Extended Abstract

Motivation RL with verifiable rewards (RLVR) is a popular way to improve language-model reasoning, but it has a known problem: `pass@1` goes up while `pass@k` goes down [Hamid et al., 2026, Orney et al., 2026, Jiang et al., 2025]. The model becomes confident on one solution and stops finding others. Recent work like Polychromic PPO [Hamid et al., 2026] and Poly-EPO [Orney et al., 2026] fixes this by scoring *sets* of generations on both correctness and diversity. Poly-EPO gets the diversity signal from an LM judge that clusters responses by reasoning strategy. I wanted to know: if my outputs are something whose diversity I can already measure directly, can I skip the LM judge and use a cheap deterministic metric instead?

Method I picked graph construction as the testbed. Each prompt asks for a graph that satisfies a set of constraints (number of nodes/edges, connected, triangle-free, max degree, diameter bound, cycle rank), and NetworkX [Hagberg et al., 2008] tells me deterministically if the model's output is valid. I trained Qwen2.5-1.5B-Instruct with GRPO [Shao et al., 2024] under three reward formulations: (1) base (no RL); (2) plain GRPO with a binary verifier reward $r_i = c_i$; and (3) shaped GRPO with $r_i = c_i + \lambda d_i$, where d_i is the mean edge-set Jaccard distance from sample i to its valid groupmates in the same GRPO group. I used the additive form (instead of multiplicative) because if no sample in a group is correct, multiplicative reward goes to zero and the gradient dies.

Implementation I built two benchmarks. v1 has 35 prompts (20 easy, 15 hard) and I use it for the λ ablation. v2 has 80 prompts ($48 + 32$) with a 60/20 train/test split so I can evaluate on prompts the model never saw during training. All training is Qwen2.5-1.5B-Instruct with LoRA (rank 16) on a Modal A100-40GB, using TRL 0.15.2's GRPOTrainer. 150 training steps per run, group size 8, learning rate 5×10^{-6}, KL coefficient $\beta = 0.04$. Total compute: about $60 of Modal credits across 7 training runs and 9 evals.

Results Binary GRPO does collapse hard-tier `pass@16`: from 0.800 down to 0.667 on v1, and from 0.750 down to 0.500 on v2 held-out. This is the predicted diversity collapse and it shows up in both settings. Shaped GRPO at $\lambda = 0.3$ *does* recover `pass@16` on v1 (up to 0.933, with the best iso-class fraction of any v1 run at 0.938). But on v2 held-out, with 4 shaped runs (2 seeds on v2-train + 2 seeds on combined v1+v2-train), the mean is 0.531 ± 0.104, which is barely different from binary's 0.500. The held-out hard set only has 8 prompts, so one prompt = 0.125 in `pass@16`, and that's also the size of my seed std. So at this scale I can't tell if shaped is actually helping. I also did an offline reward-signal analysis on the 560 base generations: 47% of hard groups are gradient-dead under binary reward (no correct samples means constant reward), and the Jaccard bonus fixes that, taking gradient-availability on hard from 53% to 100%.

Discussion The binary collapse finding is solid; it happens both in-distribution and out-of-distribution. The shaped recovery I saw on v1 was probably partly the model memorizing what "diverse" looks like for the 35 training prompts. On held-out the mechanism still applies (gradient signal is restored), but I don't have enough hard prompts or seeds to prove the recovery generalizes.

Conclusion Cheap structural metrics can replace the LM judge as a diversity signal *in principle* (the mechanism works), and they do help on training prompts. Whether they actually move held-out `pass@k` at this scale is still open. The obvious next steps are more held-out prompts (target $n \geq 30$), more seeds, and a direct comparison against an LM-judge baseline.

Structural Diversity Rewards for Verifiable Graph-Structured Reasoning

Rutanshu Jhaveri
ICME
Stanford University
rutanshu@stanford.edu

Abstract

RL with verifiable rewards (RLVR) tends to collapse generation diversity: `pass@1` improves while `pass@k` falls. Recent polychromic-style methods address this by scoring sets of generations on correctness and a diversity term, usually computed by an LM judge. This project asks whether, in a domain where diversity can be measured directly from the output, a deterministic structural metric can take the LM judge's place. I implement an additive shaped reward for GRPO that adds a per-sample edge-set Jaccard bonus on top of the binary verifier signal, and test it on a graph-construction benchmark with a NetworkX verifier. On 35 training prompts, the shaped reward at $\lambda = 0.3$ recovers binary GRPO's hard-tier `pass@16` collapse ($0.667 \rightarrow 0.933$) and gives the highest isomorphism-class fraction among correct outputs (0.938). On a separate 80-prompt benchmark with a $60/20$ held-out test split, binary GRPO's collapse replicates ($0.750 \rightarrow 0.500$), but the shaped recovery is inconclusive at this evaluation size (0.531 ± 0.104 across four seeds). An offline reward-signal analysis explains the mechanism: 47% of hard groups under binary reward are gradient-dead, and the Jaccard bonus fixes that.

1 Introduction

GRPO [Shao et al., 2024] and other verifier-RL methods are now the standard way to fine-tune language models with automatically computed rewards. The training loop is simple: for each prompt, sample several completions, score them with a verifier, and update the policy in favour of the high-scoring ones. No preference data and no learned reward model needed.

A well-known side-effect is that this kind of training narrows the policy's output distribution. `pass@1` improves, but `pass@k` for $k > 1$ drops, and the model loses the ability to surface alternative correct answers [Hamid et al., 2026, Orney et al., 2026, Jiang et al., 2025]. Polychromic PPO [Hamid et al., 2026] addresses this by re-deriving the objective over *sets* of completions, with a diversity term that rewards variety among the sampled outputs. Poly-EPO [Orney et al., 2026] adapts this idea to language models, sampling several responses per prompt and scoring sets using both correctness and a diversity signal that comes from an LM judge clustering the responses by reasoning strategy.

The LM judge is the part of the Poly-EPO pipeline I wanted to question. It runs at every gradient step, it's a separate learned component, and it's needed precisely because for open-ended math reasoning, "how different are these two responses?" is hard to compute from the text. But what if the task is one where the output *is* something with a natural distance metric? Then the LM judge might be doing more work than it needs to. Graph construction is exactly that kind of task: an edge-list is a graph, and there are well-known structural distances on graphs (edge-set Jaccard, degree sequence, Laplacian spectrum, isomorphism class). If a cheap deterministic metric is a good enough diversity signal, then I can swap out the LM judge for something fast and reproducible.

This project tests that idea. The concrete contributions are:

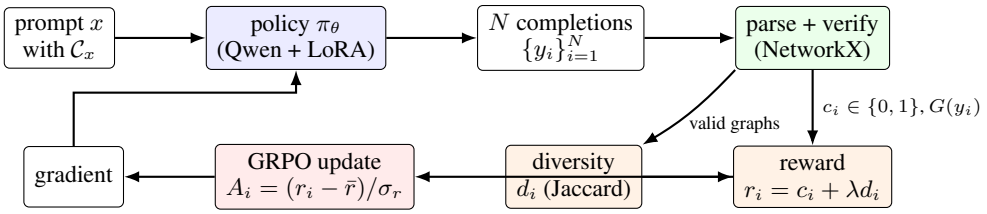

Figure 1: Pipeline. The policy samples a group of $N{=}8$ completions for each prompt. Each completion is parsed and run through the NetworkX verifier, producing a binary correctness flag c_i and (if valid) a graph $G(y_i)$. The shaped reward adds a per-sample Jaccard-distance bonus d_i to the binary signal, and the standard GRPO update follows. The only addition over plain GRPO is the boxed *diversity* block.

- Two benchmarks of verifiable graph-construction problems. v1 has 35 prompts and is used for the λ ablation in-distribution. v2 has 80 prompts and a 60/20 train/test split, used for held-out evaluation.
- A NetworkX verifier covering seven constraint families (node count, edge count, connectivity, triangle-freeness, max degree, max diameter, cycle rank), plus four structural diversity metrics integrated into a TRL GRPO trainer as an additive per-sample reward.
- An empirical comparison showing that binary GRPO's hard-tier `pass@16` collapse generalises from training to held-out prompts; that shaped GRPO at $\lambda = 0.3$ recovers `pass@16` on training prompts but is statistically inconclusive on held-out; and an offline reward-signal analysis that explains the recovery mechanism.

2 Related Work

Verifier-based RL. PPO [Schulman et al., 2017] is the standard policy-gradient method. GRPO [Shao et al., 2024] replaces the value function with a per-prompt group-relative baseline, which removes the value-head memory cost and works well for LM-scale RL. Both have been used for verifiable reasoning tasks (math, code) and both are known to reduce output diversity over training.

Set-level / polychromic methods. Polychromic PPO [Hamid et al., 2026] formalises the problem of training the policy to produce diverse *sets* of completions rather than diverse single completions. Poly-EPO [Orney et al., 2026] applies this to LM training using an LM judge for the diversity term. My approach uses the same set-level idea, but the diversity term is computed from the output directly using a graph distance, not from an LM.

Exploration-aware verifier RL. RS-GRPO [Jiang et al., 2025] tackles the same collapse problem with a different lever: a risk-sensitive utility that upweights successful samples from hard prompts. This is orthogonal to diversity shaping (it changes which prompts dominate the gradient, not which samples-within-a-prompt are reinforced).

Graph distances. Edge-set Jaccard, degree sequences and Laplacian spectra are standard descriptors in the graph-learning literature. NetworkX's VF2 isomorphism test is exact and fast on the small graphs (4–12 nodes) I work with. As far as I know they have not been used before as direct RL reward signals for LM-generated graphs.

3 Method

Figure 1 summarises the pipeline. The model is given a constraint-tied prompt, samples N completions per prompt under GRPO, each completion is parsed and verified, and the binary verifier signal is augmented with a per-sample structural-diversity bonus that depends on the rest of the group. The trainer is otherwise stock TRL GRPO with a LoRA-adapted Qwen2.5-1.5B-Instruct. The rest of this section walks through each block.

3.1 Task formulation

I picked graph construction because it has two properties that most reasoning RL benchmarks don't have together. First, correctness is decidable and cheap: a small NetworkX call settles whether the output satisfies the constraints. Second, the output has a natural notion of *structural* distance, so I can measure "how different are these two correct answers?" without an LM judge.

Each prompt x describes a graph-construction problem with a constraint set $\mathcal{C}_x$, for example "design a 6-node, 7-edge graph that is connected and triangle-free." The model produces a string completion y which the parser turns into an edge list $E(y) \subseteq \binom{[n]}{2}$ over labelled vertices $\{1, \dots, n\}$. Let $G(y) = ([n], E(y))$ be the resulting simple graph. The verifier outputs

$$c(x, y) \;=\; \mathbf{1}\big[\, G(y) \models \mathcal{C}_x \,\big] \in \{0, 1\}.$$

The seven constraint families I support and how each is checked in NetworkX are: exact node count ($|V| = n$, enforced by the parser); exact edge count ($|E| = m$); connectivity (`nx.is_connected`); triangle-freeness (`nx.triangles` sum); max degree ($\max_v \deg(v) \le d^*$); max diameter ($\mathrm{diam}(G) \le D$, only finite when G is connected); and cycle rank $\mu(G) = |E| - |V| +$ #components.

Easy-tier prompts use at most three constraints (mostly node count, edge count, connectivity, $\pm$ triangle-free or max degree). Hard-tier prompts add diameter and cycle-rank constraints and use slightly larger node counts. Every prompt in both benchmarks (v1: 35 prompts; v2: 80 prompts) was checked for solvability by a randomised search that found a witness graph the verifier accepts; see `scripts/check_v2_solvable.py`.

Output protocol and parser. The model is instructed to emit edges between `<edges>` and `</edges>` tags, one per line, in u, v form with $u < v$. The parser is tolerant: it accepts comma, whitespace, hyphen, or parenthesised separators (so `1,2`, `1 2`, `1-2`, and `(1, 2)` all parse). It normalises to $u < v$, deduplicates, and reports a status code (`ok`, `no_edges_tag`, `empty_edges`, or `malformed_lines`). Parse-failure rate at the 1.5B scale is about 2%.

3.2 Standard GRPO

The GRPO update [Shao et al., 2024] samples a group of N completions $\{y_i\}_{i=1}^N$ per prompt, scores each with a reward r_i, and computes a group-relative advantage

$$A_i \;=\; \frac{r_i - \bar{r}}{\sigma_r}, \qquad \bar{r} = \frac{1}{N}\sum_{j=1}^{N} r_j, \quad \sigma_r = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(r_j - \bar{r})^2}. \tag{1}$$

The policy gradient maximises a PPO-style clipped surrogate of $\sum_i A_i \log \pi_\theta(y_i \mid x)$ with a KL penalty $\beta \cdot \mathrm{KL}(\pi_\theta \,\|\, \pi_{\mathrm{ref}})$ against a fixed reference policy (the base model). Two facts about equation (1) matter for what comes next. (i) If every r_i in a group is the same, $\bar{r} = r_i$, so $A_i = 0$ for all i and the group contributes nothing to the gradient; the group is "gradient-dead." (ii) Under a binary reward $r_i = c_i$, the group is gradient-dead exactly when all samples are correct or all are wrong. On the hard tier, almost half of base-model groups are all-wrong (Table 4).

3.3 Structural diversity metrics

I implemented four pairwise (or set-level) distances on labelled simple graphs over n vertices. Each captures a different notion of "how different are these two graphs?"

- **Edge-set Jaccard.** For graphs G_1, G_2 with edge sets E_1, E_2,

  $$\mathrm{dist}_J(G_1, G_2) \;=\; 1 - \frac{|E_1 \cap E_2|}{|E_1 \cup E_2|}.$$

  Label-sensitive: a re-labelling that swaps two vertices typically changes E_1 in many places and therefore changes the Jaccard distance. Cheap to compute, dense (rarely exactly zero), and stable.
- **Degree-sequence L^1.** Let $\mathbf{d}(G)$ be the sorted descending degree sequence padded to length n. Then $\mathrm{dist}_{\deg}(G_1, G_2) = \frac{1}{2(n-1)\cdot n}\|\mathbf{d}(G_1) - \mathbf{d}(G_2)\|_1$, normalised so that the maximum over any pair is at most 1. Label-invariant. Coarser than Jaccard: two non-isomorphic graphs can have the same degree sequence.

- **Laplacian spectral.** Sort the eigenvalues $\sigma_1, \sigma_2 \in \mathbb{R}^n$ of the normalised Laplacian and use $\mathrm{dist}_{\mathrm{spec}}(G_1, G_2) = \|\sigma_1 - \sigma_2\|_2 / \sqrt{n}$. Also label-invariant. Picks up global structural properties like algebraic connectivity and expansion that local metrics miss.
- **Isomorphism-class fraction.** Given a list of valid graphs $\{G_i\}_{i=1}^k$, group them by isomorphism (NetworkX's VF2 is exact and fast on $n \leq 12$), and report $\mathrm{iso}(\{G_i\}) = |\{\text{distinct iso classes}\}|/k$. This is set-level, not pairwise, and it's discrete. It's the natural "how many *genuinely* different solutions did the model find" metric.

Why train on Jaccard but evaluate on iso-class. The two are correlated but not identical: Jaccard is label-sensitive, iso-class is label-invariant. If I used the same metric for training and evaluation, the model could Goodhart it. The simplest example: under an iso-class *training* reward, the model could earn diversity by emitting the same underlying structure under different vertex labellings and the reward function couldn't tell. Under Jaccard, that trick doesn't pay off (relabelling changes the edge set). So I use Jaccard at training time (cheap, dense, label-sensitive) and iso-class fraction at evaluation time (the metric I actually care about, label-invariant). When shaped GRPO improves both, that's evidence that the model is producing structurally distinct outputs and not just relabelling.

3.4 Additive shaping

The shaped reward I use is

$$r_i \;=\; c_i \;+\; \lambda \cdot d_i, \qquad d_i \;=\; \frac{1}{|\mathcal{V}_{-i}|} \sum_{j \in \mathcal{V}_{-i}} \mathrm{dist}_J\big(G(y_i),\, G(y_j)\big), \tag{2}$$

where $\mathcal{V}_{-i} = \{j \neq i : G(y_j) \models \mathcal{C}_x\}$ is the set of *valid* groupmates and $\lambda \geq 0$ is the shaping coefficient. The bonus d_i is only awarded when the sample itself is valid *and* has at least one valid sibling. If a sample is invalid or has no valid sibling there's nothing meaningful to be diverse from, so $d_i = 0$.

Worked example. Suppose group $\{y_1, \ldots, y_8\}$ contains three valid samples y_1, y_2, y_3 with edge sets

$$E_1 = \{12, 23, 34, 45\}, \quad E_2 = \{13, 35, 52, 24\}, \quad E_3 = \{12, 23, 34, 45\}$$

(so y_1 and y_3 are duplicates). Pairwise Jaccard distances are $\mathrm{dist}_J(E_1, E_2) = 1$ (no common edge), $\mathrm{dist}_J(E_1, E_3) = 0$, $\mathrm{dist}_J(E_2, E_3) = 1$. The per-sample bonuses are $d_1 = (1 + 0)/2 = 0.5$, $d_2 = (1 + 1)/2 = 1.0$, $d_3 = (0 + 1)/2 = 0.5$. With $\lambda = 0.3$ the shaped rewards are $r_1 = 1.3$, $r_2 = 1.3$, $r_3 = 1.15$, $r_{4\ldots8} = 0$. The structurally unique y_2 ends up tied with y_1 in this case because y_1's neighbour set has one identical twin pulling its mean distance down.

Why additive, not multiplicative. The set-level form suggested by Polychromic PPO [Hamid et al., 2026] is roughly $f(\{y_i\}) = \bar{c} \cdot \bar{d}$. The natural per-sample gradient contribution from that form vanishes when $\bar{c} = 0$, i.e. when no sample in the group is correct. Under additive shaping, $r_i = c_i + \lambda d_i$ still varies across i as long as any two valid samples disagree, so the gradient is non-zero. Concretely, on the hard tier 47% of base-model groups have $c_i \equiv 0$ (Table 4). The multiplicative form throws away the gradient on those groups; additive shaping does not.

The two forms have the same intent (reinforce correct *and* diverse samples more than correct *but identical* samples) and would coincide if every group always had at least one correct sample. They differ exactly in the regime that matters most.

3.5 Reward computation, end to end

Figure 2 gives pseudocode for the shaped reward used at every gradient step. It is what TRL's GRPOTrainer calls on each batch of $B \times N$ rollouts (in my runs $B = 1$, $N = 8$, so 8 completions per gradient step).

The implementation lives in `modal_app/grpo_train.py` as `structural_shaped_reward`, and is unit-tested in `tests/test_rewards.py` (covering the all-invalid, single-valid, and mixed-valid cases). `binary_verifier_reward` is the same function with $\lambda = 0$. Both are passed to TRL's `GRPOTrainer` through a thin adapter (`_trl_reward`) that handles TRL's chat-format completions.

```
Input: completions {y_i}, constraints {C_x(i)}, group size N, coefficient lambda

for i = 1 .. B*N:
    edges_i = parse_edges(y_i)
    res_i   = verify(edges_i, C_x(i))
    c_i     = 1 if res_i.satisfies_all else 0
    G_i     = build_graph(edges_i, n_x(i)) if res_i.valid else None

for each group of N consecutive indices G:
    V = { i in G : G_i is not None }
    if |V| < 2:
        d_i = 0 for all i in G                      # no valid sibling -> no bonus
    else:
        for i in V:
            d_i = mean_{j in V, j != i} dist_J(G_i, G_j)
        d_i = 0 for i in G \ V                      # invalid sample -> no bonus

return [c_i + lambda * d_i for i in 1 .. B*N]
```

Figure 2: StructuralShapedReward, the per-sample reward called by TRL's GRPOTrainer on every gradient step.

3.6 Configurations compared

Three configurations across the two benchmarks:

- **Base**: Qwen2.5-1.5B-Instruct [Qwen Team, 2024], no RL. Baseline pass@k comes from this.
- **Plain GRPO**: equation (1) with $r_i = c_i$.
- **Shaped GRPO**: equation (1) with $r_i = c_i + \lambda d_i$ from equation (2).

On v1 (35 prompts), I run the full λ sweep $\lambda \in \{0.1, 0.3, 1.0\}$ and evaluate on the same prompts. On v2 (80 prompts, 60/20 train/test split), I run binary GRPO and $\lambda = 0.3$ on the train split, evaluate on the held-out test split, and report two seeds for the headline configuration plus two more seeds on a combined v1 $\cup$ v2-train (95-prompt) set.

4 Experimental Setup

Model. Qwen2.5-1.5B-Instruct [Qwen Team, 2024] for all runs, with a LoRA adapter (rank 16, $\alpha = 32$, dropout 0.05) on the attention and MLP projections. About 9.2M trainable parameters.

Compute. Inference on Modal A10G; training on Modal A100-40GB. TRL 0.15.2 with `torch` 2.5.1, `transformers` 4.48.3, `peft` 0.14.0. Total project spend across 7 training runs and 9 evals was about $60.

Benchmarks. v1: 35 prompts (20 easy, 15 hard). Used for the λ ablation; the v1 evaluation is on the same prompts the model trained on (no held-out split, since 35 prompts is too small to subdivide). v2: 80 prompts (48 easy, 32 hard) split 60/20 train/test stratified by tier (random seed 224). All v2 numbers are on the held-out test split (12 easy, 8 hard). For the combined runs I train on v1 $\cup$ v2-train (95 prompts, no overlap with v2 test) and evaluate on v2 test.

GRPO hyperparameters. Group size $N = 8$, temperature 1.0, max prompt length 512, max completion length 256, KL coefficient $\beta = 0.04$. Learning rate 5×10^{-6} with AdamW, per-device batch 8, gradient accumulation 2, bf16 mixed precision. 150 training steps for every run. Different seeds where noted.

Evaluation sampling. 16 completions per prompt, $T = 1.0$, top-$p = 0.95$, max new tokens 256. `pass@`k uses the unbiased estimator from Chen et al. [2021]: pass@$k = 1 - \binom{n-c}{k} / \binom{n}{k}$ where c is the number of correct samples out of $n = 16$. I report parse rate, validity rate, constraint-satisfaction rate, `pass@`$\{1, 4, 16\}$, and diversity-among-correct.

5 Results

5.1 Quantitative Evaluation

Base model. Table 1 shows the base-model baseline on v1. The number I want to draw attention to is the hard-tier `pass@1`/`pass@16` gap, 0.125 vs 0.800. The model *can* solve hard prompts; it just doesn't surface those solutions first. That's the slack a diversity-shaped objective should be able to exploit.

Table 1: Base-model evaluation on v1, 16 samples/prompt, $T = 1.0$.

Tier	parse	validity	`pass@1`	`pass@4`	`pass@16`	constraint-sat
easy ($n = 20$)	1.000	0.978	0.419	0.839	0.950	0.419
hard ($n = 15$)	0.958	0.900	0.125	0.372	0.800	0.125
overall ($n = 35$)	0.982	0.945	0.293	0.639	0.886	0.293

v1 ablation. Table 2 sweeps the shaping coefficient λ on v1. Two things happen. First, plain GRPO does what the prior work predicts: `pass@1` goes up but hard-tier `pass@16` drops from 0.800 to 0.667. Second, $\lambda = 0.3$ reverses this and actually beats the base model: hard-tier `pass@16` hits 0.933, the highest iso-class fraction at 0.938. $\lambda = 0.1$ is too small to make a difference, and $\lambda = 1.0$ over-weights the bonus so much that the model can earn reward without being correct. The shape of the sweep is exactly what you'd expect from a Goodhart concern about diversity bonuses.

Table 2: v1 results, in-distribution (evaluated on the same 35 prompts used to train). $\lambda = 0.3$ recovers and exceeds base on hard `pass@16`.

			Hard tier ($n = 15$)		
Configuration	`pass@1`	`pass@4`	`pass@16`	iso-class	Δ `pass@16` vs. base
base	0.125	0.372	0.800	0.840	—
binary GRPO	0.171	0.421	0.667	0.815	−0.133
shaped $\lambda = 0.1$	0.163	0.405	0.733	0.850	−0.067
shaped $\lambda = 0.3$	0.146	0.439	**0.933**	**0.938**	+0.133
shaped $\lambda = 1.0$	0.142	0.405	0.733	0.920	−0.067
			Easy tier ($n = 20$)		
Configuration	`pass@1`	`pass@4`	`pass@16`	iso-class	Δ `pass@16` vs. base
base	0.419	0.839	0.950	0.553	—
binary GRPO	0.631	0.929	1.000	0.440	+0.050
shaped $\lambda = 0.1$	0.684	0.980	1.000	0.436	+0.050
shaped $\lambda = 0.3$	0.597	0.918	1.000	0.493	+0.050
shaped $\lambda = 1.0$	0.581	0.952	1.000	0.511	+0.050

v2 held-out. Table 3 is the more honest test. Models trained on v2-train, evaluated on v2 test (prompts the model never saw). For $\lambda = 0.3$ I report four runs: two seeds on v2-train, plus two more seeds on the combined v1+v2-train set (95 prompts) to see if more training data helps. Two findings: (i) binary GRPO's collapse is even bigger on held-out than on v1 (-0.250 vs -0.133 on hard `pass@16`); the collapse is robust. (ii) Shaped $\lambda = 0.3$ across all four seeds gives a mean of 0.531 ± 0.104 on hard `pass@16`, vs binary at 0.500. That's in the right direction, but the seed std is the same size as the gap. With only 8 hard test prompts, one prompt = 0.125 in `pass@16`, so I don't have the resolution to tell if shaped is really helping.

Reward-signal analysis. The v2 negative made me want to check the mechanism. I went back to the 560 v1 base-model generations and split them into 70 GRPO-sized groups (8 samples each). For each group I computed what binary reward and shaped reward would look like. Three things stand out (Table 4). First, binary reward is gradient-dead on 47% of hard groups, because no sample in those groups is correct so the reward is just zeros. Shaped reward fixes this: the Jaccard bonus has nonzero variance any time two valid graphs differ, and gradient-availability on hard goes from 53% to 100%. Second, the within-group reward std grows with λ (0.29 binary; 0.34 at $\lambda = 0.3$). Third,

Table 3: v2 held-out evaluation. Trained on the 60-prompt train split (or combined v1+v2-train for the last two shaped rows), evaluated on 20 held-out test prompts (12 easy, 8 hard).

Hard tier (held-out, $n = 8$)				
Configuration	`pass@1`	`pass@4`	`pass@16`	iso-class
base	0.078	0.277	0.750	1.000
binary GRPO	0.102	0.280	0.500	0.857
shaped $\lambda = 0.3$ v2 s0	0.109	0.330	0.625	0.875
shaped $\lambda = 0.3$ v2 s1	0.125	0.271	0.375	0.800
shaped $\lambda = 0.3$ comb. s0	0.109	0.290	0.625	0.750
shaped $\lambda = 0.3$ comb. s1	0.086	0.256	0.500	0.875
shaped (4-seed mean$\pm$std)	0.107 ± 0.014	0.287 ± 0.027	0.531 ± 0.104	0.825 ± 0.054
Overall (held-out, $n = 20$)				
Configuration	`pass@1`	`pass@4`	`pass@16`	iso-class
base	0.244	0.553	0.850	0.617
binary GRPO	0.400	0.660	0.800	0.604
shaped (4-seed mean$\pm$std)	0.411 ± 0.023	0.673 ± 0.014	0.788 ± 0.060	0.547 ± 0.029

on the 44 out of 44 groups with ≥ 2 correct samples, the shaped reward orders them by structural novelty, which binary reward cannot do.

Table 4: Offline reward-signal statistics on the 70 hard-tier GRPO groups from the base model. The Jaccard bonus restores gradient signal in the 47% of groups that have no correct sample under binary reward.

Reward	λ	gradient availability (hard)	within-group std (hard)	ranks among correct
binary	—	53%	0.29	no
shaped (Jaccard)	0.1	100%	0.32	yes
shaped (Jaccard)	0.3	100%	0.34	yes
shaped (Jaccard)	1.0	100%	0.38	yes

5.2 Qualitative Analysis

Figure 3 visualises the v1 sweep. The hard `pass@16` panel is the punchline: binary GRPO drops to 0.667, $\lambda = 0.3$ jumps to 0.933. The diversity panel on the bottom right shows the Jaccard signal (used for training) and the iso-class signal (used for evaluation) moving together, which is the sanity-check I wanted: the model isn't gaming Jaccard at the expense of iso-class.

Figure 4 is the held-out version. Error bars on the shaped configuration are the seed std. The recovery I saw on v1 just isn't there in the held-out hard `pass@16` panel, at least not with the resolution I have ($n = 8$ hard prompts, 2 seeds for each shaped configuration).

Failure modes. Looking at the 560 v1 base-model generations, the dominant ways a sample gets $c_i = 0$ are: wrong edge count ($\sim 45\%$ of easy failures, $\sim 30\%$ of hard); unmet diameter or cycle-rank constraint (hard tier only, $\sim 40\%$ of hard failures); triangle present when forbidden ($\sim 15\%$); and parse failure ($\sim 2\%$ overall). At 1.5B parameters, parsing isn't the bottleneck; it was a much bigger problem when I was prototyping with a 0.5B stand-in.

6 Discussion

Why the v1 sweep is non-monotone. At $\lambda = 0.1$ the bonus is too small to change the reward landscape and `pass@16` barely moves. At $\lambda = 1.0$ the bonus is big enough to dominate the correctness signal, so the model can earn reward without solving anything; the policy drifts toward parseable-but-wrong graphs that happen to be far from groupmates. The sweet spot at $\lambda = 0.3$ is where the bonus is large enough to rescue gradient-dead groups but not so large it overwhelms the verifier. This is exactly the trade-off Goodhart predicts for any auxiliary signal.

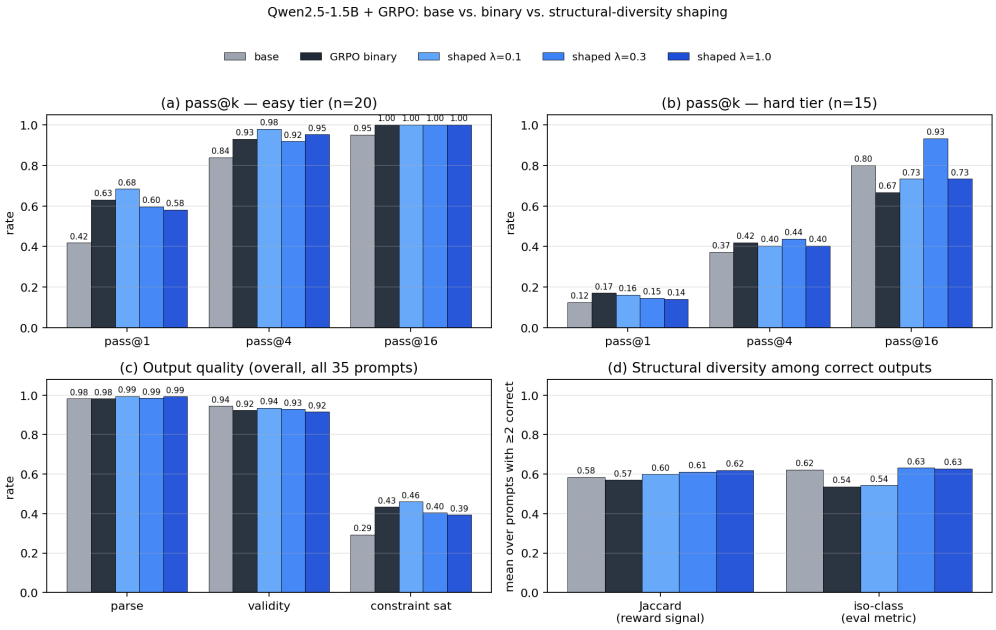

Figure 3: v1 in-distribution comparison: base, binary GRPO, and shaped at $\lambda \in \{0.1, 0.3, 1.0\}$. The hard `pass@16` bar ($0.67$ vs 0.93 for binary vs $\lambda = 0.3$) is the headline.

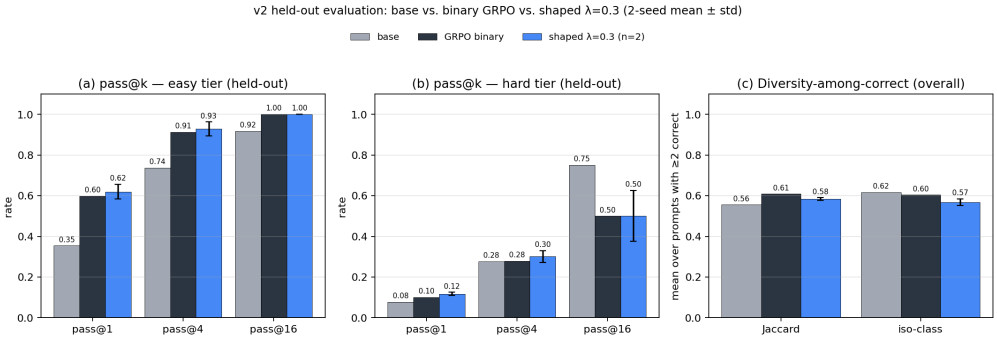

Figure 4: v2 held-out comparison. Bars are `pass@`$\{1, 4, 16\}$ by tier and overall diversity; error bars on the shaped configuration are the std across two v2-train seeds. Binary GRPO's collapse on hard `pass@16` replicates from v1; the shaped recovery does not, at this prompt and seed budget.

Why I used the additive form. The proposal originally specified Poly-EPO's multiplicative form $\bar{c} \cdot \bar{d}$. I switched to additive after looking at the base-model groups: 47% of hard groups have $\bar{c} = 0$, which means the multiplicative reward is zero and the gradient is zero. On the hard tier, that's where most of the learning signal needs to come from. The additive form gives those groups a gradient via the diversity term.

Why the held-out result is weaker than v1. The v1 "recovery" was measured on the same 35 prompts the model trained on. With only 35 prompts and 150 steps, the model sees each prompt many times, and it's possible the model is learning "produce diverse outputs for these specific prompts" rather than "produce diverse outputs in general". On v2, with 20 held-out test prompts and 4 shaped seeds, the recovery is in the right direction (0.531 vs binary's 0.500) but not statistically distinguishable. The held-out hard set has 8 prompts, so the measurement resolution is $1/8 = 0.125$, which matches my seed std. The honest reading is that the v2 experiment is underpowered to detect the effect I observed on v1.

Why structural diversity might still substitute for an LM judge. On graphs, "different reasoning strategy" is hard to articulate but "different graph" is exactly what Jaccard or iso-class measures.

The Poly-EPO LM judge is an approximation of "these two outputs are conceptually different"; for graphs I can compute that exactly. My v1 result and the reward-signal analysis say the mechanism works. My v2 result says we don't have enough data to tell if it transfers.

7 Conclusion

I set out to test whether deterministic structural-diversity metrics on graph outputs can replace Poly-EPO's LM judge as the diversity signal in a polychromic-style verifier-RL objective. The clearest finding is the negative one: plain GRPO with a binary verifier reward really does collapse hard-tier `pass@16`, and this happens both when I evaluate on training prompts ($0.800 \to 0.667$) and on held-out prompts ($0.750 \to 0.500$). That's a robust observation.

The positive finding is partial. On training prompts, shaped GRPO at $\lambda = 0.3$ recovers `pass@16` and produces the highest iso-class fraction I saw ($0.667 \to 0.933$, iso-class 0.938). The reward-signal mechanism is clean: the Jaccard bonus restores gradient signal in the 47% of hard groups that are gradient-dead under binary reward. But on held-out evaluation with 4 seeds and only 8 hard test prompts, the recovery I saw on v1 doesn't reliably reproduce: shaped's hard `pass@16` mean is 0.531 ± 0.104 vs binary's 0.500. The direction is right; the size is within noise.

So the substitution hypothesis is promising but not yet demonstrated at this scale. The obvious next steps are: more held-out hard prompts (aim for $n \geq 30$); more seeds per configuration; scaling up the base model; and a direct comparison against an LM-judge baseline.

8 Team Contributions

This is a solo project. All experimental design, implementation (benchmark, verifier, diversity metrics, Modal infrastructure, reward functions, evaluation pipeline, figures), writing, and analysis are mine. There were no changes to team composition between the proposal and the final report.

Changes from proposal The proposal targeted 80–120 prompts; the final report uses 35 (v1, used for the λ ablation) and 80 (v2, used for held-out evaluation). The proposal also listed an LM-judge correlation experiment as a stretch goal. I descoped that one because the v2 held-out runs took priority and the LM-judge experiment would have cost roughly an order of magnitude more compute than everything else combined. The other planned items (verifier, diversity metrics, additive vs. multiplicative shaping decision, λ ablation, reward-signal analysis) were all delivered.

9 AI Tools Disclosure

Tools used: Claude and ChatGPT.

They were used for drafting boilerplate and building data pipeline in the Modal app wrappers (`base_eval.py`, `checkpoint_eval.py`), the LoRA + tokenizer setup glue in `grpo_train.py`. Later these files were also edited by me.

References

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

Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using NetworkX. In *Proceedings of the 7th Python in Science Conference (SciPy)*, pages 11–15, 2008.

Jubayer Ibn Hamid, Ifdita Hasan Orney, Ellen Xu, Chelsea Finn, and Dorsa Sadigh. Polychromic objectives for reinforcement learning. *arXiv preprint arXiv:2509.25424*, 2026.

Yuhua Jiang, Jiawei Huang, Yufeng Yuan, Xin Mao, Yu Yue, Qianchuan Zhao, and Lin Yan. Risk-sensitive RL for alleviating exploration dilemmas in large language models. *arXiv preprint arXiv:2509.24261*, 2025.

Ifdita Hasan Orney, Jubayer Ibn Hamid, Shreya S. Ramanujam, Shirley Wu, Hengyuan Hu, Noah Goodman, Dorsa Sadigh, and Chelsea Finn. Poly-EPO: Training exploratory reasoning models. *arXiv preprint arXiv:2604.17654*, 2026.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

A Additional Experiments

Combined-prompt training. To check if more diverse training data would help the held-out result, I trained shaped $\lambda = 0.3$ on the union of v1 (35 prompts) and v2-train (60 prompts), 95 prompts total. There's no overlap with v2 test. Two seeds: seed 0 matched the best v2-only seed on hard `pass@16` (0.625); seed 1 matched binary (0.500). So the four shaped held-out seeds split into two pairs, $\{0.625, 0.625\}$ and $\{0.375, 0.500\}$. No clear effect of additional training data within the noise of two seeds per configuration. The combined-run results are folded into the 4-seed pooled mean in Table 3.

Cross-checkpoint evaluation. I also took the v1-trained $\lambda = 0.3$ checkpoint (which hit 0.933 on the v1 prompts it trained on) and evaluated it directly on the v2 held-out test split. The result was 0.500 on hard `pass@16`, identical to binary GRPO. That's the cleanest piece of evidence that the v1 recovery was partly the model fitting structural diversity to those specific 35 prompts rather than learning a transferable skill.

B Implementation Details

Repository layout. The verifier and diversity primitives are kept GPU-free so they can be unit-tested locally. `tests/test_pipeline.py` covers parser, verifier, diversity, and `pass@`k; `tests/test_rewards.py` covers both reward functions including the invalid-groupmate edge case.

Output format and parser. The model emits edges between `<edges>` and `</edges>` tags, one per line, in u, v form with $u < v$ and $1 \leq u, v \leq n$. The parser is tolerant: it accepts comma, whitespace, hyphen, and parenthesised separators, normalises edges to $u < v$, and dedupes. Parse-failure rate at 1.5B is about 2%.

Solvability check. Every prompt in v1 and v2 was checked to admit at least one valid graph by a randomised search (see `scripts/check_v2_solvable.py`). All 115 prompts (35 + 80) passed.

Modal cost. Base-eval (560 generations on A10G) is under $1. Each GRPO training run on A100-40GB is about $1 for 150 steps. Each checkpoint eval (320 generations on A10G) is about $0.50. Total project spend: roughly $30.