, avg@n is the mean correctness of n sampled responses per question, averaged over all questions in that benchmark. We then average the seven benchmark accuracies with equal weight. All accuracy results use a token sampling temperature of 1.0 and a maximum response length of 4,096 tokens, matching the rollout limit during training. Prompts longer than 512 tokens after chat formatting are excluded.

All models in Table 1 use Moonlight's default deterministic top-K routing ($K = 6$), selecting the experts with the largest sigmoid scores plus the expert correction bias. Expert sampling is disabled; method names refer to the training procedure. For inference-time K-expert sampling, we instead select K distinct experts using Eq. (4) with Moonlight's selection scores from Eq. (1). Appendix F.2 evaluates this alternative on the trained checkpoints.

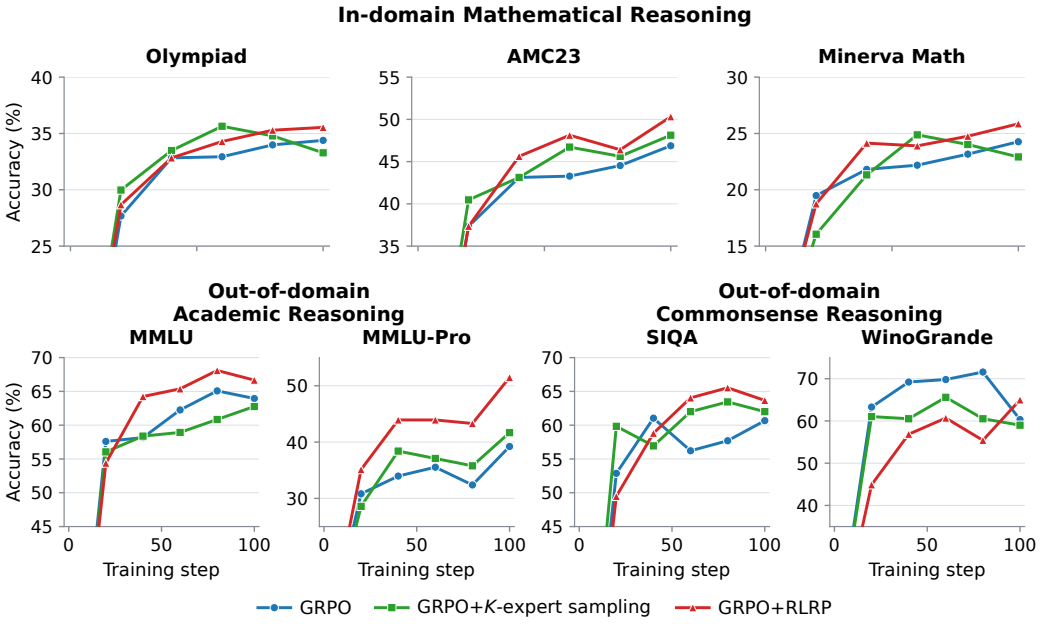

Figure 6: **Moonlight-16B-A3B accuracy curves on seven reasoning benchmarks.** We compare GRPO, GRPO+K-expert sampling, and GRPO+RLRP using the same runs and training checkpoints as Figure 4. The top row shows in-domain mathematical reasoning. The bottom row groups out-of-domain academic reasoning on the left (MMLU and MMLU-Pro) and out-of-domain commonsense reasoning on the right (SIQA and WinoGrande). Markers indicate training steps 20, 40, 60, 80, and 100. Lines connect points without smoothing.

Routing probes. Inference-only routing probes additionally use AIME25 (Mathematical Association of America, 2025), the 30 problems from the 2025 AIME I and II, with integer answers between 000 and 999. The temperature sweep in Figure 2 uses the frozen Moonlight-16B-A3B Base checkpoint. For the router-domain analysis in Figure 5, we keep each checkpoint fixed and collect router scores under deterministic top-K inference.

F Additional Experimental Results

F.1 Accuracy Curves on Individual Benchmarks

Figure 6 breaks down the mean accuracy in Figure 4 into the seven constituent benchmarks. All three methods are trained on DAPO-Math. The upper row tracks mathematical reasoning, while the lower row tracks academic and commonsense reasoning beyond the RL training domain.

The final evaluations in Table 1 show that GRPO+RLRP has the highest accuracy among the three methods on all seven benchmarks. The largest final gain over GRPO is on MMLU-Pro: 51.47% versus 39.22%, compared with 41.67% for GRPO+K-expert sampling.

F.2 Expert Sampling at Inference

Table 1 evaluates all models with deterministic top-K routing. Here we instead apply expert sampling at inference to the final GRPO, GRPO+K-expert sampling, and GRPO+RLRP checkpoints. Benchmarks, prompts, response counts, and decoding settings match Table 1.

Table 6 shows that inference-time expert sampling raises the seven-benchmark average for GRPO and GRPO+K-expert sampling by 0.19 and 1.81 percentage points, respectively. For GRPO+RLRP, it instead lowers the average from 51.22% with deterministic top-K routing to 50.04%, a decrease of 1.17 percentage points before rounding. Accuracy decreases on six of the seven benchmarks; only

Table 6: **Effect of inference-time expert sampling on Moonlight-16B-A3B.** For each method, we show deterministic top-K accuracies and accuracies with inference-time expert sampling using the $0 \rightarrow 0.01$ expert-temperature schedule. All values are in %; parentheses show absolute accuracy changes from the same method's top-K result. Avg. is the unweighted seven-dataset mean. Means and changes are computed before rounding. Boldface marks the highest accuracy in each column.

Inference routing	In-domain			Out-of-domain				Avg.
	Olympiad	AMC23	Minerva	MMLU	MMLU-Pro	SIQA	WinoGrande	
GRPO								
Top-K	34.39	46.88	24.26	63.95	39.22	60.67	60.33	47.10
Expert sampling	33.43 (−0.95)	45.47 (−1.41)	25.86 (+1.59)	64.18 (+0.24)	45.10 (+5.88)	58.33 (−2.33)	58.67 (−1.67)	47.29 (+0.19)
GRPO+K-expert sampling								
Top-K	33.28	48.13	22.92	62.77	41.67	62.00	59.00	47.11
Expert sampling	34.44 (+1.15)	45.78 (−2.34)	24.14 (+1.23)	64.54 (+1.77)	48.53 (+6.86)	**64.00** (+2.00)	61.00 (+2.00)	48.92 (+1.81)
GRPO+RLRP								
Top-K	**35.54**	**50.31**	25.86	**66.67**	**51.47**	63.67	**65.00**	**51.22**
Expert sampling	34.74 (−0.80)	49.69 (−0.63)	**27.08** (+1.23)	65.60 (−1.06)	48.53 (−2.94)	62.00 (−1.67)	62.67 (−2.33)	50.04 (−1.17)

Minerva improves. The largest drops are on MMLU-Pro (2.94 percentage points) and WinoGrande (2.33 percentage points).

Because sampling changes which experts are selected without increasing the active-expert budget, it can help some predictions and disrupt others. GRPO+RLRP still achieves the highest average under both routing conditions. These results suggest that training-time exploration can help learn better deterministic routing, while additional randomness at inference need not improve average response accuracy ($\mathrm{avg@}n$).

G Background on Group Relative Policy Optimization

Group Relative Policy Optimization (GRPO) is an RL post-training algorithm commonly used for improving the reasoning ability of large language models. Unlike actor–critic methods that use a learned value function, GRPO estimates advantages by comparing multiple responses sampled for the same prompt.

Given an input prompt x, the old policy π_{old} samples a group of G responses $\{y_i\}_{i=1}^{G}$. Each response is assigned a scalar reward R_i by a predefined reward function. The advantage of each response is computed by normalizing rewards within the group:

$$\hat{A}_i = \frac{R_i - \text{mean}(\{R_j\}_{j=1}^{G})}{\text{std}(\{R_j\}_{j=1}^{G})}. \tag{40}$$

Our experimental variant omits standard-deviation normalization and uses a fixed maximum-response-length normalizer, as detailed in Appendix E.1.

The standard token-policy loss is the negative PPO-style clipped objective:

$$\mathcal{L}_{\text{GRPO}} = -\mathbb{E}_{x,\{y_i\}} \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \min \left(r_{i,t} \hat{A}_i,\ \text{clip}\left(r_{i,t}, 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}}\right) \hat{A}_i \right) \right], \tag{41}$$

where

$$r_{i,t} = \frac{\pi_\theta(y_{i,t} \mid x, y_{i,<t})}{\pi_{\text{old}}(y_{i,t} \mid x, y_{i,<t})} \tag{42}$$

is the token-level importance ratio, and ϵ_{low} and ϵ_{high} control the clipping range.