Feature-Space Curriculum Learning for Countdown Reasoning

Extended Abstract

Vishaal Saraiya (saraiyas@stanford.edu)

Motivation. Reinforcement-learning (RL) post-training of large language models typically samples training problems uniformly or by coarse difficulty proxies such as loss or perplexity. These signals are *coarse*; a scalar does not say *which* structural property of a problem makes it hard, and hence inherently looses some contextual information. These signals are also *static*; the difficulty is usually estimated once, even though difficulty is a property of the *current* model and shifts as it learns. The project proposes that the difficulty is both *model-state-dependent* and *structurally legible*, and that a curriculum should exploit both facts.

Method. The project proposes a **Feature-Space Curriculum** (FSC) that (i) periodically probes the current policy to measure its **edge of learnability** from rollout success, and (ii) describes that edge in an interpretable, **Difficulty Vector** (DV) of task structure (operand count, shortest-solution structure, valid-solution count, operator usage). FSC resamples training toward the model's current frontier and can generalize from observed edge/failure examples to structurally similar problems through DV-region weighting or a DV-space kernel.

Implementation. FSC is implemented as a sampling layer on top of an RLOO loop for the Countdown arithmetic task using an SFT-tuned Qwen2.5-0.5B policy. At every refresh interval, the cuirriculum logic probes a random set, recomputes per-example difficulty, and rebuilds a weighted sampling distribution; unobserved examples carry a neutral prior placed exactly at the edge target. All FSC variants share matched RLOO hyperparameters and objective function, so the curriculum is the only major variable.

Results. Firstly, dynamically recomputing the edge during training beats freezing an initialization-time edge by roughly $+7.5$ points pass@1 and $+5$–7 points pass@8/pass@16, with the scoring rule held fixed. Second, DV structure is itself a strong sampling signal: a frozen DV-region curriculum improves pass@8 by $+4.3$ and pass@16 by $+6.0$ over a frozen uniform control (and pass@16 by $+10.0$ over a frozen per-example edge), isolating feature-space weighting from dynamic recomputation. The DV also localizes the edge interpretably, to high shortest-solution operand count and low valid-solution-count regions. Importantly, in addition to the core ideas of th eproposal, the project combines the dynamic edge and the DV lead to the best results in the study.

Discussion. The project's results validate the proposal that recomputing model-dependent difficulty over time matters; probe snapshots show the edge migrating toward harder structure as the model learns. The results also show that interpretable task structure carries an independent, and the project's strongest, high-k signal in the static case. Combining the two yields the best pass@1 in the study; the remaining high-k gap of the joint DV-space kernel is traced to an equal-weight, single-bandwidth distance and is a tuning opportunity (parameter selection, correlation-aware weights, feature subsets) rather than a structural limit.

Conclusion. FSC shows that a moving, feature-space description of the edge of learnability is both an effective curriculum and an interpretability tool, and that its two ingredients contribute independently: dynamic recomputation of the edge supplies the pass@1 gains, while interpretable DV structure supplies the strongest high-k signal, with their combination best overall. The benefit is leverage rather than coverage (a 128-prompt probe every 10 steps leaves the sampling distribution nearly uniform yet lifts usable RLOO gradient signal by $\sim 69\%$) so the curriculum is essentially free to run. Because the edge is read out in human-legible task structure (high shortest-solution operand count, low valid-solution count), the same signal that drives sampling also explains *where* the model is learning and reframes difficulty as a measurable, moving property of the policy rather than a fixed label. This directly motivates the natural next step: conditioning data *generation*, not just resampling, on the measured edge profile.

Feature-Space Curriculum Learning for Countdown Reasoning

Vishaal Saraiya
Department of Computer Science
Stanford University
saraiyas@stanford.edu

Abstract

Reinforcement-learning post-training of language models usually samples problems uniformly or by coarse scalar difficulty, signals that are both uninformative about *which* structure makes a problem hard and static with respect to the learning model. The paper introduces a Feature-Space Curriculum (FSC) that probes the current policy to measure its edge of learnability from rollouts and describes that edge in an interpretable, Difficulty Vector (DV). Built on an RLOO loop for the Countdown task with an SFT-tuned Qwen2.5-0.5B policy, FSC resamples training toward the current frontier. With curriculum the only varied factor, dynamically recomputing the edge beats freezing the edge to its initial value by $+7.5$ pass@1 and $+5$–7 pass@8/16; and a frozen DV-region curriculum beats a frozen uniform control by $+6.0$ pass@16 (two-seed means) and a frozen per-example edge by $+10.0$ pass@16, isolating feature-space structure from dynamic recomputation. Fresh probe snapshots show the edge migrating toward harder structure as easy problems are mastered, and curriculum combing both the proposed methods (dynamic edge and structural/contextual information) attain the best pass@1 in the study. Strikingly, this curriculum is nearly free: its only fresh, unbiased signal is a 128-prompt probe every 10 steps (the per-step train-batch updates reuse RLOO rollouts at no extra cost, $\sim 3\%$ example coverage), and although it leaves the sampling distribution essentially uniform (effective-sample fraction $\approx 99.5\%$) it still raises usable RLOO gradient signal by $+69\%$, evidence that the gains are *leverage* rather than coverage. The paper analyzes why an additive DV policy currently leads a joint DV-space kernel at high k, argue the remaining gap is a tuning rather than a structural limitation, and outline correlation-aware weighting and feature-conditioned data generation as next steps.

1 Introduction

Curriculum learning (Bengio et al., 2009) argues that ordering training from easy to hard improves convergence and final quality. Recent reasoning work refines this to the *edge of learnability*: models gain the most from problems just beyond their current mastery, and easy-to-hard schedules improve LLM reasoning (Parashar et al., 2025; Chen et al., 2025). However, the difficulty signals used to drive such curricula have two weaknesses. First, they are **coarse**: a scalar loss or perplexity does not indicate *which* structural property of a problem makes it hard. Second, they are **static**: difficulty is typically estimated once, even though it is a property of the *current* model and shifts as the model learns.

Thesis. Difficulty is model-state-dependent and structurally legible. FSC therefore (i) remeasures the student's edge from rollouts as it trains, and (ii) represents that edge in a Difficulty Vector

of interpretable, solver-derived task features. Rollout performance identifies *which* examples are currently at the edge; the DV says *where* that edge sits in task-feature space.

Task. The paper studies Countdown (Gandhi et al., 2024; Pan et al., 2025): given a target integer and a small set of numbers (3–4 numbers here), produce an equation inside `<answer>...</answer>` that uses each provided number at most once and evaluates to the target. Scoring is 0.0 for no answer tags, 0.1 for a present-but-incorrect answer, and 1.0 for a valid, correct answer. The base policy is an SFT-tuned Qwen2.5-0.5B (Qwen Team, 2024); FSC is added as a curriculum layer on an RLOO training loop (Ahmadian et al., 2024), extending the standard SFT $\rightarrow$ IPO (Azar et al., 2023) $\rightarrow$ RLOO pipeline. Countdown is a convenient testbed because an exact solver provides ground-truth structural features and verifiability.

Contributions. The paper advances *two* central claims, established with controlled comparisons in which the curriculum is the only varied factor:

- **Immediate edge feedback helps.** Remeasuring the edge of learnability *during* training beats a delayed/frozen (`static_once`) or absent estimate (Section 5.1.2); probe snapshots further show the edge *migrating* through feature space as the model learns (Section 5.2). This rests on an analytical basis: for the un-normalized RLOO advantage with the non-binary Countdown reward $\{0, c, 1\}$, the gradient-norm-optimal success rate is $p^\star = \frac{1}{2} - \frac{c}{2}m$ (with partial-credit mass m), which recovers the binary $p = 0.5$ result as $c, m \to 0$, so the edge the curriculum chases is provably the maximum-signal region (Section 3.5, App. E).
- **DV-style feature selection informs the curriculum.** An interpretable, solver-derived Difficulty Vector is an independent sampling signal: a DV-region curriculum lifts high-k accuracy even with the distribution frozen, isolating task structure from dynamic recomputation (Section 5.1.3).

Supporting these, the project contributes **FSC**, the method realizing both claims (probe-measured edge described in DV coordinates, with several fusion policies), and analyses that explain *why* they hold: an analytical characterization of the gradient-norm-optimal edge target for non-binary RLOO rewards (Section 3.5), the migrating-edge profile, a training-diet account of the pass@1-vs-pass@16 split (Section 5.2), and evidence that the two signals compose to the best pass@1 in the study.

2 Related Work

Curricula and automatic task selection. Curriculum (Bengio et al., 2009) and self-paced (Kumar et al., 2010) learning order training from easy to hard, and a large body of RL work *automates* this ordering rather than fixing it a priori. Easy-to-hard schedules improve LLM reasoning (Parashar et al., 2025); teacher–student curricula learn a policy over tasks from the student's learning progress (Matiisen et al., 2019); prioritized level replay reweights levels by a TD-error/value-loss proxy for their learning potential (Jiang et al., 2021); and self-evolving curricula treat problem categories as bandit arms and adapt the schedule online from rollout-derived absolute advantage (Chen et al., 2025). These methods share a *scalar* view of a task: each problem is summarized by a single number—a difficulty rank, a replay score, a learning-progress estimate—that induces a total ordering. FSC departs from this by describing the edge in an *interpretable, multi-dimensional* feature space (the Difficulty Vector) rather than as a scalar ordering, so the curriculum is simultaneously adaptive, human-readable, and able to say *which* structural property makes a problem hard rather than only *how* hard it is.

The edge of learnability and its signal. The principle that intermediate-difficulty prompts carry the most learning signal is, for *binary* rewards, well supported: Chen et al. (2025) show the GRPO (Shao et al., 2024) normalized absolute advantage $2\sqrt{p(1-p)}$ peaks at pass rate $p = 0.5$, Bae et al. (2025) lower-bound a reverse-KL learnability term by $p(1-p)$ under the KL-regularized objective, and Foster et al. (2025) use success variance $p(1-p)$ as a learnability heuristic under PPO (Schulman et al., 2017)/VinePPO (Kazemnejad et al., 2024). These results are each tied either to the GRPO *normalized* advantage or to the regularized objective and assume a strictly binary reward; the optimum for the *un-normalized* RLOO leave-one-out estimator under *non-binary* reward is left uncharacterized, which is the regime FSC operates in and which the paper treats analytically in Section 3.5, recovering the binary $p = 0.5$ result as a special case. FSC also differs in *what* it does with this signal: rather

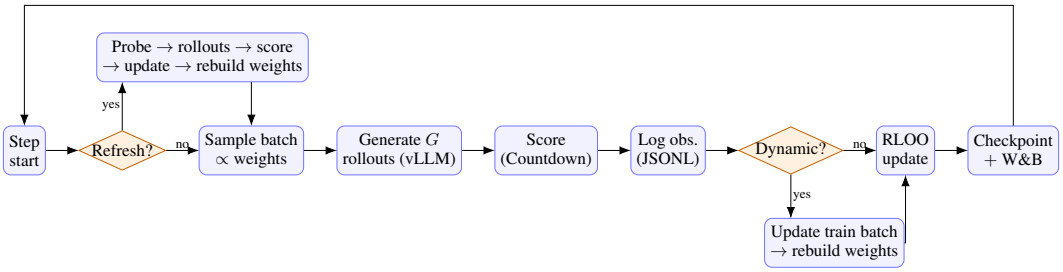


Figure 1: **FSC per-step loop**. Each step optionally probes the current policy (`dynamic` every N steps, `static_once` only at step 0), recomputes per-example difficulty and rebuilds the sampling weights, draws a training batch from the curriculum distribution, generates grouped rollouts, scores and logs them, updates difficulty from the batch (`dynamic` mode), and performs the RLOO update. Algorithm 1 states the same loop precisely.

than choosing a single target pass rate, it locates the maximum-signal region in feature space and resamples toward it.

RL post-training and gradient leverage. The underlying optimizer is standard pure-RL reasoning training: DeepSeek-R1 (DeepSeek-AI, 2025) and Kimi k1.5 (Moonshot AI, 2025) reward correct final answers, and REINFORCE-style estimators such as RLOO (Ahmadian et al., 2024) make this practical at small scale. Several systems exploit the fact that all-correct and all-wrong groups contribute no gradient. Closest in spirit, DAPO's dynamic sampling (Yu et al., 2025) *filters out* prompt groups whose rollouts are all-correct or all-wrong (the zero-variance, zero-gradient prompts quantified in Section 5.2), and online difficulty filtering (Bae et al., 2025) similarly keeps only mid-difficulty prompts. FSC instead *reweights* toward the measured edge and describes it in interpretable DV coordinates rather than hard-filtering degenerate groups, which lets it raise usable RLOO gradient signal while leaving the sampling distribution nearly uniform (Section 5.2)—evidence that the gains are *leverage* rather than coverage.

Difficulty estimation, generation, and the Countdown testbed. Countdown (Gandhi et al., 2024; Pan et al., 2025) is a standard small-scale testbed for verifiable reasoning because an exact solver supplies both ground-truth structural features and a reliable correctness signal, which is what makes a solver-derived Difficulty Vector cheap to compute. Process reward models and step-level verification estimate difficulty or correctness with *learned* models (Zhang et al., 2025b; Lightman et al., 2023), whereas FSC uses cheap solver-derived structural features together with rollout success, avoiding a learned difficulty model while remaining interpretable. On the generation side, self-play and teacher-generation methods create training data from delayed downstream rewards (Zhang et al., 2025a; Wang et al., 2025); most closely, SOAR (Sundaram et al., 2026) learns a teacher generator of stepping-stone problems through a bilevel meta-RL loop rewarded only by the student's *delayed, downstream* improvement on a fixed target set. FSC is complementary: it measures the current student's frontier *directly* in interpretable DV space rather than inferring it through delayed teacher reward, and it leaves feature-conditioned *generation* around that measured frontier as the natural next step (Section 6).

3 Method

3.1 Problem formulation

Let $\mathcal{D} = \{x_i\}$ be the training problems and π_θ the policy. RLOO draws a batch, samples a group of G rollouts per prompt, and updates θ with leave-one-out advantages. FSC replaces uniform batch sampling with a curriculum distribution $p_t(x) \propto \max(\text{score}_t(x), 0) + \epsilon$, where score_t is a policy-dependent difficulty score recomputed over training. FSC reuses the same RLOO objective and optimizer, altering only the distribution from which training prompts are sampled. Algorithm 1 makes the per-step loop precise; `RebuildWeights` recomputes score_t for *all* examples under the active policy (Section 3), so a small number of fresh observations (from the periodic probe and, in `dynamic` mode, from *every* training batch) reshape the whole sampling distribution.

Algorithm 1: FSC per-step training loop (one RLOO step).

Require : policy π_θ; problems $\mathcal{D}$ with static DV features; per-example sampling weights $w \in \mathbb{R}^{|\mathcal{D}|}$ with $w_i \propto \max(\text{score}_t(x_i), 0) + \epsilon$ over all $x_i \in \mathcal{D}$ (the sampling distribution p_t); step t; mode $\in \{\texttt{static}, \texttt{static\_once}, \texttt{dynamic}\}$

if $t \bmod N = 0$ (`dynamic`) or $t = 0$ (`static_once`) **then**
  draw probe $P \sim \text{Uniform}(\mathcal{D})$; roll out π_θ on P; score
  update dynamic metadata (`fail_rate`, `pass_at_k`, ...) from P
  $w \leftarrow$ `RebuildWeights()` `// recompute and normalize` w `over all` $x \in \mathcal{D}$
sample batch $B \sim \text{Categorical}(w)$ with replacement `//` $\Pr(x_i) \propto w_i$
generate G rollouts per prompt in B (vLLM); score; append observation rows
if mode $=$ *`dynamic`* **then**
  update dynamic metadata from B; $w \leftarrow$ `RebuildWeights()`
$\theta \leftarrow$ RLOO update on B (leave-one-out advantages, entropy, KL)

3.2 Difficulty Vector

Following the proposal, the DV is $V = \{n, m, u, nps, s\}$: cardinality n, magnitude m, operator uniqueness u, search-space sparsity nps, and operator signs s. It is realized as deterministic, precomputed static fields, including `num_count`, the solver-derived `shortest_operand_count` and `shortest_expression_depth`, `all_numbers_required`, and the valid-solution counts `nps_capped`/`nps_log1p` obtained from an exact `Fraction`-based subset solver. These fields are intrinsic to the problem and stable across training.

3.3 Dynamic performance metadata

Recomputed from grouped rollouts per example, the dynamic metadata includes `reward_mean`, `fail_rate`, and `pass_at_k`. These are kept separate from the task-structure DV: model-behavior fields describe what the model *wrote*, not what the problem *requires*.

3.4 Edge of learnability

The edge score is a Gaussian peaked at a target success rate,

$$\text{edge}(x) = \exp\left(-\tfrac{1}{2}\left(\frac{s(x) - \tau}{\sigma}\right)^2\right), \tag{1}$$

with edge target $\tau = 0.5$ and temperature $\sigma = 0.25$; examples the model solves about half the time score highest.

3.5 Why the edge target maximizes the learning signal

The choice $\tau = 0.5$ is not arbitrary: it is, to leading order, the success rate that maximizes the RLOO update magnitude, which gives the edge policy an analytical justification rather than a purely heuristic one. For a prompt rolled out G times with rewards $r_1, \ldots, r_G$ (mean $\bar{r}$), the leave-one-out advantage (which is computed here with no per-group standard-deviation normalization, unlike GRPO) simplifies to a scaled, mean-centered reward,

$$a_i = r_i - \frac{\sum_{j \neq i} r_j}{G - 1} = \frac{G}{G - 1}\,(r_i - \bar{r}), \tag{2}$$

so the size of the step a sample produces scales with $|a_i| \propto |r_i - \bar{r}|$. Averaging this over a prompt's rollouts, the prompt's overall learning signal is the *typical* size of its advantages, the mean absolute deviation $\text{MAD} = \mathbb{E}|r - \mu|$ of the rewards. Both summaries aggregate over the whole group rather than reacting to a single largest advantage, since the policy gradient sums one term per rollout. The paper uses the reward variance $\text{Var}(r) = \mathbb{E}[a^2]$ as a convenient proxy: it is easier to work with and is in fact the more rigorous of the two. The two track each other because both are measures of reward spread, each is zero exactly when all rewards are equal, each grows as

the rewards spread out around their mean, and (as shown below) both are maximized by the same prompts, so ranking prompts by one is essentially the same as ranking them by the other. Treating the per-sample score directions $\nabla \log \pi_i$ as roughly uncorrelated, the expected squared per-prompt step is $\mathbb{E}\|\sum_i a_i \nabla \log \pi_i\|^2 \approx \sum_i a_i^2 \, \mathbb{E}\|\nabla \log \pi_i\|^2 \propto \sum_i a_i^2 \propto \mathrm{Var}(r)$, so the variance is the leading-order expected squared update magnitude. A prompt that every rollout solves, or that every rollout fails, has $\mathrm{MAD} = \mathrm{Var}(r) = 0$ and contributes *zero* gradient no matter how often it is sampled, so sampling weight cannot rescue a degenerate prompt.

For a strictly binary reward with success probability p, both signals are monotone in $p(1-p)$ ($\mathrm{MAD} = 2p(1-p)$, $\mathrm{Var} = p(1-p)$) and peak at $p = 0.5$, exactly where Eq. (1) does; this is the binary edge-of-learnability result (Chen et al., 2025; Bae et al., 2025). Countdown rewards are $\{0, c, 1\}$ with $c = 0.1$, not binary. Writing the partial-credit mass as $m = \Pr(r = c)$ and maximizing the faithful MAD signal over the full-success rate at fixed m gives the objective-correct target

$$p^\star = \tfrac{1}{2} - \tfrac{c}{2}\, m = 0.5 - 0.05\, m, \qquad \mu^\star = \mathbb{E}[r] = \tfrac{1}{2} + \tfrac{c}{2}\, m, \tag{3}$$

so the optimal full-success rate sits just below 0.5 and the optimal mean reward just above it. (The variance proxy instead places the optimum at mean reward exactly $\frac{1}{2}$, i.e. $p = 0.5 - cm$; the two share the leading term and differ only at $\mathcal{O}(cm)$.) Because $c = 0.1$ is small the correction $0.05\, m$ is negligible, so the implementation simply targets $p = 0.5$ in Eq. (1): the derivation *justifies* the implemented edge target as the small-c limit of the exact RLOO optimum rather than as something tuned to it. App. E gives the full derivation. Selecting edge examples is thus, mechanically, selecting the maximum-gradient examples: the curriculum steers mass toward the prompts the RLOO objective can actually learn from. (Consistent with $c > 0$ shrinking the attainable signal, the empirically observed edge-band variance 0.20 (the 0.3–0.7 fail-rate band, Table 6) sits just below the binary-reward maximum 0.25.)

Note that the implemented target exact rather than approximate. The curriculum does not steer the ternary reward directly: difficulty is measured on a *binarized* signal, the dynamic feature $\texttt{fail\_rate} = \Pr(r < 1)$, which counts the partial-credit value c as a failure, so every edge, failure-rate, and DV-kernel policy reweights prompts by the Bernoulli full-success rate $p = \Pr(r{=}1)$ rather than by $\mu = \mathbb{E}[r]$. For that binarized quantity the optimum is exactly $p = 0.5$ (the $m = 0$ case above), so the $\mathcal{O}(cm)$ ternary correction never enters the quantity the curriculum actually controls: the edge target is exact by construction, not merely the small-c limit (App. E).

3.6 Curriculum modes

The project implements three modes:

1. `static`: never probes (uniform weights; the no-curriculum control).
2. `static_once`: probes once at step 0, computes weights from the *initial* model, then freezes them.
3. `dynamic`: re-probes every refresh interval and also updates from each training batch's rollouts, so weights track the moving edge.

Concretely, in `dynamic` mode `RebuildWeights` runs after *every* training batch (not only at the N-step probe), so the distribution adapts each step: the probe supplies the only *unbiased*, uniformly-drawn refresh, while the per-step train-batch updates are curriculum-sampling-biased but *free*, i.e. those rollouts already exist for the RLOO update and do not add any compoutational cost. note that unobserved examples carry a *neutral prior* (`fail_rate` $= 0.5$), which sits exactly at the edge target, so unprobed examples receive near-peak edge weight. Dynamic mode steadily replaces these priors with measurements; `static_once` cannot, which is part of the reason it underperforms dynamic edge.

3.7 Curriculum policies

Scores are combinable and normalized into a sampling distribution. The implementation uses four:

1. `failure_rate`: weights by observed failure.
2. `edge`: uses Eq. (1).

3. `feature_failure`: the first of two feature-space policies, it treats the DV *additively*, i.e. it bins each DV field, estimates a per-bin failure rate, and averages over fields,

$$\mathrm{ff}(x) = \frac{1}{F} \sum_{f=1}^{F} g_f(\mathrm{bin}_f(x)) \,. \tag{4}$$

4. `edge_dv_kernel`: the second feature-space policy, it treats the DV *jointly*, i.e. it selects observed edge/failure anchors and upweights examples near them with a Gaussian in normalized DV space, which factorizes into a product of per-feature kernels,

$$\exp\left(-\tfrac{1}{2b^2} \sum_f d_f^2 \right) = \prod_f \exp\left(-\tfrac{d_f^2}{2b^2} \right) , \tag{5}$$

so an anchor influences an example only when it is close in *all* features at once, representing the feature interactions that additive scoring cannot. An optional *density correction* divides each score by the local kernel mass $\sum_j K(x_i, x_j)$, converting the kernel-weighted sum into a density-normalized average so the policy follows anchor difficulty rather than the density of DV space (analyzed in Section 5).

3.8 Fusion, scheduling, and correlation-aware weighting

Policies can be fused (multiplicatively, by weighted average, by split-batch sampling, or gated on observation status), scheduled across phases, and optionally weighted per-feature by each field's observed |Spearman| with failure rate (an Automatic Relevance Determination style lengthscale that down-weights low-signal dimensions in Eqs. (4)–(5)).

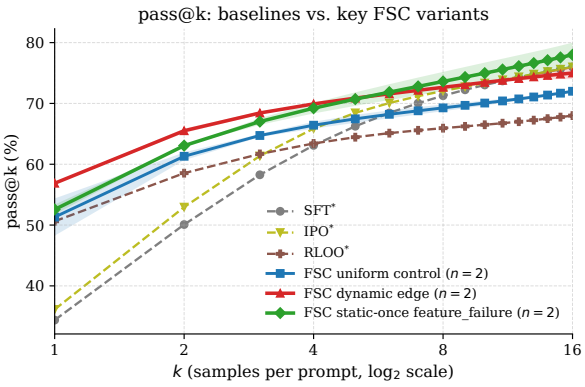
4 Experimental Setup

Model, data, pipeline. The base policy is an SFT-tuned Qwen2.5-0.5B, trained with RLOO on a public 3–4-number Countdown dataset. FSC inherits matched RLOO defaults so that the curriculum is the only major variable: batch size 128, group size 8, 100 training steps, learning rate 10^{-5} (constant), gradient accumulation 128, entropy and KL coefficients 10^{-3}, and temperature/top-p $= 1.0$. Curriculum settings: probe size 128, refresh interval 10, edge target 0.5, edge temperature 0.25, sampling $\epsilon = 0.05$.

Evaluation. Pass@k using the unbiased estimator $1 - \binom{n-c}{k} / \binom{n}{k}$ with $n = 16$ samples per problem, averaged over 50 shared test problems, at $k \in \{1, 8, 16\}$ is used in the evaluation. Because the eval set is small, the paper treats differences below roughly 10 points pass@16 as requiring multiple seeds; the paper reports paired comparisons on the shared problems and flag single-seed claims explicitly.

The number of seeds, and hence the number of multi-run comparisons, was constrained primarily by the cost of compute rather than by design. This was compounded by repeated out-of-memory (OOM) and runtime crashes on the fused and kernel runs (`fsc_032`, `fsc_041`, and `fsc_042`), which forced single-run configurations and consumed a substantial share of the budget. `fsc_032` and `fsc_041` each crashed before the fixed 100-step horizon on early attempts but were re-run to completion (`fsc_032` via its `r002` seed; `fsc_041` after a Ray crash), and the numbers reported for them are taken from those completed 100-step checkpoints; `fsc_042` (weighted-average fusion) never reached the horizon and yields no usable results, so it is omitted. Because all reported numbers are taken at the 100-step checkpoint, only fully completed runs enter the analysis. The combined effect is that several comparisons rest on single seeds, which are flagged in the interest of full disclosure.

Choosing the appropriate control. The correct internal control for FSC is a **uniform FSC control** (the `static` mode run through the same pipeline), not the RLOO baseline. FSC samples prompts *with replacement* from a weighted distribution, whereas the RLOO baseline uses a largely without-replacement shuffled dataloader; each prompt is visited at most once *within* an epoch, but examples recur *across* the multiple epochs of a run. Different prompt multiplicities yield different gradient trajectories. All curriculum claims below are therefore made against the uniform FSC control and matched static/dynamic conditions.



Shaded band: ±1 s.d. across training seeds (*n* given in legend). * single seed only; additional seeds not run due to compute budget.

Figure 2: pass@k for the baselines and key FSC variants on the shared 50×16 Countdown eval set. Curves are seed means and shaded bands are ± 1 s.d. across training seeds (n in legend); single-seed runs ($*$) lacked replicates owing to the project's compute budget. Dynamic edge lifts pass@1 well above RLOO, while a frozen DV-region curriculum (`feature_failure`) reaches the highest pass@16.

Diagnostics and excluded runs. The trainer logs *counterfactual distances*: at each curriculum refresh the implementation forms, from the *same* model state, the alternative sampling distributions it *would* have used under reference policies (uniform sampling and an edge-only policy that targets the learnability frontier but ignores feature/DV structure) without ever training on them, and measures how far the live (*active*) curriculum departs from each. The primary distance the paper relies on is total variation, $\mathrm{TV}(p, q) = \frac{1}{2} \sum_i |p_i - q_i| \in [0, 1]$, between the active weight vector and each reference (active-vs-uniform and active-vs-edge), together with a normalized weight entropy that flags whether the distribution actually concentrates. The normalized weight entropy is $H(w)/\log N$, where $H(w) = -\sum_i w_i \log w_i$ is the Shannon entropy of the normalized per-prompt weight vector w over the N prompts; it equals 1.0 when the weights are uniform and falls toward 0 as mass collapses onto a few prompts. A fuller battery of secondary diagnostics (Jensen–Shannon divergence, weight rank correlation, batch-level set overlap, and effective sample size) and the per-example observation logs used for offline DV-correlation analysis are described in Appendix D.

These diagnostics underwrite the central *leverage-not-coverage* finding. The dynamic-edge policy keeps its sampling distribution essentially uniform (normalized weight entropy ≈ 1.00 and effective-sample fraction $\approx 99.5\%$, i.e. near-zero total variation from uniform) yet it raises usable RLOO gradient signal by $+69\%$. Because coverage barely changes, the gains cannot be attributed to broadly resampling a different set of prompts; what drives them is the concentration of gradient-bearing mass, not the breadth of coverage.

Early runs trained with incorrect parameters are excluded from every claim, as are runs that crashed before the 100-step horizon (the weighted-average fusion `fsc_042`). For the dynamic-vs-static comparison the frozen per-example-edge reference is the completed `fsc_028`, whose solution-DV edge scoring matches the dynamic `fsc_027` exactly, so recomputation is the only varied factor. The earlier basic static-once-edge run `fsc_022` uses a different (non-solution-DV) edge score; it completed two seeds and is reported in Appendix C for completeness, but it is not the matched control.

5 Results

The project establishes the right control, shows dynamic $>$ static, isolates DV structure with three frozen runs, and combines the two ideas in multiple ways. Figure 2 shows pass@k curves for the baselines and the main FSC variants; Table 1 summarizes the headline numbers.

Table 1: Headline pass@k (%) on the shared 50×16 eval set. Replicated runs report the seed mean (per-seed values in Appendix C). The uniform FSC control is the internal baseline; RLOO/SFT/IPO are reference points.

Method / run	pass@1	pass@8	pass@16
SFT (reference)	34.38	71.27	76.00
IPO (reference)	36.12	72.05	76.00
RLOO (reference)	50.62	65.93	68.00
FSC uniform control (internal baseline)	51.31	69.27	72.00
FSC dynamic edge (`fsc_027`, 2-seed)	56.88	72.65	75.00
FSC static-once edge (`fsc_028`)	49.38	67.37	68.00
FSC static-once feature_failure (`fsc_031`, 2-seed)	52.56	**73.61**	**78.00**
FSC `edge_dv_kernel` (`fsc_032` r002)	56.38	70.82	72.00
FSC edge×feature_failure (`fsc_040`)	57.75	73.38	74.00
FSC split-batch fusion (`fsc_041`)	**57.88**	71.90	74.00

Table 2: Controlled comparison: identical edge scoring, only recomputation differs.

Condition	pass@1	pass@8	pass@16
Dynamic edge (`fsc_027`, 2-seed)	56.88	72.65	75.00
Static-once edge (`fsc_028`)	49.38	67.37	68.00
Difference (dynamic − static-once)	$+7.50$	$+5.28$	$+7.00$

5.1 Quantitative Evaluation

The control is not a result. The uniform FSC control scores $51.31/69.27/72.00$ versus RLOO's $50.62/65.93/68.00$. The small gap reflects sampling-with-replacement pipeline variance, not a curriculum effect, which is why all curriculum claims are made against this control.

5.1.1 R1: Dynamic edge > uniform control.

Against the designated internal baseline, dynamic edge (`fsc_027`) beats the uniform FSC control by $+5.57$ pass@1, $+3.38$ pass@8, and $+3.00$ pass@16 (Table 1, 2-seed means). This is the curriculum effect; the edge curriculum moves pass@1 from 51.3 to 56.9 over the same pipeline with weighting flattened, despite having a tiny probe budget (See Section 5.2).

5.1.2 R2: Dynamic > static.

Holding the scoring rule fixed and varying only whether the edge is recomputed, dynamic solution-DV edge (`fsc_027`) beats static-once solution-DV edge (`fsc_028`) by $+7.50$ pass@1, $+5.28$ pass@8, and $+7.00$ pass@16 (Table 2, Figure 3). Dynamic edge lifts pass@1 to $\sim$57 without sacrificing the high-k ceiling, partly escaping RLOO's diversity collapse; fusions (`fsc_040`/`fsc_041`) reach the study maximum at $\sim$58.

5.1.3 R3: Static + DV > static.

To isolate the DV contribution from dynamic recomputation, three *frozen* runs are compared. A static-once DV-region curriculum (`feature_failure`, `fsc_031`) beats both the frozen uniform control by $+4.3$ pass@8 and $+6.0$ pass@16, and a frozen per-example solution-DV edge (`static_once edge`, `fsc_028`) by $+6.2$ pass@8 and $+10.0$ pass@16 (Figure 4). The two replicated arms (`feature_failure` and the uniform control) are reported as two-seed means; the frozen-edge reference (`fsc_028`) is a single completed seed. Because all three arms are static, this isolates DV structure as the cause: upweighting hard feature regions generalizes to structurally similar examples that were never probed. The two `feature_failure` seeds bracket pass@16 at $80.0/76.0$ (mean 78.0), a spread within training-seed noise.

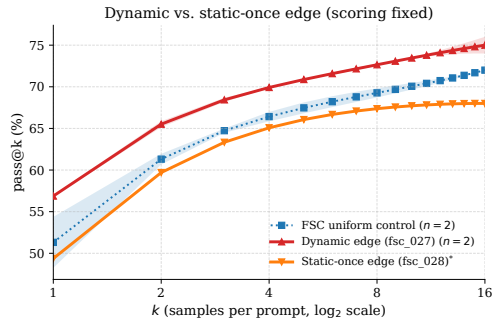


Figure 3: **Result R2 (dynamic $>$ static).** With the edge scoring rule held fixed, recomputing the edge during training (dynamic, `fsc_027`) dominates a frozen initialization-time edge (`fsc_028`) at every k; the uniform control is shown for reference. Curves are seed means with ± 1 s.d. bands (n in legend); the static-once edge ($*$) is single-seed (compute budget).

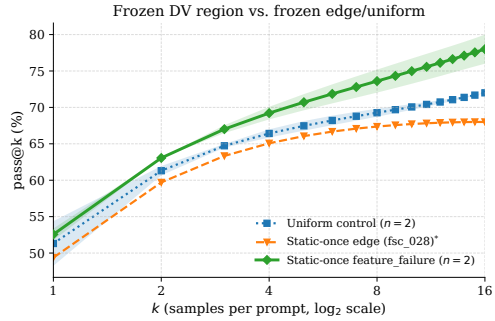


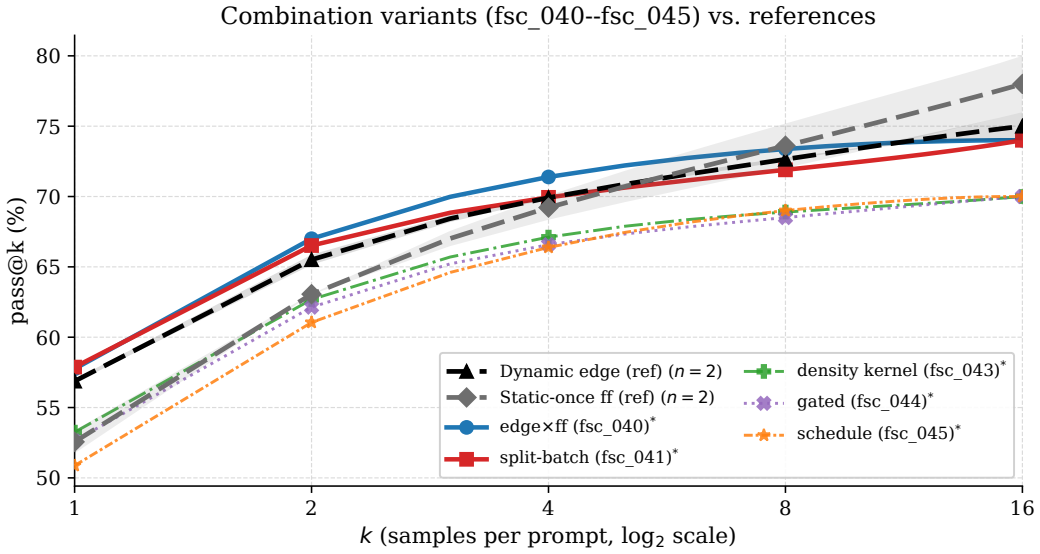
Figure 4: **Result R3 (DV structure is an independent signal).** All three runs freeze the sampling distribution, so dynamic recomputation cannot explain the gap. A frozen DV-region curriculum (`feature_failure`) pulls clearly ahead at high k of both a frozen per-example edge and the uniform control. The `feature_failure` and uniform-control curves are 2-seed means with ± 1 s.d. bands; the frozen per-example edge (`fsc_028`) is a single completed seed. The `feature_failure` band reflects the two-seed mean (78.0 pass@16), not the favorable single seed.

5.1.4 R4: Combining dynamic difficulty and DV.

The end-of-pipeline policies fuse both ideas through different mechanisms: multiplicative scoring (`fsc_040`), split-batch sampling (`fsc_041`), a density-corrected kernel (`fsc_043`), gating the DV term to unobserved examples (`fsc_044`), and a two-phase edge→feature_failure schedule (`fsc_045`); a weighted-average variant (`fsc_042`) crashed in training and is omitted. Table 3 and Figure 5 report this batch against the dynamic-edge and frozen-`feature_failure` references. The multiplicative and split-batch fusions (`fsc_040`, `fsc_041`) reach the highest pass@1 in the entire study (57.75, 57.88), edging out dynamic edge (56.88) while preserving the hard-example diet, and the density-corrected kernel restores the hard-example share that the uncorrected kernel suppressed. These are genuine improvements: a fused curriculum gives the best pass@1 observed in the study without collapsing high-k accuracy. The remaining gap is at high k, where no combination yet matches the frozen `feature_failure` level of $\sim$78 (the best reach 74.0, tying dynamic edge at $\sim$75). This is interpreted as *headroom rather than a ceiling*: every combination here uses a single, untuned setting of its fusion knobs (mixing ratio, kernel bandwidth, equal-weight features), and the 74-vs-78 gap is itself within single-seed paired noise ($\sim\pm$8–10 points pass@16). Better parameter selection, correlation-aware (ARD) per-feature weights, and curated feature subsets or fusion ratios are the natural levers for converting the strong pass@1 result into a matching high-k gain (Section 6).

Table 3: Combination fusion variants (`fsc_040`–`fsc_045`, single seed) vs. references. `fsc_042` (weighted average) crashed during training.

Run	Fusion / sampling	pass@1	pass@8	pass@16
`fsc_040`	multiplicative edge×ff	57.75	73.38	74.00
`fsc_041`	split-batch 70/30 (corr. ff)	**57.88**	71.90	74.00
`fsc_043`	density-corrected kernel	53.25	68.90	70.00
`fsc_044`	gated-unobserved ff	52.62	68.50	70.00
`fsc_045`	schedule edge→ff @ 50	50.88	69.00	70.00
`fsc_027`	dynamic edge (ref, 2-seed)	56.88	72.65	75.00
`fsc_031`	static-once ff (ref, 2-seed)	52.56	**73.61**	**78.00**



Shaded band: ±1 s.d. across training seeds (*n* given in legend). * single seed only; additional seeds not run due to compute budget.

Figure 5: Combination variants (`fsc_040`–`fsc_045`) against the dynamic-edge and frozen-`feature_failure` references. The fusions track the dynamic-edge curve and do not reach the frozen DV policy's high-k ceiling. References are 2-seed means with ± 1 s.d. bands; the combination variants ($*$) are single-seed, as the compute budget did not allow replicate seeds for this batch.

What is statistically resolvable. Because all runs are scored on the *same* 50 problems, a paired bootstrap (per-problem differences, 5K resamples) is far more powerful than the $\sim\pm 12$-point absolute single-eval noise (Table 4). The dominant source of that single-eval variance is *which* problems happen to populate the small 50-problem set (some Countdown instances are hard (or easy) for *every* method) so this shared problem-difficulty term inflates each method's absolute CI yet cancels exactly in the per-problem difference, leaving only the contrast of interest. The crucial caveat is that pairing removes *only* problem-level noise: it does nothing about *training-seed* noise (i.e. the run-to-run variation of the optimizer itself) which is an independent source that can be addressed only with replicate seeds. This is why a tight paired CI and a single-seed caveat coexist throughout: the two guard against different things, and a result is fully robust only when it survives *both*. Three contrasts survive in the predicted direction: dynamic edge beats RLOO at pass@1 ($+6.4$, P=0.98); the additive frozen DV policy beats the kernel at pass@16 ($+8.0$, CI excluding 0) supporting the preserve-vs-suppress prediction of the training-diet analysis (Section 5.2); and dynamic edge beats the additive DV policy at pass@1 (-5.2 paired diff, Table 4). Edge *ties* the kernel at pass@1 ($+0.6$; suppressing the hard share costs high-k diversity but not pass@1) yet *trails* the frozen additive policy at pass@16 (75.0 vs. 78.0, two-seed means), while edge-vs-uniform at pass@1 is *not* robust at this eval size. The paper therefore foregrounds the diet mechanism, which is measured on $\sim$14K training

Table 4: Paired bootstrap on the shared 50 eval problems (5K resamples). Pairing cancels the common problem-difficulty noise that inflates absolute single-eval CIs to $\sim\pm 12$ points. "Edge" is `fsc_027` r002; "uniform control" is `fsc_000` r001 (headline Table 1 reports 2-seed means where noted).

Contrast ($A - B$)	metric	diff	95% CI
edge − RLOO	pass@1	$+6.4$	$[-0.4, +13.8]$
edge − uniform control	pass@1	$+2.6$	$[-2.4, +7.9]$
edge − kernel	pass@1	$+0.6$	$[-3.6, +5.2]$
edge − kernel	pass@16	$+4.0$	$[-4.0, +8.0]$
static-ff − kernel	pass@16	$+8.0$	$[+0.0, +14.0]$
static-ff − edge	pass@1	-5.2	$[-10.4, -0.2]$

Table 5: Per-bin failure rate with 95% CIs over the observed `fsc_027` rows. Per-*bin* coverage is effectively complete despite $\sim$3% per-*example* coverage, so the difficulty map is tightly pinned.

Field	value	n	mean fail	95% CI
`num_count`	3	6,992	0.338	± 0.009
`num_count`	4	7,088	0.750	± 0.008
`shortest_operand_count`	3	7,390	0.381	± 0.009
`shortest_operand_count`	4	5,192	0.720	± 0.010
`all_numbers_required`	0	2,506	0.778	± 0.012
`all_numbers_required`	1	11,574	0.495	± 0.008

rows rather than 50 eval problems, and the direction-of-effect results, and flags any sub-10-point pass@16 gap as requiring additional seeds.

5.2 Qualitative Analysis

The edge is interpretable. Offline Spearman analysis of the observation logs consistently identifies the same structural predictors of failure across dynamic, static-once, and kernel runs. `num_count` is the strongest direct predictor ($\rho \approx 0.5$), followed by solver-derived fields: `shortest_operand_count` (≈ 0.22–0.31), `all_numbers_required` (≈ -0.25–-0.28), `nps_log1p` (≈ -0.22–-0.27), and `shortest_expression_depth` (≈ 0.16–0.22). Concretely, problems whose shortest solution uses ≤ 3 operands fail about 38% of the time, versus $\sim$72% for 4-operand shortest solutions (Table 5).

Difficulty is low-dimensional and cheap to estimate. This is arguably the most transferable finding. In Countdown, the difficulty the RL student actually experiences is a low-dimensional, structurally legible function of a few solver-derived features (Table 5). Although per-*example* coverage is only $\sim$3% of the 490K-example dataset (a matched run rolls out $\sim$14K unique examples), per-*bin* coverage is essentially complete because the DV fields take only a handful of distinct values, so each difficulty cell is estimated with very tight confidence intervals. The landscape is in fact recoverable from a tiny, near-constant number of rollouts: subsampling the observed set and attempting to recover the dominant difficulty axis (the `num_count` 3-vs-4 gap, true value 0.411 from Table 5) returns the correct sign in $20/20$ random trials from as few as 20 observations ($\sim$0.004% of the dataset), and the correct magnitude (≈ 0.41) from $\sim$50 (Appendix B, Table 7). Per-example difficulty estimation scales with the dataset so every example must be rolled out, but *structural* difficulty estimation scales with the number of feature bins, tens of observations per bin, independent of dataset size. This is precisely why a DV curriculum works at $\sim$3% coverage where a per-example curriculum cannot, and it *decouples curriculum cost from dataset size*. This estimates a coarse difficulty *map*, not per-example labels (within-bin fail rates still vary widely), and low dimensionality is a property of this task for which the DV-correlation analysis is exactly the test.

A tiny, cheap probe buys a large learning change. The intervention is deliberately small along *every* axis, which is precisely what makes the effect notable. The only *fresh, unbiased* measurement is a probe of 128 prompts every 10 training steps, so just 10 probes, $\approx$1,280 probe rollouts, over a 100-step run and the per-step train-batch updates that also refresh the weights reuse rollouts already

Table 6: Within-group reward variance—the RLOO gradient proxy of Eq. (2)—by difficulty band, over the $14{,}080$ observed rows of the dynamic-edge run (`fsc_027`). About 55% of observed examples (solved + impossible) carry essentially zero gradient regardless of how often they are sampled.

Band (fail rate)	share	mean reward var.	mean reward
solved (< 0.1)	18.9%	0.000	1.00
easy (0.1–0.3)	20.7%	0.118	0.84
edge (0.3–0.7)	15.0%	**0.203**	0.56
hard (0.7–0.9)	9.6%	0.123	0.24
impossible (> 0.9)	35.8%	0.002	0.06

produced for the RLOO update, so they add *no extra rollout cost* ($\approx$14K total observed rows, $\sim 3\%$ of the 490K-example pool). The resulting distribution barely moves: the active dynamic-edge distribution stays essentially uniform, with normalized weight entropy ≈ 1.00 and effective-sample fraction $\approx 99.5\%$ (near-zero total variation from uniform). The natural critique is therefore that the gains must be marginal. But coverage and distributional movement are the wrong yardsticks; *learning signal* is the right one. The RLOO gradient proxy of Eq. (2) is distributed extremely unevenly across difficulty (Table 6): roughly 55% of observed examples (fully solved or impossible) contribute essentially zero gradient, while the thin edge band carries the most. Reweighting a true uniform draw, the step-0 probe ($n = 128$) at a fixed model state, by the edge policy raises mean usable gradient signal per sampled example from 0.084 to 0.143, a $+69\%$ gain (5,000-sample bootstrap 95% CI $[+54\%, +89\%]$, all resamples positive). A near-free reweighting that looks almost uniform by entropy thus lifts the per-step learning signal by well over half, simply by moving mass off the zero-signal extremes. This is the mechanism behind the headline pass@1 gains over both plain RLOO ($+6.4$ pass@1, paired P=0.98) and the static-once edge ($+7.5$ pass@1), and it is why a small, cheaply estimated reweighting is evidence of *leverage* rather than a marginal effect. (The $+2.6$ pass@1 from the per-problem paired bootstrap in Table 4 is smaller than the $+5.6$ pass@1 of the 2-seed headline means (Table 1) only because it pairs single seeds against the more favorable uniform seed (`fsc_000` r001, 54.4); the effect holds under both comparisons.)

Interactions. The strongest pairwise interactions are `nps_*` $\times$ `shortest_operand_count` (failure-rate spread $\sim$0.80–0.82) and `nps_*` $\times$ `shortest_expression_depth` ($\sim$0.77). These interactions are precisely what the joint kernel (Eq. (5)) is meant to capture and what the additive policy (Eq. (4)) cannot.

Training-diet analysis. A useful explanatory lens is how each policy reshapes the dataset's naturally $\sim 50\%$ "hard" (4-operand) share, measured on the realized training diet from the observation logs. Policies split into *preserve* (edge, `feature_failure`; near-50% retained), *amplify* (correlation-weighted `feature_failure`), and *suppress* (the equal-weight kernel, which collapses the hard share to $\sim 24\%$). pass@16 tracks this diet while pass@1 does not. All five combination variants (`fsc_040`–`fsc_045`) land in the *preserve* band (48.9–54.1% hard share, effective-sample fraction $\geq 96\%$). The *effective-sample fraction* is the effective sample size $\left(\sum_i w_i^2\right)^{-1}$ (with w the normalized per-prompt sampling weights) expressed as a fraction of the dataset: it equals 1.0 when the curriculum weights are as diverse as uniform sampling and falls toward 0 as mass concentrates on a few prompts, so $\geq 96\%$ confirms these fusions barely narrow coverage.

The two kernel variants make the suppression mechanism concrete. The *uncorrected* kernel (i.e. the equal-weight `edge_dv_kernel` of Eq. (5) with no density correction) scores each example by the kernel-weighted *sum* of nearby anchor difficulties, $\sum_j K(x_i, x_j)\, a_j$. Because that sum grows wherever DV space is densely populated (for example, the easy, 3-number majority) it inflates easy-region scores and collapses the hard share to 23.9%. The *density-corrected* kernel (`fsc_043`) divides each score by the local kernel density $\sum_j K(x_i, x_j)$, replacing the sum with a density-normalized *average* nearby difficulty so the policy follows difficulty rather than DV-space sample density; this removes the policy's suppressive behaviour toward hard problems and restores the hard share to 48.9%. This explains why the additive frozen policy still leads at high k (preserving the hard diet is necessary but, on this batch, not sufficient to exceed the frozen `feature_failure` policy's high-k level of ~ 78) and motivates correlation-aware weighting for the kernel.

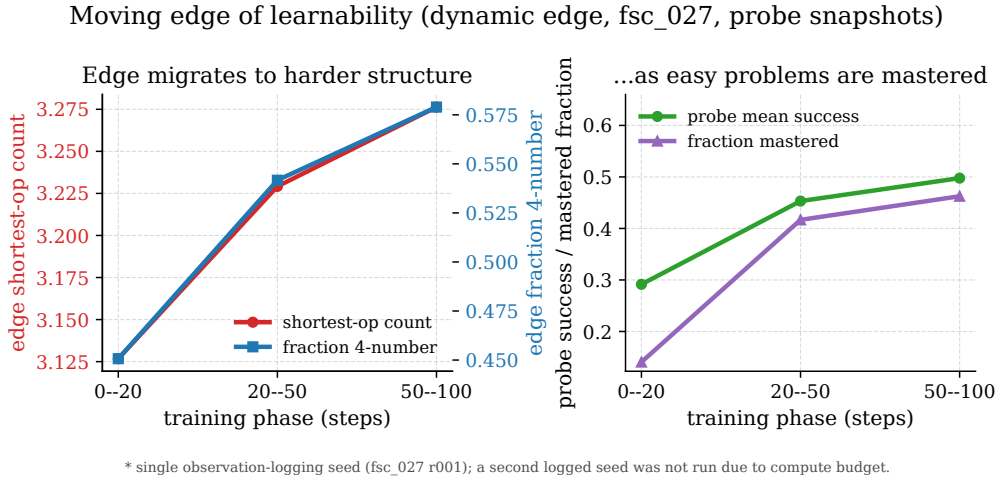


Figure 6: Moving edge of learnability from fresh probe snapshots (`fsc_027`). *Left:* as training proceeds, the edge-region ($s(x) \approx 0.5$) examples shift to harder structure (rising shortest-solution operand count and fraction of 4-number problems). *Right:* this happens as a growing fraction of problems become mastered. The structural frontier moves, which a frozen static-once curriculum cannot track. This interpretability figure uses the single `fsc_027` r001 probe log ($*$); a second observation-logging seed was not run due to compute budget.

Where the kernel loses high-k. Conditioning on the 4-number problems each policy *does* keep exposes a second-order effect that the diet shares alone miss. The uncorrected kernel (`fsc_032`) not only halves the hard share but reshapes *within* it: it over-concentrates on the all-numbers-required, 4-operand subset ($\sim$78% of its retained 4-number diet versus $\sim$73% for edge) while dropping *deep* expressions (`shortest_expression_depth` ≥ 3 falls to $\sim$18% from $\sim$28%). Under-training deep 4-number problems is a concrete, sliced-eval-testable explanation for the kernel's weaker pass@16, and it reinforces that the deficit is a distance/weighting *configuration* issue, i.e., the single-bandwidth, equal-weight metric follows DV-space density toward the easy majority, rather than a limit of joint feature scoring.

The curriculum actually moves the distribution. Counterfactual diagnostics confirm that the active runs differ materially from both uniform and edge-only sampling. An early lesson was the opposite failure mode: a first FSC pass *measured* difficulty (weight entropy ≈ 0.99997) without *applying* pressure, which motivated the neutral prior and a smaller sampling ϵ.

The edge migrates toward harder structure over training. Profiling the *probe* snapshots of the dynamic-edge run (`fsc_027`) by training phase (steps 0–20, 20–50, 50–100) reveals that the edge of learnability moves through feature space as the model learns (Figure 6). The analysis uses the probe rows rather than the training rows because probes are freshly measured on the current model every 10 steps, free of the Exponential Moving Average (EMA) smoothing and curriculum-sampling bias of the training stream. As probe mean success rises ($0.29 \to 0.45 \to 0.50$) and the fraction of mastered problems grows ($0.14 \to 0.42 \to 0.46$), the edge-region examples shift to structurally harder problems: the fraction of 4-number problems at the frontier rises $0.45 \to 0.54 \to 0.58$ and the mean shortest-solution operand count rises $3.13 \to 3.23 \to 3.28$. This is direct evidence for the central thesis: difficulty is model-state-dependent, the frontier is legible in DV coordinates, and it migrates toward harder structure as easy problems are mastered; a static-once curriculum cannot follow this signal.

6 Discussion

The two central claims are supported. The paper makes two proposals, and both are borne out within the scope of this study (a shared 50×16 eval, replicated where noted). First, *immediate feedback about the edge helps*: dynamically remeasuring the edge beats a delayed/frozen estimate

by $+7.5$ pass@1 with the scoring rule held fixed, and probe snapshots show the edge *measurably migrating* toward harder structure as easy problems are mastered (Figure 6) which is a moving target that a static-once or no-curriculum baseline cannot track. Second, *DV-style feature selection informs the curriculum*: with the sampling distribution frozen in both arms, a DV-region policy beats a frozen uniform control by $+6.0$ pass@16 and a frozen per-example edge by $+10.0$ pass@16 (`fsc_031` vs. `fsc_028`, two-seed `feature_failure` mean), isolating interpretable task structure as an independent high-k driver. The moving-edge analysis is evidence for the first claim and the static isolation is evidence for the second.

The implemented edge target is analytically justified, not tuned. The success-rate target $\tau = 0.5$ underpinning the first claim is the small-c limit of the gradient-norm-optimal target for the unnormalized RLOO advantage on the non-binary $\{0, c, 1\}$ reward: the exact optimum is $p^\star = 0.5 - \frac{c}{2}m$ (Section 3.5, App. E), so with $c = 0.1$ the deviation from 0.5 is at most $0.05\,m$ and the implementation simply uses 0.5. This frames both controls. Against *plain RLOO* (no curriculum), which spends a large fraction of rollouts on zero-signal prompts, steering mass to the $p \approx 0.5$ band raises usable gradient signal by $+69\%$ while leaving coverage essentially uniform (effective-sample fraction $\approx 99.5\%$, near-zero total variation from uniform) (Table 6). Against the *uniform-weight FSC* control (the same pipeline with the edge weighting flattened), the edge target is the only varied factor, so its gains isolate the value of aiming at the analytically optimal difficulty rather than of the surrounding machinery. The residual $0.05\,m$ term is the second-order signal left on the table by not solving for the exact ternary optimum online, and is negligible at $c = 0.1$.

Additional observations. Beyond the two central claims, several secondary findings are worth calling out. **(i) The two ideas compose:** fusing them (`fsc_040`/`fsc_041`) gives the best pass@1 in the study while preserving the hard-example diet. **(ii) pass@1 and pass@16 are governed by different mechanisms:** pass@16 tracks the realized training diet (preserve/amplify/suppress of the hard-example share) whereas pass@1 does not. **(iii) Additive beats joint at high k:** the binned DV policy currently outperforms the DV-space kernel, which the paper traces to the kernel's equal-weight, single-bandwidth distance rather than to any limit of feature-space curricula. **(iv) Dynamic edge escapes diversity collapse:** it lifts pass@1 to ~ 57 without sacrificing the high-k ceiling, in contrast to plain RLOO; fusions reach the study maximum (~ 58). **(v) Structure, not just scalars, is what helps:** naive failure-rate-only and failure-rate+edge policies do not improve over the control (Appendix A), underscoring that the gains come from *structured* feedback. **(vi) The gains are leverage, not coverage:** a curriculum whose only fresh, unbiased signal is a 128-prompt probe every 10 steps (plus free per-step train-batch updates, $\sim 3\%$ coverage overall) leaves the active distribution essentially uniform (effective-sample fraction $\approx 99.5\%$, near-zero total variation from uniform) yet still raises usable RLOO gradient signal by $+69\%$ (Table 6), because $\sim 55\%$ of examples are zero-variance and teach nothing; the edge target is analytically the maximum-gradient success rate (Section 3.5). **(vii) A tiny probe characterizes the whole landscape:** the dominant difficulty axis is recoverable from tens of rollouts ($\sim 0.004\%$ of the dataset), so the difficulty map and hence curriculum cost decouples from dataset size, which is what makes a $\sim 3\%$-coverage curriculum effective at all.

Additive versus joint feature handling, and the tuning headroom. The additive policy currently wins at high k while the product kernel only matches per-example edge. The likely reason is the kernel's equal-weight, single-bandwidth distance: because Eq. (5) is a product over features, irrelevant dimensions inflate the joint distance and shrink the kernel for genuinely near neighbors, diluting exactly the interactions the policy exists to capture. Crucially, this is a *configuration* limitation, not a limitation of feature-space curricula: every combination variant used a single untuned setting of its mixing ratio, kernel bandwidth, and (equal) feature weights. The clear static-case win and the best-in-study pass@1 of the fusions suggest the high-k gap is closable with better parameter selection, ARD-style per-feature lengthscales $w_f \propto |\text{Spearman}_f|$, and curated feature subsets or fusion ratios (these have been implemented but not yet varied).

Limitations. The eval set is small (50×16), several headline numbers are single-seed (reported with paired comparisons and seed means where available), and results are for one task and one model size. The DV-kernel-versus-per-example-edge comparison at high k remains open, and the adversarial feature-conditioned *generation* envisioned in the proposal is not yet built. These gaps are substantially a *compute* constraint rather than a design one: with a single 0.5B policy on a fixed budget, the seed count, the refresh interval and probe size, the eval-set size, and the breadth of the

policy/bandwidth sweep were all chosen to fit available GPU time, and the bilevel teacher–student generation loop discussed below is markedly more expensive than the resampling layer the project ran. Larger budgets would directly relax the single-seed and small-eval limitations and permit the omitted hyperparameter sweeps.

Future work. The natural next step is feature-conditioned data *generation* around the moving frontier, and the cleanest framing is a direct comparison against SOAR (Sundaram et al., 2026). SOAR learns a teacher generator that proposes synthetic stepping-stone problems and is rewarded, through a bilevel meta-RL loop, only by the student's *delayed, downstream* improvement on a fixed hard target set. FSC offers a complementary handle on the same edge-of-learnability problem: it already measures the current student's frontier *directly* in interpretable DV space, so generation can be conditioned on a measured edge profile (and solver/calculator-verified) rather than inferred only through delayed teacher reward. Future work would therefore add a solver-verified, DV-conditioned Countdown generator with a baseline-plus-generated-data arm and evaluate it head-to-head with a SOAR-style grounded-teacher baseline, isolating whether direct feature-space edge measurement matches or improves on learned teacher generation while removing the bilevel reward loop (at the cost of requiring a verifier). Near-term, and largely gated by compute: additional seeds (the combination batch is currently single-seed); a refresh-interval and training-length ablation that, with probe size held fixed, varies unbiased probe frequency at $\{1, 5, 10, 25, 50, 100\}$ steps under a longer global step budget so each arm receives multiple uniform probes, isolating how often recalibration is needed given that per-step train-batch updates already refresh weights at no extra rollout cost; sliced evaluation by DV bin; a fuller correlation-weighted (ARD) kernel study; and a learned DV$\rightarrow$weight mapping.

7 Conclusion

The paper presented Feature-Space Curriculum learning, which measures the policy's edge of learnability from rollouts and describes it in an interpretable, solver-derived Difficulty Vector. The central claim of the proposal (that difficulty is both *model-state-dependent* and *structurally legible*) is borne out on both counts, and, importantly, by two *separable* mechanisms. Dynamically recomputing the edge during training beats freezing it to its initialization-time value by $+7.5$ pass@1 and $+5$–7 pass@8/16 with the scoring rule held fixed, confirming that the frontier is a moving target the curriculum must track. Holding recomputation fixed, a frozen DV-region curriculum still beats a frozen uniform control by $+6.0$ pass@16 (and a frozen per-example edge by $+10.0$), showing that interpretable task structure is an independent (and the strongest high-k) sampling signal in its own right. Combining the two attains the best pass@1 in the study, and fresh probe snapshots make the underlying dynamics legible: the edge migrates toward harder structure (high shortest-solution operand count, low valid-solution count) as the model masters easy problems, so the same vector that steers sampling also localizes *where* learning is happening.

These gains come at almost no cost. The only fresh, unbiased signal is a 128-prompt probe every 10 steps ($\sim 3\%$ example coverage; per-step updates reuse RLOO rollouts for free), and although the resulting distribution stays nearly uniform (effective-sample fraction $\approx 99.5\%$) it raises usable RLOO gradient signal by $+69\%$ which is evidence that FSC works by *leverage*, not coverage. The edge target $\tau{=}0.5$ is analytically grounded, not tuned: for un-normalized RLOO on the ternary Countdown reward $\{0, c, 1\}$, the gradient-norm-optimal success rate is $p^\star = \frac{1}{2} - \frac{c}{2}m$, recovering the standard $p{=}0.5$ edge-of-learnability result in the binary limit (Section 3.5). The one open gap—the high-k shortfall of the joint DV-space kernel relative to the additive DV policy—is traced to an untuned equal-weight, single-bandwidth distance and is a tuning opportunity rather than a structural limit. Taken together, the results reframe difficulty as a measurable, moving property of the policy that can be read out in human terms, and point to the natural next step the proposal anticipated: conditioning data *generation*, not merely resampling, on the measured edge profile, with a direct comparison against learned-teacher approaches such as SOAR (Section 6).

8 Team Contributions

- **Vishaal Saraiya:** sole contributorfor problem formulation, FSC implementation, all training and evaluation runs, analysis, and writing.

Use of AI assistance. Research direction, experimental design, and empirical findings are the author's own; AI coding assistants were used as an implementation and review aid, and all AI-assisted output was reviewed and validated by the author. The one place AI shaped an analytical result is the supplementary edge-target proof (item 3 below), which is not load-bearing: the implemented $p = 0.5$ follows from the author's choice to treat partial credit as failure. AI assistance contributed to:

1. metrics and logging instrumentation, and post-hoc analysis scripts; the author specified *what* to look for but a number of metrics were suggested by AI
2. implementation of certain flags and their effects that were **never exercised**, left as possible future work if time and compute permitted: `curriculum_refresh_unit`, `curriculum_smoothing`, `edge_dv_kernel_top_anchors`, `operator_source`, `include_model_solution_features`, `log_observation_prompts`, `feature_failure_fields`, `feature_failure_num_bins`, `feature_failure_min_observations`, `feature_failure_smoothing`, `edge_dv_kernel_fields`, `edge_dv_kernel_bandwidth`, `edge_dv_kernel_anchor_mode`, `edge_dv_kernel_min_anchor_score`, `edge_dv_kernel_min_anchors`, and `edge_dv_kernel_categorical_penalty`, among others
3. bug fixing, error checking, documentation, and the unit-test suites described below
4. the edge-target proofs: the author set up the SEC-style expected-advantage framework over p, q, m ($p + q + m = 1$) and the binary modeling choice fixing the target at $p = 0.5$; AI proposed the gradient-norm (variance) argument and produced both the binary $p = 0.5$ result and its (non-load-bearing) ternary generalization $p^\star = \frac{1}{2} - \frac{c}{2}m$ (Section 3.5, Appendix E)
5. surfacing the Self-Evolving Curriculum work (Chen et al., 2025) in a final literature pass—a double-edged, second last-day find that both sharpened the related-work positioning (Section 2) and forced substantial late additional work
6. the "What is statistically resolvable" analysis (Section 5.1.4)
7. LaTeX minor restructuring, rephrasing and reformatting of this report and reformatting the poster.

Changes from Proposal. *Scope.* The proposal's committed deliverable was the feature-space *resampling* curriculum which has been achieved. The adversarial feature-conditioned data *generation* was a conditional, aspirational stretch goal, that was depriritized given teh addition insights, investigations and conclusions listed below. It remains as part of possible future work (Section 6). This report delivers and validates the committed resampling half (dynamic edge and DV-as-signal) in full.

Additional work beyond the proposal. Several pieces were designed and built during the work that the proposal did not specify:

1. The Difficulty Vector was extended with exact-solver-derived fields (shortest-solution operand count and expression depth, an `all_numbers_required` flag, and capped valid-solution counts) beyond the originally hand-defined components, computed by an exact `Fraction`-based subset solver (Appendix D).
2. The proposal treated difficulty as a precomputed, static quantity; the project made it *model-state-dependent*, adding the edge-of-learnability probe with a refresh interval and EMA-smoothed dynamic metadata, and the `static`/`static_once`/`dynamic` mode distinction that isolates dynamic recomputation as a variable.
3. The proposal's three abstract uses of the DV (scalar projection, learned mapping, feature analysis) were realized as two concrete, contrasting policies (an *additive* `feature_failure` binning score and a *joint* `edge_dv_kernel` (Section 3)), together with fusion modes (multiplicative, weighted-average, split-batch, and observation-gated), phase scheduling, a correlation-aware (ARD-style) per-feature weighting, and a density correction, all of which were unanticipated in the proposal.
4. The project identified and justified the correct internal control (the uniform FSC control rather than the RLOO baseline, Section 4) and added counterfactual/concentration diagnostics logging that the proposal did not contemplate.

5. The project derived an analytical justification for the edge target: for un-normalized RLOO on the ternary Countdown reward $\{0, c, 1\}$, the gradient-norm-optimal full-success rate is $p^\star = \frac{1}{2} - \frac{c}{2}m$, recovering the binary p=0.5 edge-of-learnability result in the limit and showing that the implemented τ=0.5 is exact once difficulty is measured on the binarized `fail_rate` (Section 3.5, Appendix E) which was not part of the original proposal

Testing within scope. Correctness of this machinery was validated with two unit-test suites. The feature-extraction suite checks that known- and shortest-solution structure is recovered from answer tags and ground truth, that the subset solver finds shorter solutions and reports the `all_numbers_required` flag, and that the valid-solution-count cap and its `log1p` transform are applied correctly. The curriculum-state suite checks the sampling machinery end to end: that `feature_failure` upweights the hard DV region; that sparse bins and sparse kernel anchors fall back to the neutral prior; that the edge fuses correctly under each combiner (mean, multiply, weighted-average, observation-gated, split-batch); that schedule-style policy switches and the density-corrected kernel behave as specified; that ARD-style correlation weighting emphasizes signal fields and prunes noise; and that the counterfactual diagnostics (active-minus-edge, active-minus-uniform) are emitted.

A note on reference formatting. As a quality-of-life convenience, and as a deliberate deviation from the default template, each arXiv-hosted entry in the bibliography below is hyperlinked directly to its arXiv abstract page. This changes presentation only—citation keys, ordering, and the underlying `plainnat` style are otherwise unchanged—and is intended purely to make the references easier to follow.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms, 2024. https://arxiv.org/abs/2402.14740.

Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. A general theoretical paradigm to understand learning from human preferences, 2023. https://arxiv.org/abs/2310.12036.

Sanghwan Bae, Jiwoo Hong, Min Young Lee, Hanbyul Kim, JeongYeon Nam, and Donghyun Kwak. Online difficulty filtering for reasoning oriented reinforcement learning, 2025. https://arxiv.org/abs/2504.03380.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, pages 41–48, 2009.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. Self-evolving curriculum for llm reasoning, 2025. https://arxiv.org/abs/2505.14970.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. https://arxiv.org/abs/2501.12948.

Thomas Foster, Anya Sims, Johannes Forkel, Mattie Fellows, and Jakob Foerster. Learning to reason at the frontier of learnability, 2025. https://arxiv.org/abs/2502.12272.

Kanishk Gandhi, Denise Lee, Gabriel Grand, Muxin Liu, Winson Cheng, Archit Sharma, and Noah D. Goodman. Stream of search (sos): Learning to search in language, 2024. https://arxiv.org/abs/2404.03683.

Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. Prioritized level replay. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in rl training of llms, 2024. https://arxiv.org/abs/2410.01679.

M. Pawan Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2010.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023. https://arxiv.org/abs/2305.20050.

Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Teacher-student curriculum learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2019.

Moonshot AI. Kimi k1.5: Scaling reinforcement learning with llms, 2025. https://arxiv.org/abs/2501.12599.

Jiayi Pan, Junjie Zhang, Xingyao Wang, Lifan Yuan, Hao Peng, and Alane Suhr. Tinyzero. https://github.com/Jiayi-Pan/TinyZero, 2025. Accessed: 2025.

Shubham Parashar, Shurui Gui, Xiner Li, Hongyi Ling, Sushil Vemuri, Blake Olson, Eric Li, Yu Zhang, James Caverlee, Dileep Kalathil, and Shuiwang Ji. Curriculum reinforcement learning from easy to hard tasks improves llm reasoning, 2025. https://arxiv.org/abs/2506.06632.

Qwen Team. Qwen2.5 technical report, 2024. https://arxiv.org/abs/2412.15115.

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

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, 2024. https://arxiv.org/abs/2402.03300.

Shobhita Sundaram, John Quan, Ariel Kwiatkowski, Kartik Ahuja, Yann Ollivier, and Julia Kempe. Teaching models to teach themselves: Reasoning at the edge of learnability, 2026. https://arxiv.org/abs/2601.18778.

Haozhe Wang, Haoran Que, Qixin Xu, Minghao Liu, Wangchunshu Zhou, Jiazhan Feng, Wanjun Zhong, Wei Ye, Tong Yang, Wenhao Huang, Ge Zhang, and Fangzhen Lin. Reverse-engineered reasoning for open-ended generation, 2025. https://arxiv.org/abs/2509.06160.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at scale, 2025. https://arxiv.org/abs/2503.14476.

Ruize Zhang, Zelai Xu, Chengdong Ma, Chao Yu, Wei-Wei Tu, Wenhao Tang, Shiyu Huang, Deheng Ye, Wenbo Ding, Yaodong Yang, and Yu Wang. A survey on self-play methods in reinforcement learning, 2025a. https://arxiv.org/abs/2408.01072.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning, 2025b. https://arxiv.org/abs/2501.07301.

Appendices

A Additional Experiments

The project additionally ran failure-rate-only and failure-rate+edge policies, the weighted-average fusion (`fsc_042`, which crashed in training), and correlation-weighted versions of both DV policies (`fsc_035`, `fsc_036`). The correlation-weighted `feature_failure` (`fsc_036`) reaches $54.0/70.9/72.0$ pass@1/8/16 and the correlation-weighted kernel (`fsc_035`) $53.8/69.0/70.0$; neither exceeds the dynamic-edge or frozen-`feature_failure` references. Per-conclusion pass@k curves are given in the body (Figures 2–5); the complete per-run numeric table is reproduced from `pass_at_k_all_json.csv`.

B Subsampling sample-complexity check

To audit the claim that structural difficulty is recoverable from a tiny number of rollouts, the project subsamples the `fsc_027` r001 observation log ($\sim$14K rows) uniformly without replacement to a fixed total budget, estimate mean `fail_rate` in the `num_count`$= 3$ and $= 4$ bins, and take their difference as the recovered gap. The true gap on the full observed set is 0.411 (Table 5). The project repeats each budget with 20 independent random subsamples (trial seeds 0–19). Table 7 reports the mean and sample standard deviation of the recovered gap and the number of trials with the correct sign. The observed set is curriculum-selected, so these are resampling variances on a fixed log rather than a fully uniform draw over the 490K pool; the uniform step-0 probe mitigates but does not remove that bias.

C Per-seed replicate results

Four configurations were run with two training seeds; all other runs are single-seed for the compute reasons given in Section 4. Table 8 lists each seed individually together with the two-seed mean and sample standard deviation. The paper shows the raw per-seed values rather than relying on the s.d. alone because at $n = 2$ the standard deviation is a fragile estimate (it is simply $|a - b|/\sqrt{2}$); the individual values make the size of the sample explicit. The body tables (e.g. Table 1) report the two-seed means listed here. All eight evaluations are complete (50 problems $\times$ 16 samples), and both training seeds of each configuration reached the full 100-step horizon; the per-seed values are recomputed directly from the stored evaluation outputs. The `fsc_031` r002 seed required two restarts—two earlier attempts crashed at steps 36 and 87—before a third attempt completed all 100 steps; the project verified that the cited evaluation comes from that completed run by confirming its training observation log spans every step 0–99, so no partial-run data enters the analysis.

D Implementation Details

FSC wraps the existing RLOO sampling and update workers. Static DV features are precomputed with an exact `Fraction`-based subset solver; dynamic metadata is updated from grouped rollouts with EMA smoothing. Per-example observation rows are appended as JSONL and analyzed offline for DV correlations, binned failure rates, and pairwise interactions. All runs use the matched RLOO defaults listed in Section 4.

Full diagnostics suite. At each curriculum refresh the trainer logs, against both the uniform and edge-only counterfactual references, the complete set of distribution-distance and concentration diagnostics summarized in Section 4. The body foregrounds total variation and normalized weight entropy because those are the two quantities the later arguments use; for completeness the remaining logged diagnostics are: (i) Jensen–Shannon divergence between the active and reference weight vectors; (ii) Spearman rank correlation of the active and reference weights; (iii) batch-level set overlap between the realized batch and a same-size draw from each reference; (iv) effective sample size and effective-sample fraction; and (v) top-$1/5/10\%$ weight-mass concentration. These are recorded for every run but are not load-bearing for any claim in the body; the per-example JSONL observation logs are the input to the offline DV-correlation, binned-failure-rate, and pairwise-interaction analyses.

Table 7: Recovering the `num_count` 3-vs-4 fail-rate gap by uniform subsampling of the `fsc_027` r001 observation log (20 trials per budget). True gap $= 0.411$ (Table 5).

n_{obs}	% of 490K pool	recovered gap	s.d.	correct sign
2,000	0.41%	0.412	0.016	20/20
500	0.10%	0.421	0.029	20/20
100	0.020%	0.438	0.065	20/20
50	0.010%	0.402	0.124	20/20
20	0.004%	0.461	0.191	20/20

Table 8: Per-seed pass@k (%) for the four replicated configurations, with two-seed mean $\pm$ sample s.d. ($n = 2$, s.d. $= |a - b|/\sqrt{2}$). The means match the values reported in the body.

Run	seed	pass@1	pass@8	pass@16
FSC uniform control (`fsc_000`)	r001	54.38	68.72	72.00
	r002	48.25	69.82	72.00
	mean $\pm$ s.d.	51.31 ± 4.33	69.27 ± 0.78	72.00 ± 0.00
Static-once edge, basic (`fsc_022`)	r001	49.50	71.10	72.00
	r002	47.88	66.79	68.00
	mean $\pm$ s.d.	48.69 ± 1.15	68.95 ± 3.05	70.00 ± 2.83
Dynamic edge (`fsc_027`)	r001	56.75	72.76	74.00
	r002	57.00	72.53	76.00
	mean $\pm$ s.d.	56.88 ± 0.18	72.65 ± 0.16	75.00 ± 1.41
Static-once `feature_failure` (`fsc_031`)	r001	51.75	75.19	80.00
	r002	53.37	72.03	76.00
	mean $\pm$ s.d.	52.56 ± 1.15	73.61 ± 2.23	78.00 ± 2.83

Extended W&B logging. The diagnostics named above are the load-bearing subset of a substantially larger telemetry payload emitted to Weights & Biases at every curriculum refresh. In addition to standard optimization curves (loss, KL, policy entropy, mean reward, and per-step sample/generation tables), each refresh records probe-level summaries (reward, pass@k, failure and format-failure rates, mean edge distance, edge-target success rate); the full distribution-distance battery computed against *three* counterfactual references (uniform, edge-only, and feature-failure-only) rather than the two foregrounded in the body; and policy-internal state for the active modes, including feature-failure score moments, non-neutral fraction, and active-bin counts, and edge-DV kernel score/anchor statistics, neutral-fallback flags, high-score exposure, nearest-anchor distances, and density-correction terms, alongside static-DV summaries for the sampled and probe batches. The paper omits the exhaustive key list here; none of these additional channels are load-bearing for any claim in the body, but all are retained per run. Several standalone analysis scripts were written to pull these run histories from W&B and to compute the offline DV-correlation, binned-failure-rate, and pairwise-interaction summaries reported above.

E Variance- and MAD-optimal difficulty for non-binary RLOO

This appendix derives the objective-correct edge target used in Section 3.5 for the non-binary Countdown reward, and shows that the implemented $p = 0.5$ is its small-c limit.

Setup. Let the reward take three values $\{0, c, 1\}$ with $c = 0.1$, and write $p = \Pr(r{=}1)$, $q = \Pr(r{=}0)$, and partial-credit mass $m = \Pr(r{=}c) = 1 - p - q$, on the simplex $p, q, m \geq 0$. The two moments are

$$\mu \equiv \mathbb{E}[r] = p + c\,m, \qquad \mathbb{E}[r^2] = p + c^2\,m. \tag{6}$$

From the leave-one-out advantage (2), the per-prompt update magnitude scales with $|a_i| \propto |r_i - \bar{r}|$, so the faithful signal is the mean absolute deviation $\mathrm{MAD} = \mathbb{E}|r - \mu|$; the within-group variance $\mathrm{Var}(r) = \mathbb{E}[r^2] - \mu^2$ is a convenient proxy.

Variance proxy. Maximizing $\mathrm{Var}(r)$ over the simplex with a Lagrange multiplier λ for $\sum_i \pi_i = 1$ requires, for every *active* outcome, $\partial_{\pi_i}[\,\mathbb{E}[r^2] - \mu^2\,] = r_i^2 - 2\mu r_i - \lambda = 0$. Every active reward value thus solves the same quadratic $f(r) = r^2 - 2\mu r = \lambda$; a parabola attains any value at *at most two* points, so the variance-maximizing distribution is supported on at most two of $\{0, c, 1\}$. Of the three two-point options, the gap-squared-over-four rule gives maximum variances $\frac{1}{4}$ for $\{0, 1\}$, $(1-c)^2/4 = 0.2025$ for $\{c, 1\}$, and $c^2/4 = 0.0025$ for $\{0, c\}$; the *global* optimum is therefore the binary corner $p = q = \frac{1}{2}$, $m = 0$, $\mathrm{Var} = \frac{1}{4}$—partial credit only dilutes the signal. When m is exogenously fixed (the policy emits format-correct-but-wrong answers at some rate), $\partial_p \mathrm{Var} = 1 - 2\mu = 0$ gives the constrained optimum at *mean reward* $\mu = \frac{1}{2}$, i.e. $p = \frac{1}{2} - cm$.

MAD (the RLOO-faithful objective). With $\mu > c$ (true near $\mu \approx \frac{1}{2}$), $|c - \mu| = \mu - c$ and

$$\mathrm{MAD} = q\,\mu + m\,(\mu - c) + p\,(1 - \mu) = 2p\,(1 - p - cm), \tag{7}$$

using $\mu = p + cm$ and $q = 1 - p - m$. Maximizing over p at fixed m, $\partial_p \mathrm{MAD} = 2(1 - 2p - cm) = 0$, yields the objective-correct target of Eq. (3),

$$p^\star = \tfrac{1}{2} - \tfrac{c}{2}m, \quad \mu^\star = \tfrac{1}{2} + \tfrac{c}{2}m, \quad q^\star = \tfrac{1}{2} - \big(1 - \tfrac{c}{2}\big)m, \tag{8}$$

valid while $q^\star \geq 0$, i.e. $m \leq 1/(2 - c) \approx 0.53$; beyond this the optimum slides onto the $\{c, 1\}$ edge. The maximum signal is $\mathrm{MAD}^\star = (1 - cm)^2/2$, decreasing in m.

Reduction and implementation. As $c \to 0$ or $m \to 0$ both objectives reduce to the binary $\mathrm{MAD} = 2p(1-p)$, $\mathrm{Var} = p(1-p)$, maximized at $p = 0.5$, recovering the GRPO (Shao et al., 2024)/REINFORCE edge-of-learnability result (Chen et al., 2025; Bae et al., 2025). For $c = 0.1$ the MAD correction is only $0.05\,m$ and the variance correction $0.10\,m$, so the two agree to leading order and the implemented edge target $\tau = 0.5$ (Eq. (1)) is the exact RLOO optimum up to an $\mathcal{O}(cm)$ term. Moreover the curriculum measures difficulty on a *binarized* reward—the dynamic feature `fail_rate` $= \Pr(r < 1)$ buckets the partial-credit 0.1 as a failure—so the quantity it steers is exactly the Bernoulli full-success rate p, for which $p = 0.5$ is exactly optimal. The derivation therefore justifies the implemented target rather than prescribing a value tuned to it. The paper notes that Bae et al. (2025)'s reverse-KL bound is estimator-agnostic but assumes a β-KL-regularized objective, which the un-normalized RLOO advantage of Eq. (2) does not contain, so it does not by itself fix the RLOO non-binary target.