RUBRIC DROPOUT: A SIMPLE WAY TO MITIGATE REWARD HACKING IN RUBRIC-BASED RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Rubric-based reinforcement learning, in which an LLM judge scores responses against prompt-specific criteria, has become a standard way to post-train language models on tasks without verifiable answers. The rubric, however, is only a proxy for response quality, and a policy trained against it long enough learns to exploit the difference. We train models with GRPO on medical and science rubrics and grade their responses on out-of-distribution (OOD) benchmarks throughout training with two judges: the training judge and a stronger OOD judge. The two scores first rise together and then diverge. The policy is learning to satisfy the training judge without writing better responses, which is a form of reward hacking. To mitigate it, we propose ***Rubric Dropout***, which applies dropout to rubric criteria: at each training step, a random subset of each rubric's criteria is removed from the reward, so the rewarded subset of criteria is resampled at each step. All responses sampled for the same prompt are scored on the same subset, so their rewards remain comparable, and evaluation always uses the full rubric. We also introduce ***Block Dropout***, which drops semantically similar criteria together so that a near-duplicate of a dropped criterion cannot keep rewarding the same behavior. Across both benchmarks and models from 1.7B to 8B, dropout improves the gold judge's score on OOD benchmarks by up to 3% on HealthBench-Hard and 7% on ResearchQA, and it reduces both the gap between the two judges and over-crediting. It keeps in-domain reward close to the baseline, and dropout rates of 30–50% work best. Block Dropout improves the gold judge's OOD score further at the same dropout rate. Changing which criteria are rewarded at each step is a simple, inexpensive way to make rubric rewards harder to exploit. Anonymized Code

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) works well in domains where there is a ground truth to check the answer against (Lambert et al., 2025; Guo et al., 2025). However, many of the tasks we most want large language models to be good at, such as giving medical advice (Arora et al., 2025; Zhang et al., 2025) and explaining a research area, are open-ended, with no ground truth. A common approach is *Rubric-based* RL: write down a list of criteria for each prompt, have an LLM judge grade each criterion, and use the weighted fraction satisfied as the reward (Gunjal et al., 2026; Huang et al., 2025). The recipe is attractive because rubrics make quality explicit and auditable, and recent work reports strong gains from it.

The recipe also has a built-in weakness. The rubric score the training judge assigns is only a *proxy* for quality, and this proxy is *fixed*. The same criteria are scored at every training step, many of them generic templates that repeat across prompts ("uses clear language", "well-organized"). Prompt-specific, verifiable criteria would be ideal. At dataset scale, though, keeping every criterion that specific and verifiable is hard, and generic templates end up in the training set. This immutability makes the proxy exploitable. A criterion rewarded identically at every step is a stable target. Once the policy finds a cheap, surface-level way to satisfy it, the shortcut is reinforced at every subsequent step, and when the criterion is a shared template, it is reinforced on every prompt at once. A policy that learns to open every answer with a tidy bulleted summary satisfies "well-organized" everywhere, regardless of content. Classic results on reward misspecification indicate that optimizing hard against a fixed, imperfect proxy ends in reward hacking (Amodei et al., 2016; Skalse et al., 2022; Pan et al., 2022), and Gao et al. (2023) showed exactly this for learned reward models.

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

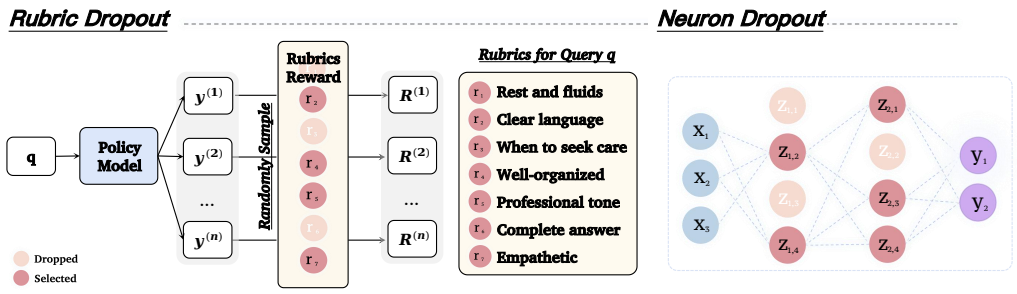


Figure 1: **Rubric Dropout randomly drops rubric criteria from the training reward.** Left: every rollout of a query is scored on the same randomly sampled sub-rubric (faded criteria are dropped). The mask is re-drawn at each training step. Dropout is train-only and evaluation uses the full rubric.

To see how severe this is for rubric rewards in practice, we trained with the standard recipe and tracked quality with a signal the training loop cannot see. Every 20 steps we graded an OOD evaluation set with two judges: the training judge, whose score is the *proxy*, and a stronger cross-family judge, whose score is the *gold*. We train against the proxy, not the gold judge, for the same reason practitioners do. The proxy is cheap enough to grade every rollout at every step, while the gold judge costs $20\times$ as much[1] and is affordable only for periodic evaluation. On two independent train→eval pairs, RubricHub-Medical→HealthBench-Hard and RubricHub-Science→ResearchQA, the proxy and gold scores diverge. The gold score rises, peaks, and then declines, while the proxy score keeps improving and the set of criteria it overclaims keeps growing. The proxy−gold gap widens from 29% to as much as 44% on HealthBench-Hard, and on ResearchQA gold falls 22 points from its peak. We read this divergence as a sign of hacking rather than judge bias. A merely biased judge would shift the gold curve by a constant and would not bend it downward while the proxy rises. This test alone cannot rule out a gold judge whose errors change with the policy. We therefore also look at which criteria the proxy overclaims (§ 4.4) and at how the base policy's outputs change (§ 6); both point to template-like shortcuts rather than to a bias in the gold judge. These runs indicate that rubric rewards are hacked in practice. Our study has two objectives: 1) Quantify reward hacking as the gap between training reward and a quality signal independent of the training judge and rubrics. 2) Develop a dynamic rubric-reward method that reduces reward hacking and works with existing RL infrastructure.

First, we introduce a two-judge evaluation protocol: at regular checkpoints, the *proxy* judge and a stronger *gold* judge grade the same responses on OOD tasks. Because hacking sets in late in training, this ind view catches degradation that end-of-run scores miss.

For the second objective, we propose ***Rubric Dropout*** (Fig. 1). At every training step, we randomly drop a fraction of the rubric's criteria before computing the reward, so the policy is scored on a sampled sub-rubric each time. No single criterion is then guaranteed to be rewarded at every step. This means the policy cannot easily exploit the reward with simple template-based strategies. The change is one line in the reward function, has a single hyperparameter, and adds no judge calls. We draw one mask per rollout group, so all rollouts of a prompt are graded on the same sub-rubric. We prove that under this scheme the advantage does not depend on the choice of reward normalizer. We also show that, before group standardization, dropout only rescales the expected advantage, while it acts as a variance regularizer on updates that depend heavily on any single criterion. Because near-duplicate criteria can preserve a signal even when one of them is dropped, we also introduce ***Block Dropout***, which jointly drops semantically similar criteria (§ 3.4).

Using this measurement and method, we compare three conditions throughout the paper: no dropout (*base*), dropout at $f\approx30\%$, and dropout at $f\approx50\%$. In both domains and at three model sizes, dropout mitigates the hacking. Peak scores are tied between base and dropout, and there is no measurable in-domain cost. We also find that dropout preserves output diversity late in training, while the base policy collapses to low-entropy, template-like responses (Fig. 7). Sweeping f shows a broad plateau, and reweighting, used in place of dropout, scores *below* base.

Our contributions are summarized as follows:

[1]Per million input/output tokens: `gpt-4o-mini` \$0.15/\$0.60, `claude-sonnet-4-6` \$3/\$15.

1. We quantify OOD reward hacking by evaluating the policy every 20 training steps with both the training judge and a stronger cross-family judge. On two independent benchmark pairs, the training-judge score continues to rise after the stronger-judge score peaks.
2. We propose Rubric Dropout, a simple regularizer for mitigating rubric reward hacking. Its group-shared masking scheme is theoretically compatible with GRPO. We further introduce block dropout, which drops semantically correlated criteria together.
3. Across two benchmark pairs and models from 1.7B to 8B, Rubric Dropout improves OOD gold reward and reduces reward hacking, with no loss of in-domain performance.

2 Related Work

Rubric-based reinforcement learning. Grading free-form text against explicit criteria began as an evaluation idea (Zheng et al., 2023; Arora et al., 2025) and has since been adopted for training. QA-LIGN grades responses against question checks decomposed from a constitution (Dineen et al., 2025). RaR (Gunjal et al., 2026) and Rubric Anchors (Huang et al., 2025) use rubric scores directly as RL rewards. Checklist feedback does the same with per-instruction checklists (Viswanathan et al., 2025), and follow-up work scales rubric generation (Liu et al., 2026b) or uses rubrics to scaffold exploration (Zhou et al., 2026). These methods all optimize against a fixed rubric per prompt and evaluate in-domain quality. Recent work addresses the brittleness of a fixed rubric by changing its *content*. RIFL, for example, appends a fixed set of negative rubrics that penalize known failure modes (He et al., 2026). Writing these negative rubrics adds authoring cost. We avoid this cost by keeping the rubric exactly as written and randomizing which criteria are scored. We also measure what a policy does *to* the rubric out of distribution.

Reward hacking and over-optimization. That optimizing a proxy degrades the true objective is one of the oldest observations in alignment (Amodei et al., 2016; Skalse et al., 2022; Pan et al., 2022). Gao et al. (2023) made this observation quantitative for learned reward models and showed that gold reward rises, peaks, and then decays as optimization proceeds. The mitigations developed in that literature operate on the reward model itself, by ensembling several such models (Coste et al., 2024; Eisenstein et al., 2024), averaging their weights (Rame et al., 2024), or disentangling the hackable length component (Chen et al., 2024). All of these train and serve extra reward models. Sampling sub-rubrics instead yields an implicit ensemble of sub-objectives with no extra judge calls. Out of distribution, we observe the same Gao-style rise and decay for rubric rewards. A concurrent work by Mahmoud et al. (2026) documents rubric reward hacking as a divergence between the training verifier and a stronger judge. They attribute it to verifier failure and rubric limitations. CHERRL (Wang et al., 2026) reproduces the same failure in a controlled setting. It injects known biases into the judge and detects when the policy exploits them. Both studies diagnose rubric reward hacking. We also evaluate rubric reward hacking out of distribution and propose a mitigation.

Regularization and reweighting. Neuron dropout prevents co-adaptation by ensuring that no single unit can be relied on (Srivastava et al., 2014). We apply the same idea to the training objective, so that the policy cannot rely on any single criterion. Dropout-style masking also appears elsewhere in the training loop. Vocabulary Dropout masks the proposer's output vocabulary to prevent diversity collapse in self-play (Dineen et al., 2026), while we mask the reward's criteria. Other work goes in the opposite direction. POW3R (Tyagi et al., 2026) *reweights* criteria by the rollout group's verdict variance, so the most discriminative criteria receive the most optimization pressure. Unlike POW3R, dropout limits how often any single criterion can influence the updates. The comparison in § 5.2 favors dropout. GDPO (Liu et al., 2026a) normalizes reward components separately within the group and targets signal resolution, not hacking. Dropout is orthogonal to both methods and only changes which criteria are scored.

3 Method

Preliminaries. For a query x and response y, a rubric consists of K criteria indexed by k, each with a weight w_k. A single judge call evaluates all criteria and returns $s_k(x, y) \in \{0, 1\}$, indicating whether criterion k is satisfied. Following RubricHub (Li et al., 2026), the reward is the normalized

weighted score

$$R(x, y) = \frac{\sum_{k=1}^{K} w_k s_k(x, y)}{\sum_{k=1}^{K} w_k}. \tag{1}$$

3.1 MEASURING REWARD HACKING

We define *reward hacking* as a policy learning to cater to the training judge's preferences. Such a policy receives a high reward from that judge but shows a large proxy–gold score gap on out-of-distribution (OOD) test scenarios. Our protocol uses two judges and an OOD evaluation set. Every 20 training steps, we evaluate the current policy on the OOD evaluation set and grade each response twice, once with the training (*proxy*) judge and once with a stronger, cross-family (*gold*) judge. We track four quantities: the *gold score* (our best available estimate of true quality), the *proxy−gold gap* (how much the proxy overrates the policy), the *overclaim fraction* (the fraction of criteria the proxy marks satisfied but gold rejects), and the *in-domain full-rubric reward* (what training optimizes). We use the in-domain full-rubric reward to check that a mitigation is not just slowing training. The gold judge is a stronger model, but it is not ground truth. We therefore never interpret the size of the gap. A judge with a fixed bias shifts its curve by a constant. It cannot, however, make the gold curve fall while the proxy curve rises. Our hacking signal is the divergence between the two curves during training, and a fixed bias in either judge can neither create nor remove it.

3.2 RUBRIC DROPOUT

The method has a single hyperparameter, the dropout fraction $f \in [0, 1)$. At each training step we drop a random f-fraction of the rubric's criteria, always keeping at least three, and compute the same reward on the kept criteria only. Because the number of dropped criteria is rounded and at least three are kept, the realized fraction is only approximately f, so we label the runs $f{\approx}30\%$ and $f{\approx}50\%$. Writing $m \in \{0, 1\}^K$ for the keep-mask ($m_k = 1$ means criterion k is kept),

$$\tilde{R}(x, y; m) = \frac{\sum_k m_k \, w_k \, s_k(x, y)}{\sum_k m_k \, w_k}. \tag{2}$$

Evaluation always scores the full rubric with Eq. (1). The judge grades all K criteria in one call, so the full-rubric reward remains available for logging at no extra cost.

3.3 GRPO WITH RUBRIC DROPOUT

We train with GRPO (Shao et al., 2024). For each prompt it samples a group of G responses from the previous policy and computes a reward R_i for each one. This reward is $R(x, y_i)$ for the base run and $\tilde{R}(x, y_i; m)$ under dropout. It then standardizes the rewards within the group into advantages $\hat{A}_i = (R_i - \mu)/\sigma$, where μ and σ are the mean and standard deviation of the group's rewards, before the usual clipped policy-gradient update.

If each rollout drew its own mask, the G responses would be graded on different sub-rubrics, and comparing them within the group would be meaningless. So we draw *one mask per rollout group*, and the mask changes only between training steps. In practice, the mask's random seed is a hash of the prompt's instance id and the global training step. All rollouts of a prompt within a step therefore share one mask, and the mask is redrawn at every step.

The whole group shares one mask, so any reward normalizer that depends only on the mask cancels in GRPO's standardized advantage. The choice of normalizer in Eq. (2) therefore does not affect the update (details in Appendix A). Over the mask distribution, dropout leaves the expected advantage unchanged up to a global scale that standardization removes. Dropout's effect is instead the noise it adds to the advantage. This noise is largest for responses whose advantage hinges on a single criterion and small for responses that are broadly better than their group. This is the anti-co-adaptation argument behind neuron dropout, applied here to rubric criteria. We treat this as motivating intuition rather than an established mechanism (§ 6).

3.4 Dropping correlated criteria together

The criterion-level dropout from § 3.2 reduces overfitting by randomly dropping individual rubric criteria. In some datasets, however, correlations among criteria or annotation artifacts may cause multiple criteria to encode overlapping signals. For these datasets, we also propose block dropout, which groups related criteria and drops them jointly. Suppose a requirement is repeated across r criteria, each dropped with probability f. Independent dropout removes the requirement entirely only with probability f^r, while dropping the r criteria as one block removes it with probability f. Since $f^r < f$ for $r > 1$, independent dropout is more likely to leave a redundant copy active. We therefore group criteria expressing similar requirements and apply one mask to each group. Offline, we partition each rubric into blocks by grouping criteria whose text is semantically similar (Appendix B; two worked examples in Figs. 12, 13). Let $\{\mathcal{B}_b\}_{b=1}^{B}$ denote the resulting partition, and let $M_b \in \{0, 1\}$ indicate whether block $\mathcal{B}_b$ is retained. Define the retained criteria as $\mathcal{K}_M = \bigcup_{b:M_b=1} \mathcal{B}_b$. The masked reward is

$$R_M(x, y) = \frac{\sum_{k\in\mathcal{K}_M} w_k s_k(x, y)}{\sum_{k\in\mathcal{K}_M} w_k}. \tag{3}$$

The mask is shared by all responses in a rollout group, so the denominator is constant within the group and cancels under GRPO normalization (Proposition 1). All criteria in a block are therefore added or removed as a unit, so redundant criteria cannot keep the same reward signal active. During training, we visit blocks in random order and drop them until reaching the target weight fraction.

4 Experiments

4.1 Setup

We train Qwen3 models ranging from 1.7B to 8B parameters (Yang et al., 2025) with GRPO, using 16 rollouts per prompt and a learning rate of 10^{-6}. Our primary experiments use Qwen3-8B on two independent train→eval pairs: RubricHub-Medical → HealthBench-Hard (Arora et al., 2025) and RubricHub-Science → ResearchQA (Yifei et al., 2026). HealthBench-Hard contains 1,000 prompts with physician-written rubrics. For ResearchQA, we report results on the 368 validation prompts that do not appear in the training data (Appendix B). The proxy judge is `gpt-4o-mini`, and the gold judge is `claude-sonnet-4-6`. Throughout, we mainly compare **base** (no dropout), **30%**, and **50%** dropout. The ablations (§ 5) include additional Medical runs with dropout fractions of 20–60% and with POW3R. Because hacking grows with training time, all cross-run results use a common 600-step horizon and a fixed comparison window over steps 400–600. We also compute matched-checkpoint win counts between runs on the same steps and prompts. Block dropout (§ 3.4) uses $f \approx 50\%$ with the same settings (§ 4.6). Full details are provided in Appendix B.

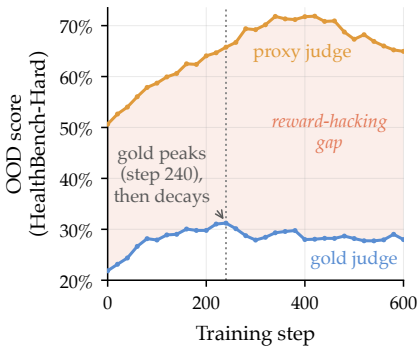


Figure 2: **The phenomenon.** Base Qwen3-8B on Medical, graded in-loop by both judges. The proxy climbs to 72% while gold peaks and declines.

4.2 Out of Distribution Reward Hacks

Figure 2 shows the base run on the Medical pair. In the first phase, proxy and gold rise together, indicating the policy is genuinely improving. Around step 240, gold peaks at 31.2% and then declines, while the proxy keeps rising to 72%. The proxy−gold gap widens from 29% to 44%. By step 600, the policy has regressed below its step-240 level, despite 360 more steps of "improvement" according to the proxy. The Science pair shows the same divergence with a steeper collapse. Gold falls 22 points from its peak within 600 steps (the base curves in Fig. 3 and Fig. 8). Gao et al. (2023) established this over-optimization pattern for learned reward models. We see the same pattern with rubric rewards and in out-of-distribution settings, where the damage is largest.

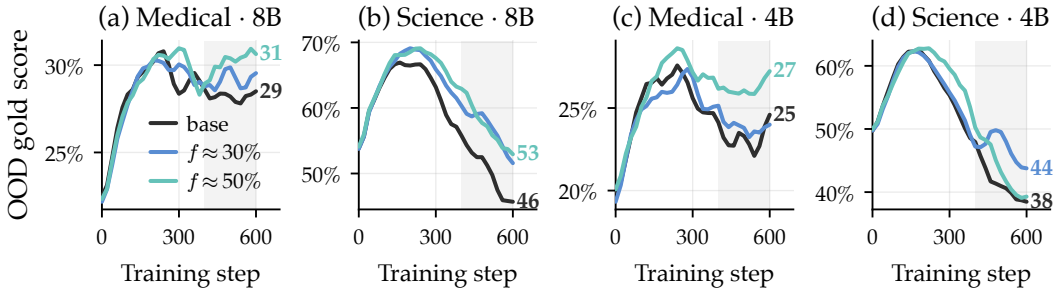


Figure 3: OOD gold score on both pairs at 8B and 4B (1.7B in Fig. 10). ■ base, ■ $f \approx 30\%$, ■ $f \approx 50\%$.

		Medical → HealthBench-Hard						Science → ResearchQA					
	Run	Peak	Gold	Δ	Proxy−gold	Overclaim	Train reward	Peak	Gold	Δ	Proxy−gold	Overclaim	Train reward
	base	31.2	28.2	0.0	40.3	40.4	**98.0**	67.5	50.4	0.0	37.2	37.3	94.8
8B	$f \approx 30\%$	30.9	29.2	+1.0	38.7	38.9	97.8	69.4	56.8	+6.4	29.9	29.8	**96.3**
	$f \approx 50\%$	**31.5**	**30.1**	**+2.0**	**38.4**	**37.2**	97.6	**69.8**	**57.4**	**+7.0**	**29.5**	**29.5**	95.7
	base	27.9	23.2	0.0	46.9	47.1	96.9	63.2	41.6	0.0	46.6	46.7	96.2
4B	$f \approx 30\%$	28.0	23.9	+0.7	**43.2**	45.9	**97.1**	62.6	**47.0**	**+5.3**	**40.1**	**40.3**	96.7
	$f \approx 50\%$	**28.8**	**26.2**	**+3.0**	44.4	**41.2**	96.8	**63.4**	43.7	+2.1	43.2	43.4	**96.8**
	base	19.8	16.3	0.0	**50.0**	57.6	**97.2**	50.3	34.4	0.0	48.1	48.3	**92.7**
1.7B	$f \approx 30\%$	**20.1**	**18.0**	**+1.7**	50.7	**53.2**	97.0	**53.6**	36.8	+2.4	45.2	45.4	92.5
	$f \approx 50\%$	19.8	16.3	0.0	52.5	55.3	96.6	52.3	**37.8**	**+3.4**	**43.9**	**44.1**	92.2

Table 1: Base and two dropout settings on both pairs at three model sizes. Gold (OOD) and Train reward (in-domain) are means over steps 400–600; Peak is the best gold score over all 600 steps. Δ is the point difference from each block's base. Bold marks column bests within each block.

4.3 Dropout improves OOD scores by 0.7–7.0 points

If these shortcuts stay stable because the rubric is fixed, resampling the rubric at every step should blunt the decline. Figure 3 and Table 1 compare base with the two dropout runs on Medical and Science over the same comparison window.

Rubric dropout mitigates reward hacking across scales. We observe the same qualitative effect across models ranging from 1.7B to 8B parameters on both Medical and Science (Figs. 3, 10, Table 1). During training, rubric dropout consistently improves OOD performance and does not lower the in-domain full-rubric reward. It also has an effect on the best OOD score attained during training. The trajectories begin to separate after their peaks, when the base runs undergo a larger decline.

The improvement extends to both gold reward and hacking measures. For every model size and domain, at least one dropout setting achieves a higher late-window Gold score than base. Peak scores are similar across runs and higher under dropout. The largest Gold difference between base and dropout occurs on Science at 8B. All three runs peak near step 200, but base then falls 21.5 points, to about 46% at step 600. The declines under $f \approx 30\%$ and $f \approx 50\%$ are 18.0 and 16.4 points, respectively, and both runs finish near 53%. Dropout also lowers overclaim in every model–domain setting and reduces the proxy–gold gap except on Medical at 1.7B (Fig. 8, Table 1). These results indicate that the higher Gold scores are accompanied by weaker evidence of reward hacking.

The preferred dropout fraction varies with model size and domain. At 8B, $f \approx 50\%$ performs best on both Medical and Science. At smaller scales the ordering is mixed. At 4B, $f \approx 50\%$ performs better on Medical, whereas $f \approx 30\%$ performs better on Science; at 1.7B, the preferred fraction again differs by domain. For the model sizes we tested, the higher dropout fraction appears more favorable at 8B. We treat this as a pattern in the present experiments only. The results do not establish a general relationship between model scale and the optimal dropout fraction. If we view the same trajectories as a tradeoff between quality and hacking, the dropout runs match or exceed the base run on gold at matched overclaim levels (Appendix B). § 6 discusses the mechanism behind the gain.

		Medical, by HealthBench axis			Science, by ResearchQA rubric type					
	Gold pass rate ↑	clinical	context-aw.	communic.	comparison	limitation	impact	other	example	citation
8B	base	50.2	31.3	35.4	41.9	52.4	48.9	50.9	59.5	0.5
	$f\approx30\%$	53.6	33.8	**37.6**	50.2	61.9	**58.3**	**56.5**	63.0	0.0
	$f\approx50\%$	**54.1**	**44.0**	35.9	**55.9**	**64.8**	58.0	55.5	**63.4**	0.5
4B	base	44.5	34.0	33.7	35.5	**51.0**	41.2	41.1	47.9	0.3
	$f\approx30\%$	44.4	26.7	**37.0**	**41.9**	47.4	**47.9**	**45.9**	**54.0**	**0.8**
	$f\approx50\%$	**48.5**	**34.5**	36.9	34.5	50.5	43.7	40.6	48.0	0.3

Table 2: **Gold pass rate by HealthBench axis and ResearchQA rubric type** (%, at step 600). Context-awareness is a subset of the clinical axes. Bold marks the best run per column within each block. Criteria with multiple type tags are counted under each of their types.

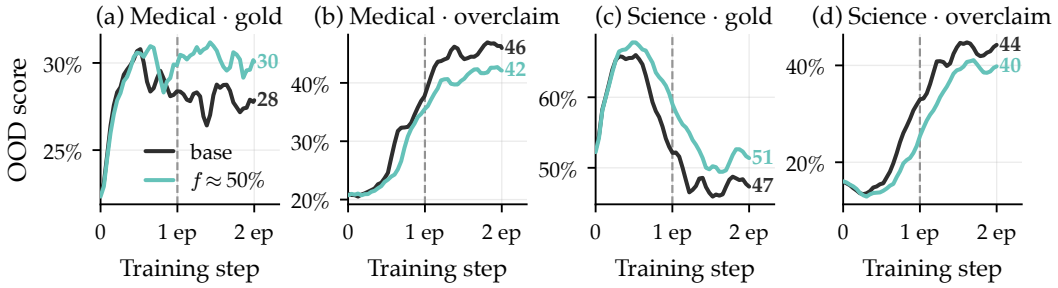


Figure 4: **Base vs $f\approx50\%$ over two full epochs at 8B.** Gold and overclaim on the Medical and Science pairs. The dashed line marks one epoch; curves are rolling means, with end values annotated.

4.4 Criterion-level breakdown

To see where the aggregate changes come from, we break down the step-600 responses by criterion. Both judges grade every criterion for each run, and we classify each criterion accepted by the proxy as either confirmed by the gold judge or overclaimed (Fig. 14 in the appendix). At matched proxy pass rates (within 1.3 points everywhere), both dropout runs have a higher gold pass rate and less overclaim. Gold pass rate increases and overclaim decreases monotonically with the dropout fraction. At $f\approx50\%$ the gains in gold pass rate are up to $+3.6$ points on Medical and $+7.3$ on Science. The proxy's error is also almost entirely one-sided. Underclaim is never above 3.1%, so the gap comes from systematic over-crediting. When we group criteria by their benchmark-provided tags (Table 2), the gains concentrate in criteria that ask for more substantive content. On Medical, the clinical axes (accuracy, completeness, context-awareness) gain the most, while the communication axes gain less. The largest single jump is on context-awareness at $f\approx50\%$, which is also the axis where the base run scores lowest. On Science the analytical types (comparison, limitation, impact) gain two to three times as much as the example and generic ones at $f\approx50\%$. The analytical-over-generic pattern also holds at $f\approx30\%$. In our runs, dropout preserves quality on the same expensive criteria that degrade first under hacking. At 4B the same breakdown picks out the better fraction in each domain ($f\approx50\%$ on Medical, $f\approx30\%$ on Science). That fraction gains 3.9 (Medical) and 5.2 (Science) points of gold pass rate and lowers overclaim by 3.2 and 5.5. The other fraction stays near base. The matched checkpoint is step 600, the edge of the comparison window, so these numbers break down the results in Table 1 and do not re-estimate them.

4.5 The separation persists for two epochs

All results above use a 600-step training horizon, corresponding to roughly 1.3 epochs. This leaves open the possibility that dropout merely postpones collapse. We therefore extended the baseline and the $f \approx 50\%$ dropout run to two full epochs on both pairs. The gap persists throughout extended training (Fig. 4). After the peak, dropout consistently preserves more gold behavior and produces fewer overclaims on both Medical and Science. Because the two runs achieve similar peak scores, the main benefit of dropout lies in greater resistance to degradation during continued training.

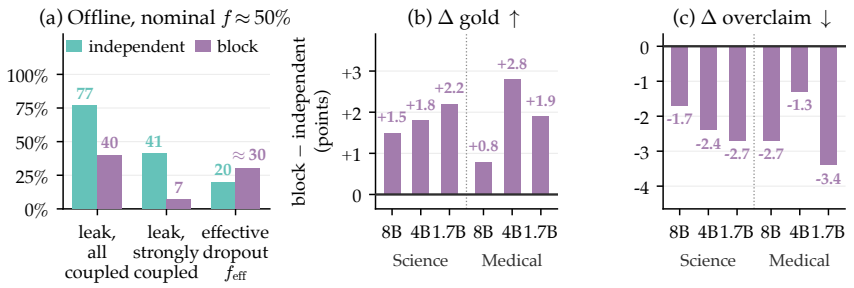


Figure 5: Block dropout: offline leak and training deltas. (a) Offline, on 8B Science verdicts at nominal $f \approx 50\%$. (b, c) Block − independent in training, window means over steps 400–600.

4.6 Block dropout narrows the leak

We first measure the leak offline, replaying both kinds of mask at nominal $f \approx 50\%$ on logged 8B Science verdicts from a deep-plateau checkpoint (Table 4, left; Fig. 5a). Leakage measures how much correlated reward signal survives a mask. For each dropped criterion we take its largest verdict correlation ϕ with any retained criterion, then average over dropped criteria. Higher leakage means a near-copy of the dropped signal is still being rewarded. We use leakage only to evaluate a mask; the blocks themselves are built from text similarity alone (§ 3.4). Among criteria whose verdicts vary within the group, leakage falls from 41% to 7%. That row, however, uses oracle blocks. They are built from the verdicts themselves ($\phi \geq 0.6$), not from the text blocks used in training, so the row only bounds what correlation-aware blocks could recover. The effective dropout rate is the share of reward variance the mask removes; it rises from 20% to about 30% (Appendix B).

In training (Table 4, Fig. 5), block dropout consistently improves gold and reduces overclaim relative to independent dropout across both Science and Medical and at every model size. Effective dropout alone does not explain these gains. At matched effective dropout, performance depends on which criteria are dropped together. We discuss the reason in § 6.

5 Ablations

5.1 The dropout fraction

We sweep $f \in \{20, 30, 40, 50, 60\}\%$ (Fig. 6; full numbers in Table 3 in the appendix). Best-checkpoint gold is essentially tied across all runs (30.6–31.5% against base's 31.2%), so the differences come from how gold holds up under continued training. Everything from 20% to 50% is at or above base. The best window mean is at 50% (+2.0). The sign flips only at 60% (−0.5). We expected this failure once the surviving sub-rubrics no longer cover all aspects of quality. The training trajectories (Fig. 9, appendix) confirm that these differences come from the post-peak phase and not from learning speed. They also show that no run gives up in-domain full-rubric reward. Any dropout fraction in the 30–50% range gives most of the benefit.

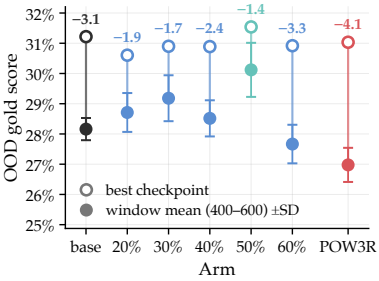


Figure 6: **Sweeping the dropout fraction (Medical).** The post-peak drop is smallest at $f \approx 50\%$.

5.2 Reweighting is not a substitute for dropping

The natural alternative to dropping criteria, reweighting toward the informative ones, performs worse out of distribution than no intervention at all. POW3R has the lowest OOD gold score of any run (27.0%) and loses to base at all matched checkpoints. Its overclaim fraction, 42.2%, is also the highest, above even base's 40.4%. One plausible mechanism is that reweighting by verdict variance concentrates pressure on exactly the criteria the policy is learning to game. This would then strengthen the loop that dropout weakens. POW3R's best checkpoint (31.0%) matches base's (31.2%), so the deficit comes from how its score decays after the peak. We test it OOD at its published defaults and do not evaluate its in-distribution claims. Both the sweep and the POW3R

comparison indicate that OOD robustness improves when optimization pressure is spread across criteria and degrades when it is concentrated.

6 Discussion

Where the gain comes from. § 4.4 shows that the gain comes from the prompt-specific, content-heavy criteria, which dropout keeps satisfied after the base policy stops satisfying them. We attribute the gain on these criteria to the same mechanism that makes neuron dropout work. In that setting, dropout prevents units from co-adapting, that is, from relying on specific partners that are always present (Srivastava et al., 2014). With the rubric resampled at every step, no criterion is reliably present. A shortcut that satisfies a generic criterion, such as a fixed template, is rewarded only on the steps where that criterion is kept, so it is never reliably reinforced. To score reliably, the policy has to answer the prompt, which is what the prompt-specific criteria measure.

A better tradeoff, not the same path. We compare the two-epoch runs of § 4.5 at matched overclaim. The comparison is informative only after one epoch, once each prompt has been scored under different masks. Past that point the $f{\approx}50\%$ run holds more gold than base at every matched point on Medical ($+0.2$ to $+2.5$ points) and at nearly every matched point on Science ($+1.6$ on average). Dropout reaches a better gold-versus-overclaim tradeoff rather than the same one later. With one run per condition, we read this as supporting evidence.

Mask structure matters. At the same nominal dropout fraction, block dropout outperforms independent dropout on both domains at every model size (§ 4.6). The difference comes from redundancy between criteria. When two criteria encode the same requirement, independent dropout usually removes only one of them, and the survivor keeps rewarding the same behavior. Dropping them together removes that signal, so the policy cannot keep exploiting it through a near-duplicate. Block masks also remove a larger share of reward variance (Table 4). The gains therefore depend on which criteria are dropped together as well as on how many.

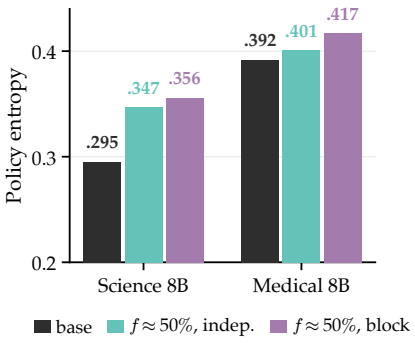


Figure 7: **Policy entropy at 8B**, mean over training steps 300–600. Both dropout variants stay above base on both pairs.

Output diversity. Late in training, base responses consist largely of template scaffolding (repeated headers and placeholder lists) that satisfies structural criteria but adds little content. On Science, every late-stage base response reaches the generation length limit. The dropout policies stay more diverse than the base policy. Single-step entropy is noisy, so we average it over training steps 300–600. Gold peaks near step 240, and base collapses during this later window. Over that span, policy entropy on Science is 0.295 for base, 0.347 for independent dropout and 0.356 for block dropout. Medical shows the same ordering with a smaller gap (Fig. 7). The base policy's collapse onto templates and its low entropy indicate overfitting to the fixed rubric. The dropout runs do not show this overfitting pattern. Their outputs stay varied and do not converge on a single template that satisfies the proxy.

7 Conclusion

Rubric-based RL can overfit a fixed reward proxy. Across both domains, OOD gold scores peak and then decline even as proxy scores continue to rise. Rubric Dropout counters this by randomly omitting criteria during training while keeping the full rubric for evaluation. Across models from 1.7B to 8B, it improves OOD gold scores and reduces overclaim without lowering in-domain reward. Dropout rates between 30% and 50% work well across most settings. By contrast, variance-based reweighting performs similar to the no-dropout baseline. Block dropout provides further gains by dropping semantically correlated criteria together. These results suggest that changing which criteria influence each update makes stable rubric shortcuts harder to exploit while promoting greater rollout diversity.

AI USE STATEMENT

Large language models were used solely for spelling and grammar checks during manuscript preparation. They were not used to generate scientific content or conduct the research. The authors reviewed the final manuscript and take full responsibility for its content.

ETHICS STATEMENT

This work uses medical prompts and benchmarks as a testbed for reward hacking. We do not release a model intended for clinical use. The failure mode we document, a policy that satisfies its training judge while true quality degrades, is itself a deployment risk for rubric-trained models. Measuring it is one of the aims of this work. All judges are commercial APIs used under their terms. No human-subjects data were collected.

REPRODUCIBILITY STATEMENT

All numbers derive from per-step trajectories logged during training and from the judges' grades of saved model outputs. The window means, win counts, and figures are regenerated by the released scripts from the cached data. Code and scripts are available in the anonymized repository. Models, data, judges, hyperparameters, and the dropout procedure, including block construction, are specified in § 3 and Appendix B.

REFERENCES

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.

Rahul K. Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, Johannes Heidecke, and Karan Singhal. HealthBench: Evaluating large language models towards improved human health. *arXiv preprint arXiv:2505.08775*, 2025.

Lichang Chen, Chen Zhu, Jiuhai Chen, Davit Soselia, Tianyi Zhou, Tom Goldstein, Heng Huang, Mohammad Shoeybi, and Bryan Catanzaro. ODIN: Disentangled reward mitigates hacking in RLHF. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 7935–7952. PMLR, 2024.

Thomas Coste, Usman Anwar, Robert Kirk, and David Krueger. Reward model ensembles help mitigate overoptimization. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

Jacob Dineen, Aswin Rrv, Qin Liu, Zhikun Xu, Xiao Ye, Ming Shen, Zhaonan Li, Shijie Lu, Chitta Baral, Muhao Chen, and Ben Zhou. QA-LIGN: Aligning LLMs through constitutionally decomposed QA. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 20619–20642, 2025.

Jacob Dineen, Aswin RRV, Zhikun Xu, and Ben Zhou. Vocabulary dropout for curriculum diversity in LLM co-evolution. In *Conference on Language Modeling (COLM)*, 2026.

Jacob Eisenstein, Chirag Nagpal, Alekh Agarwal, Ahmad Beirami, Alex D'Amour, DJ Dvijotham, Adam Fisch, Katherine Heller, Stephen Pfohl, Deepak Ramachandran, Peter Shaw, and Jonathan Berant. Helping or herding? Reward model ensembles mitigate but do not eliminate reward hacking. In *First Conference on Language Modeling (COLM)*, 2024. URL https://openreview.net/forum?id=5u1GpUkKtG.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10835–10866. PMLR, 2023.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In *International Conference on Learning Representations (ICLR)*, pp. 127924–127945, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z.

Yun He, Wenzhe Li, Hejia Zhang, Songlin Li, Karishma Mandyam, Sopan Khosla, Yuanhao Xiong, Nanshu Wang, Xiaoliang Peng, Beibin Li, Shengjie Bi, Shishir G Patil, et al. AdvancedIF: Rubric-based benchmarking and reinforcement learning for advancing LLM instruction following. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 18003–18022, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.820. URL https://aclanthology.org/2026.acl-long.820/.

Zenan Huang, Yihong Zhuang, Guoshan Lu, Zeyu Qin, Haokai Xu, Tianyu Zhao, Ru Peng, Jiaqi Hu, Zhanming Shen, Xiaomeng Hu, et al. Reinforcement learning with rubric anchors. *arXiv preprint arXiv:2508.12790*, 2025.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, et al. Tülu 3: Pushing frontiers in open language model post-training. In *Conference on Language Modeling (COLM)*, 2025.

Sunzhu Li, Jiale Zhao, Miteto Wei, Huimin Ren, Yang Zhou, Jingwen Yang, Shunyu Liu, Kaike Zhang, and Wei Chen. Rubrichub: A comprehensive and highly discriminative rubric dataset via automated coarse-to-fine generation. *arXiv preprint arXiv:2601.08430*, 2026. URL https://arxiv.org/abs/2601.08430.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, Yejin Choi, Jan Kautz, and Pavlo Molchanov. GDPO: Group reward-decoupled normalization policy optimization for multi-reward RL optimization. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026a.

Tianci Liu, Ran Xu, Tony Yu, Ilgee Hong, Carl Yang, Tuo Zhao, and Haoyu Wang. OpenRubrics: Towards scalable synthetic rubric generation for reward modeling and LLM alignment. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17417–17437, San Diego, California, United States, 2026b. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.791.

Anas Mahmoud, MohammadHossein Rezaei, Zihao Wang, Anisha Gunjal, Bing Liu, and Yunzhong He. Reward hacking in rubric-based reinforcement learning. *arXiv preprint arXiv:2605.12474*, 2026.

Alexander Pan, Kush Bhatia, and Jacob Steinhardt. The effects of reward misspecification: Mapping and mitigating misaligned models. In *International Conference on Learning Representations (ICLR)*, 2022.

Alexandre Rame, Nino Vieillard, Leonard Hussenot, Robert Dadashi-Tazehozi, Geoffrey Cideron, Olivier Bachem, and Johan Ferret. WARM: On the benefits of weight averaged reward models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42048–42073. PMLR, 2024.

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.

Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, 2022.

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

Utkarsh Tyagi, Xingang Guo, MohammadHossein Rezaei, Daniel George, Anas Mahmoud, Jackson Lee, Bing Liu, and Yunzhong He. Not every rubric teaches equally: Policy-aware rubric rewards for RLVR. *arXiv preprint arXiv:2605.20164*, 2026.

Vijay Viswanathan, Yanchao Sun, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. Checklists are better than reward models for aligning language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pp. 127195–127221, 2025.

Xuekang Wang, Zhuoyuan Hao, Shuo Hou, Hao Peng, Juanzi Li, and Xiaozhi Wang. Reproducing, analyzing, and detecting reward hacking in rubric-based reinforcement learning. *arXiv preprint arXiv:2606.04923*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Li S. Yifei, Allen Chang, Chaitanya Malaviya, and Mark Yatskar. ResearchQA: Evaluating scholarly question answering at scale across 75 fields with survey-mined questions and rubrics. *Transactions of the Association for Computational Linguistics*, 14:1344–1368, 2026. doi: 10.1162/TACL.a.732.

Ming Zhang, Yujiong Shen, Zelin Li, Huayu Sha, Binze Hu, Yuhui Wang, Chenhao Huang, Shichun Liu, Jingqi Tong, Changhao Jiang, Mingxu Chai, Zhiheng Xi, Shihan Dou, Tao Gui, Qi Zhang, and Xuanjing Huang. LLMEval-Med: A real-world clinical benchmark for medical LLMs with physician validation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 4888–4914, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.263. URL https://aclanthology.org/2025.findings-emnlp.263/.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, volume 36, 2023.

Yang Zhou, Sunzhu Li, Shunyu Liu, Wenkai Fang, Kongcheng Zhang, Jiale Zhao, Jingwen Yang, Yihe Zhou, Jianwei Lv, Tongya Zheng, Hengtong Lu, Wei Chen, et al. Breaking the exploration bottleneck: Rubric-scaffolded reinforcement learning for open-ended LLM reasoning. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026.

A Analysis of Group-Shared Rubric Dropout

Group-shared Rubric Dropout is well-behaved under GRPO for two reasons. Fix one group and one mask m, and write $s_{k,i}$ for the verdict of criterion k on response y_i and $c_i = \sum_k m_k w_k s_{k,i}$ for the masked score of y_i before any normalization. Here mean_j and std_j run over the group's responses $j = 1, \ldots, G$.

Proposition 1 (The normalizer cancels)**.** *Because the mask is shared by the whole group, any positive normalizer Z that depends only on the mask (the kept weight $\sum_k m_k w_k$ of Eq.* (2)*, its expectation, or no normalizer at all) is the same constant for every response in the group. Whenever the group's masked scores are not all equal, the normalizer therefore cancels in the standardized advantage:*

$$\hat{A}_i(m) = \frac{c_i/Z - \mathrm{mean}_j(c_j/Z)}{\mathrm{std}_j(c_j/Z)} = \frac{c_i - \mathrm{mean}_j(c_j)}{\mathrm{std}_j(c_j)}.$$

Under this standardization, the advantage depends only on which criteria the mask keeps. In particular, there is no normalizer to tune.

The second reason is the intuition behind the method. In expectation, dropout only rescales the advantage, so any effect it has comes from the noise it injects. This noise is largest for responses whose advantage depends on a single criterion. We take the mask entries to be i.i.d., keeping each criterion with probability $1 - f$. Center each verdict within the group, $\delta_{k,i} = s_{k,i} - \bar{s}_k$ with $\bar{s}_k = \frac{1}{G}\sum_{j=1}^{G} s_{k,j}$, and consider the un-normalized advantage $u_i(m) = \sum_k m_k w_k \delta_{k,i}$. With the full rubric this is the centered reward up to the constant total weight, $u_i(\mathbf{1}) = \big(\sum_k w_k\big)(R_i - \mu)$, and group standardization ignores that constant (Proposition 1). On our positive-weight training rubrics the clip in Eq. (1) is inactive.

Observation 1 (Dropout is a variance regularizer)**.** *Over the i.i.d. mask distribution,*

$$\mathbb{E}_m[u_i(m)] = (1 - f)\, u_i(\mathbf{1}), \qquad \mathrm{Var}_m[u_i(m)] = f(1 - f) \sum_k w_k^2\, \delta_{k,i}^2.$$

In expectation, the only effect of dropout on the un-normalized advantage is a global scale $1 - f$, which group standardization removes. Dropout therefore acts only through the variance term, and this term shows which responses get a noisy signal. It is largest exactly when the advantage hinges on one high-weight criterion (one large $w_k \delta_{k,i}$), and smallest when a response is broadly better than its group (many small contributions). Since gradient noise slows SGD's progress along a direction, dropout slows progress toward single-criterion exploits more than toward broad improvement. This is the same anti-co-adaptation logic as neuron dropout, applied to the reward. We treat it as motivating intuition only; it is not an established mechanism. The experiments establish where dropout helps, and § 6 discusses what they can and cannot say about why. There are two caveats to Observation 1. In practice we drop a fixed f-fraction rather than taking i.i.d. draws. Each criterion is then kept with probability $1 - f$, so the expectation is unchanged, and the variance only acquires small cross-terms from the negative mask covariances. And because GRPO's division by the group standard deviation also depends on the mask, Observation 1 describes the advantage before that division.

Observation 1 also shows at which dropout fraction the injected variance is largest. The factor $f(1 - f)$ peaks at $f = 1/2$, so under the variance-regularization reading the effect is strongest near 50% dropout. The Medical sweep agrees. Its best window mean is at $f \approx 50\%$, and the benefit collapses at 60%, where the kept sub-rubric no longer covers the criteria that define quality in the full rubric (§ 5.1). We note the agreement but do not rely on it. We treat the variance-regularization reading as motivation only, since the sweep cannot separate it from the coverage effect that dominates at large f.

A.1 Dropping correlated criteria together

Independent criterion dropout may leave redundant reward signals active. Suppose that r criteria produce identical verdicts and are independently dropped with probability f. Their common signal is removed only if all r criteria are dropped. If they belong to the same block, dropping that block

removes the signal at once:

$$\Pr(\text{common signal is removed}) = \begin{cases} f^r, & \text{independent criterion dropout,} \\ f, & \text{block dropout,} \end{cases} \tag{4}$$

where the second line assumes that the r criteria share a block. This calculation uses Bernoulli dropout for illustration. With exactly m of K criteria dropped uniformly without replacement, the first probability is $\binom{K-r}{m-r}/\binom{K}{m}$.

We next characterize how block dropout affects within-group reward variation. For a rollout group $\{y_i\}_{i=1}^G$, write $s_{ik} = s_k(x, y_i)$ and define

$$z_{ik} = s_{ik} - \frac{1}{G}\sum_{j=1}^{G} s_{jk}, \qquad T_{ib} = \sum_{k \in \mathcal{B}_b} w_k z_{ik}, \tag{5}$$

where $\{\mathcal{B}_b\}_{b=1}^B$ is a partition of the rubric. Given a block mask $M = (M_1, \ldots, M_B)$, the centered reward numerator is

$$U_i(M) = \sum_{b=1}^{B} M_b T_{ib}. \tag{6}$$

The retained-weight normalizer is shared by every response in the rollout group and therefore cancels under GRPO normalization (Proposition 1).

Consider the Bernoulli approximation in which each block is retained independently with probability $q = 1 - f$. Define Var_i and Cov_i using the population normalization $1/G$. The mask-induced variance, averaged over responses, is

$$\begin{aligned} V_{\text{block}} &\coloneqq \frac{1}{G}\sum_{i=1}^{G} \mathrm{Var}_M[U_i(M)] \\ &= V_{\text{ind}} + 2q(1-q)\sum_{b=1}^{B} \sum_{\substack{k<\ell \\ k,\ell \in \mathcal{B}_b}} w_k w_\ell \,\mathrm{Cov}_i(s_{ik}, s_{i\ell}), \end{aligned} \tag{7}$$

where

$$V_{\text{ind}} = q(1-q)\sum_{k=1}^{K} w_k^2 \,\mathrm{Var}_i(s_{ik}) \tag{8}$$

is the corresponding quantity for independent criterion dropout. Hence $V_{\text{block}} \geq V_{\text{ind}}$ when the within-block verdict covariances are nonnegative. Block dropout therefore perturbs correlated reward signals as a group, whereas under independent criterion dropout part of such a group often stays active. Negative within-block covariance can reverse the variance inequality, which motivates using a conservative similarity threshold and splitting clusters that exceed the size cap.

We construct the blocks once, offline, using criterion-text similarity. At each training step, blocks are sampled without replacement and dropped sequentially toward a target fraction f of the total rubric weight, subject to retaining at least three criteria. Because blocks are indivisible, the realized retained-weight fraction can deviate slightly from its target; for $f \approx 50\%$, it is around 50% on average. Every response in a rollout group shares the same mask. Block dropout adds one offline clustering pass and requires no extra judge calls. § 4.6 reports the results.

B Experimental Setup and Details

Policy and algorithm. We train Qwen3-8B (Yang et al., 2025) with GRPO (Shao et al., 2024), using 16 rollouts per prompt, a learning rate of 10^{-6}, and FSDP. Qwen3-4B and Qwen3-1.7B are trained with the same recipe. All 8B runs train to at least 600 steps except the $f{\approx}60\%$ run, which ends at step 559 with its last logged evaluation at step 540. The smaller sizes are described under Training runs. All comparisons use the common 600-step horizon.

Training data. RubricHub-Medical consists of medical prompts with weighted rubrics of 8–67 criteria (mean ∼30). RubricHub-Science is built the same way and has 29,418 prompts with a mean of ∼27 criteria. The proxy judge for both is `gpt-4o-mini`.

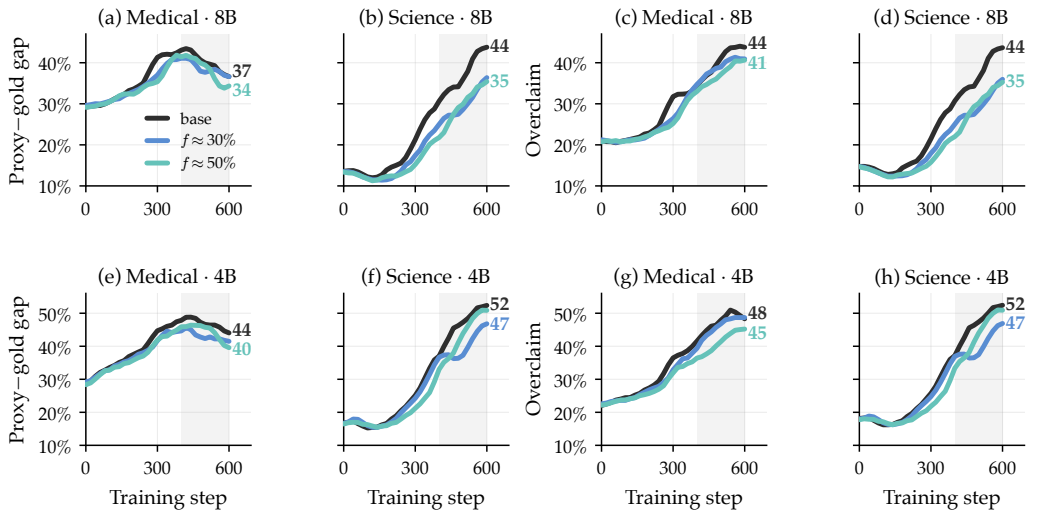


Figure 8: **Dropout lowers both hacking measures in the Medical and Science domains at 8B and 4B** (for 1.7B, see Fig. 10). Proxy−gold gap and overclaim fraction. ■ base, ■ $f \approx 30\%$, ■ $f \approx 50\%$.

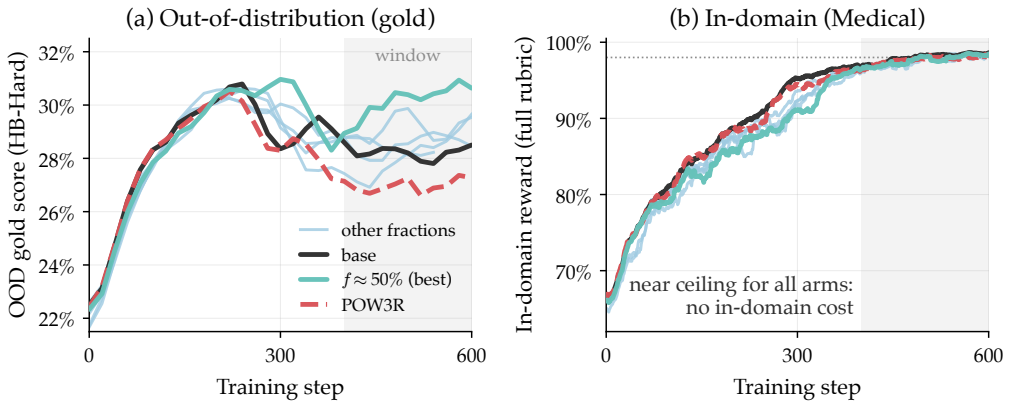


Figure 9: **Trajectory view of the sweep (Medical).** (a) OOD gold score, 3-point rolling means of the 20-step evaluations. (b) In-domain full-rubric reward, 25-step rolling mean of the per-step training signal. Window shaded.

OOD evaluation. The Medical pair is evaluated on the Hard split of HealthBench (Arora et al., 2025), which has 1,000 prompts with a mean of ∼11.9 criteria each. This split shares no prompts with RubricHub-Medical. The Science pair is evaluated on the ResearchQA validation split (Yifei et al., 2026). Since RubricHub-Science is built from ResearchQA, we score on the 368 prompts that never occur in training, with a mean of ∼7.4 equal-weight criteria per prompt. The evaluation rubrics share no criteria with the training rubrics on any prompt. Both pairs are evaluated in-loop every 20 steps and graded by the proxy (`gpt-4o-mini`) and gold (`claude-sonnet-4-6`) judges as described in § 3.1. The in-loop Science evaluation runs on the full 703-prompt split, and the reported Science numbers come from scoring the saved responses on the 368-prompt subset. The full split shows the same trends, slightly weaker.

Training runs. On Medical we run base, $f \in \{20, 30, 40, 50, 60\}\%$, and POW3R. On Science we run base and $f \in \{30, 50\}\%$. For Qwen3-4B and Qwen3-1.7B we run base and $f \in \{30, 50\}\%$ on both pairs, with the same recipe. For 4B and 1.7B we report in-loop trajectory statistics, and we evaluate the Science runs on the full 703-prompt validation split. Preemptions fragmented the 1.7B logs, and we merge the segments by step. The 1.7B Medical $f \approx 30\%$ run stops at step 599 with its last eval at step 580, so its window statistics cover steps 400–580; the other 1.7B Medical runs cover the full window. Figure 10 shows the 1.7B trajectories. The two-epoch continuations of

Run	Gold ↑	Δ ↑	Proxy−gold ↓	Overclaim ↓	Train reward ↑
base	28.2	0.0	40.3	40.4	**98.0**
$f{\approx}20\%$	28.7	+0.6	40.2	39.5	97.7
$f{\approx}30\%$	29.2	+1.0	38.7	38.9	97.8
$f{\approx}40\%$	28.5	+0.4	41.3	38.5	97.6
$f{\approx}50\%$	**30.1**	**+2.0**	**38.4**	**37.2**	97.6
$f{\approx}60\%$	27.7	−0.5	42.6	37.8	97.3
POW3R	27.0	−1.2	40.2	42.2	97.7

Table 3: **Ablations on the Medical pair** (window means over steps 400–600, values in %, Δ in points vs base). Bold marks the best value per column. The $f{\approx}50\%$ row is tinted with its figure color.

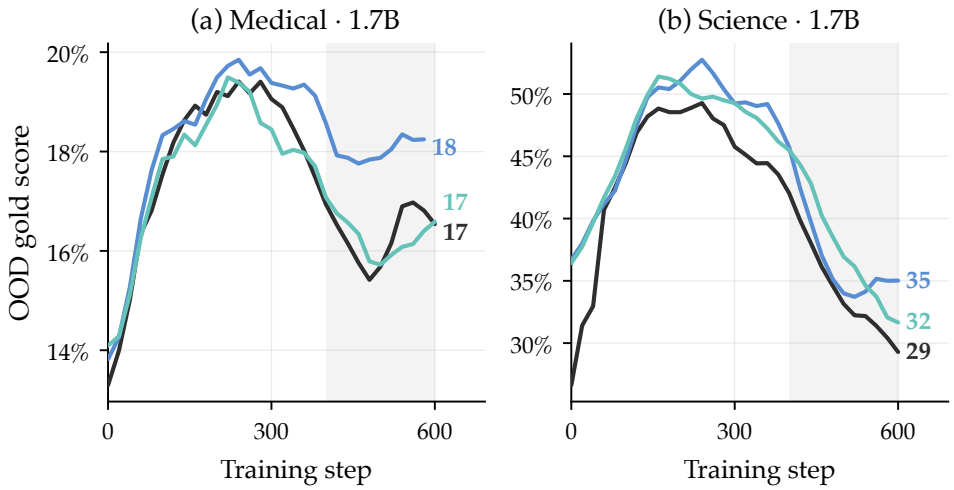


Figure 10: **Qwen3-1.7B trajectories on both pairs.** ■ base, ■ $f{\approx}30\%$, ■ $f{\approx}50\%$. Rolling means, comparison window shaded.

§ 4.5 resume each 8B run from its deepest checkpoint. At the resume boundary, the continuation's gold score agrees with the original run's gold score to within 0.7 points, and the merged curves have no eval gaps. The block-dropout runs (§§ 3.4, 4.6) use $f{\approx}50\%$ and the recipe above on both pairs at 8B, 4B, and 1.7B. We score them in-loop like the other small-model runs. There is one run per cell except Medical 4B, where both dropout kinds have two runs. In Table 4 and Fig. 5 we compare the later run of each dropout kind (window gold 25.9 for block vs 23.1 for independent). All four runs are drawn in Fig. 11, with the earlier run of each kind dashed (block 25.1, independent 26.2). Figure 11 shows four of the six domain-by-size cells.

Block construction. We embed each criterion with `all-mpnet-base-v2` and compute cosine similarity between normalized embeddings. We connect pairs with similarity at least 0.62 and take connected components as blocks. The threshold 0.62 is a conservative heuristic. We keep it fixed and do not tune it on downstream results. In a preliminary analysis at 0.60, components often grouped criteria that shared a broad topic but were not redundant requirements. Because connected components can also grow through transitive links, we split components larger than $\max(3, \mathrm{round}(0.4K))$ into singletons. The fixed size cap also limits over-merging of topically similar criteria. As with the threshold, we did not select the cap using downstream training results. At training time the group-shared mask drops whole blocks until the dropped fraction reaches f. It skips any block whose removal would leave fewer than three kept criteria. The last dropped block can overshoot the target, so the realized kept weight is around 50% rather than exactly the nominal value. Evaluation is unchanged.

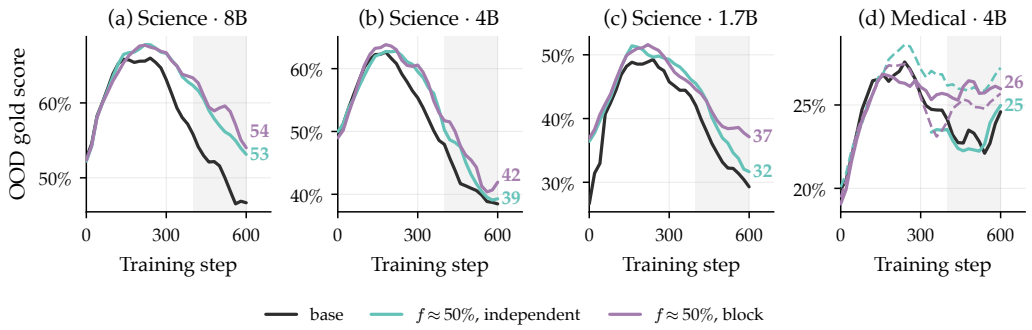


Figure 11: **Block dropout against independent dropout** (Medical 4B has two runs of each dropout kind; solid lines are the runs compared in Table 4, dashed lines the others). ■ base, ■ $f \approx 50\%$ independent, ■ $f \approx 50\%$ block. Rolling means, comparison window shaded.

Offline, nominal $f \approx 50\%$	Independent	Block
Leakage, all coupled pairs	77	40
Leakage, varying criteria ($\phi \geq 0.6$ blocks)	41	7
Effective dropout rate f_{eff}	20	$\approx$30

Training, block − independent	Δ gold	Δ overclaim
Science 8B	+1.5	−1.7
Science 4B	+1.8	−2.4
Science 1.7B	+2.2	−2.7
Medical 8B	+0.8	−2.7
Medical 4B	+2.8	−1.3
Medical 1.7B	+1.9	−3.4

Table 4: **Block dropout: offline leak and training deltas.** Left: logged 8B Science verdicts at nominal $f \approx 50\%$, in %. Leakage: mean over dropped criteria of the largest verdict correlation ϕ with a surviving criterion. The second row counts only criteria whose verdicts vary within the group. f_{eff}: share of reward variance the mask removes, measured for independent masks and derived for block masks (Appendix B). Right: block minus independent dropout, both at $f \approx 50\%$, window means over steps 400–600 in points. Figure version in Fig. 5.

Two block examples. Figures 12, 13 show the text-similarity blocks for one rubric from each pair, produced by the procedure above and used as is in training. Criteria are shown verbatim; the italic label is our gloss of the requirement a block shares. Within a block the criteria restate one requirement in different words, so dropping them independently would usually leave a near-copy active. No verdicts or scores enter the grouping. Because whole blocks are dropped, a single mask can remove several criteria at once; the rule that every rubric keeps at least three criteria therefore binds more often on such rubrics than under independent dropout, and the kept weight lands near, rather than exactly at, the nominal 50%.

Leakage and effective dropout rate. The left panel of Table 4 uses saved deep-plateau responses of the 8B Science base and $f \approx 50\%$ runs to 8 ResearchQA evaluation prompts, 16 rollouts each, re-judged by the proxy judge. For two criteria of one rubric, ϕ is the correlation of their binary verdicts across the prompt's rollouts; it is defined only when both verdicts vary within the group, which are also the only criteria that carry GRPO signal. The *leakage* of a mask is the mean, over its dropped varying criteria, of the largest ϕ (clipped at zero) to any surviving criterion. In the second row the blocks come from the verdicts themselves ($\phi \geq 0.6$ components), not from the text blocks used in training, so the panel bounds what a correlation-aware mask can recover. The *effective dropout rate* is $f_{\text{eff}} = 1 - \rho^2$, with ρ the correlation between masked and full-rubric reward across a group's rollouts: the fraction of reward variance the mask removes, which would equal f if criteria were uncorrelated. It is measured for independent masks by sampling masks on the logged verdicts, and derived for block masks as $f\,(1 - \text{leakage})$ from the first row.

Matched-overclaim comparison. We compare gold at matched overclaim levels on the 600-step 8B runs, after the hacking onset. At these levels, the dropout runs match or exceed base on gold. On Medical, at 40% overclaim, base has 28.5% gold and $f \approx 50\%$ has 31.3%. On Science, at 35% overclaim, the numbers are 50.8% versus 52.5%. Within each domain, at the same overclaim level, the dropout runs reach a higher gold score than base.

Example A: Science (RubricHub-Science). Prompt: "When a standard die (with numbers 1 to 6) is rolled, what is the probability that the number rolled is neither a multiple of 5 nor an odd number?" (16 criteria: five blocks, four singletons)

Block 1 *states the sample space and uses correct terminology*
- Explicitly lists the six possible outcomes of a standard die: {1, 2, 3, 4, 5, 6}.
- Uses correct and consistent terms such as 'standard die', 'outcome', 'odd', 'multiple of 5', and 'probability' throughout.

Block 2 *identifies the multiple-of-5 and odd outcomes*
- Correctly identifies 5 as the only multiple of 5 within the sample space.
- Correctly identifies the odd outcomes in the sample space as {1, 3, 5}.

Block 3 *interprets "neither . . . nor" and builds the exclusion set*
- Interprets the phrase 'neither a multiple of 5 nor an odd number' to exclude all numbers that are multiples of 5 and all odd numbers.
- Identifies all numbers to be excluded: the odd set {1, 3, 5} and the multiple-of-5 set {5}.
- Explicitly states that the element appearing in both the 'multiple of 5' set and the 'odd' set is counted only once, and provides a brief justification for this single counting.

Block 4 *applies the probability formula and gives 1/2*
- Computes probability as (favorable outcomes)/(total outcomes) and simplifies the fraction correctly.
- Uses the probability formula P = favorable/total to calculate the numeric value.
- Provides the final probability as the simplified fraction 1/2 (or equivalent 0.5 or 50%).

Block 5 *organizes the solution step by step*
- Presents the solution in a logical, step-by-step order: state sample space, apply exclusion criteria, determine favorable outcomes, compute probability.
- Organizes the solution into at least four clearly numbered sections with descriptive headings that follow the logical order: (1) Sample Space, (2) Excluded Numbers, (3) Favorable Outcomes, (4) Probability Calculation.

Singletons
- Lists exactly the outcomes that satisfy the condition, namely {2, 4, 6}, and does not include any excluded numbers.
- Counts three favorable outcomes and six total outcomes without error.
- Explanation is concise, free of irrelevant material, and easy to follow.
- Displays the probability calculation inside LaTeX math delimiters (e.g., `$$P = \frac{3}{6} = \frac{1}{2}$$`), showing both the unsimplified and simplified fractions within the same LaTeX block.

Figure 12: **Text-similarity blocks for a Science rubric.** Criteria are grouped offline by embedding similarity (`all-mpnet-base-v2`, cosine ≥ 0.62, connected components); the italic label is our gloss of the requirement the block shares. Block dropout drops or keeps each block as a unit.

Comparison protocol. Reward hacking grows with training, so end-of-run comparisons confound the effect with training length. All cross-run comparisons use window means over steps 400–600 and matched-checkpoint win counts. The win counts record, at each eval checkpoint in the window shared by both runs, whether the run beats base on the identical prompt set. The window sits after the hacking onset (step $\sim$240 on Medical, Fig. 2) and inside every run's logged range except that of the $f{\approx}60\%$ run, whose window statistics cover steps 400–540. Win-count denominators differ where a run's logged evaluations end mid-window.

POW3R implementation. We implement POW3R (Tyagi et al., 2026) with the method's published defaults ($\lambda = 0.5$, factors clipped to $[0.67, 1.5]$). Its dynamic per-criterion factors are derived from the within-group verdict variance of the current rollout group. The factors are applied once all rollouts in a group have been generated and judged, which is the same point at which GRPO forms advantages. We deviate from the original in two ways. RubricHub rubrics carry no per-criterion category labels, so the original's within-category balancing (their Eq. 3) is inactive here. As a result, the factors reweight criteria globally within each prompt's rubric. We also recompute factors within each step from the current group's verdicts and apply them immediately, whereas the original smooths them across prompt visits with an EMA and a one-visit lag (their Eq. 7). Our implementation is therefore the unsmoothed variant. Otherwise POW3R uses the same training recipe as every other run in this paper, and evaluation always uses the full static rubric.

Example B: Medical (RubricHub-Medical). Prompt: "What is the typical age range for children who commonly experience nursemaid's elbow or annular ligament displacement?" (16 criteria: five blocks)

Block 1 *names the condition and its core age range (1–4 years, peak 2–3)*
- The response explicitly states that nursemaid's elbow most commonly occurs in children between 1 and 4 years of age.
- The response specifies that the peak incidence of nursemaid's elbow is between 2 and 3 years of age.
- The response correctly identifies nursemaid's elbow as radial head subluxation or annular ligament displacement (or uses an equivalent term).
- Each of the two synonymous terms, "radial head subluxation" and "annular ligament displacement", appears at least twice in the response.

Block 2 *the broader at-risk age span (about 6 months to 5–6 years)*
- The response mentions a broader at-risk age span, e.g., from about 6 months up to approximately 5–6 years.
- The response states the peak incidence as the exact numeric interval "2 to 3 years" (inclusive) in a separate sentence.
- The response provides the broader at-risk span with both lower and upper numeric bounds (e.g., "6 months" and "5–6 years") in the same sentence, using exact numbers and units.

Block 3 *the upper age limit (rare after 5, very rare after 7)*
- The response notes that the condition becomes significantly less common after age 5 and is very rare after age 7.
- The response explains that incidence sharply declines after around age 5 due to ligament and bone development.
- The response contains a distinct section titled "Upper Age Limit" that explicitly says the condition is "significantly less common after age 5" and "very rare after age 7".

Block 4 *keeps the answer brief and focused on the age range*
- The response keeps the answer centered on the age range and does not include unrelated treatment or diagnosis details.
- The response provides the age range in a brief, direct statement (e.g., first sentence) without unnecessary verbosity.

Block 5 *explains the anatomical mechanism (ligament laxity, ossification)*
- The response adds a short explanation linking the risk to ligament laxity and skeletal development.
- The response explains that the injury becomes unlikely in older children because the radial head ossifies/widens and the annular ligament strengthens.
- The response includes a numbered list of three or more distinct anatomical factors (e.g., radial head shape, ligament laxity, ossification) that directly link skeletal maturation to the age-related risk.
- The exact phrase "ligament laxity" is present, describing why younger children are predisposed to the injury.

Figure 13: **Text-similarity blocks for a Medical rubric.** Same procedure as Fig. 12. All five blocks have two or more members, so every mask drops several criteria at once.

Dropout variants not explored. Per-criterion fractions f_k (with $f_k = 0$ on criteria that must never be dropped) are one natural extension. Two others are schedules that anneal f from high to zero over training, and blocks estimated online from the group's own verdicts instead of from criterion text. Viewed as an objective, dropout optimizes $\mathcal{J}_{\text{drop}}(\theta) = \mathbb{E}_m[\mathcal{J}(\theta; m)]$, a marginal over sub-rubrics, which formalizes "never optimize the same rubric twice."

Full per-criterion grid. Table 5 reports the complete step-600 per-criterion breakdown behind Fig. 14 and adds the exact proxy pass rates and the underclaim row for both model sizes.

		Medical			Science		
	Matched step 600	base	$f \approx 30\%$	$f \approx 50\%$	base	$f \approx 30\%$	$f \approx 50\%$
8B	proxy pass rate	87.7	88.8	87.1	89.5	88.3	88.3
	gold pass rate (↑)	48.7	51.5	**52.3**	46.0	51.5	**53.4**
	overclaim (↓)	41.5	39.7	**37.9**	43.4	36.4	**34.7**
	underclaim (↓)	2.5	**2.4**	3.1	0.1	0.1	0.1
4B	proxy pass rate	87.7	88.0	88.5	92.2	91.8	91.5
	gold pass rate (↑)	43.4	43.7	**47.3**	38.1	**43.3**	38.2
	overclaim (↓)	46.9	46.8	**43.7**	54.1	**48.6**	53.4
	underclaim (↓)	2.6	**2.4**	2.5	**0.0**	0.1	0.1

Table 5: **Per-criterion breakdown at step 600 for Medical and Science, 8B and 4B** (all values in %). The table gives all values underlying Fig. 14. Both judges grade every positive-weight criterion of each run's step-600 responses on matched prompts (temp. 0). Medical 992 prompts, 7,608 criteria; Science 362–363 prompts. 4B: Medical 992 prompts, 7,667 criteria; Science 686 prompts, 5,099 criteria. Overclaim: proxy accepts, gold rejects. Underclaim: proxy rejects, gold accepts. Bold: best run per row, pair, and block.

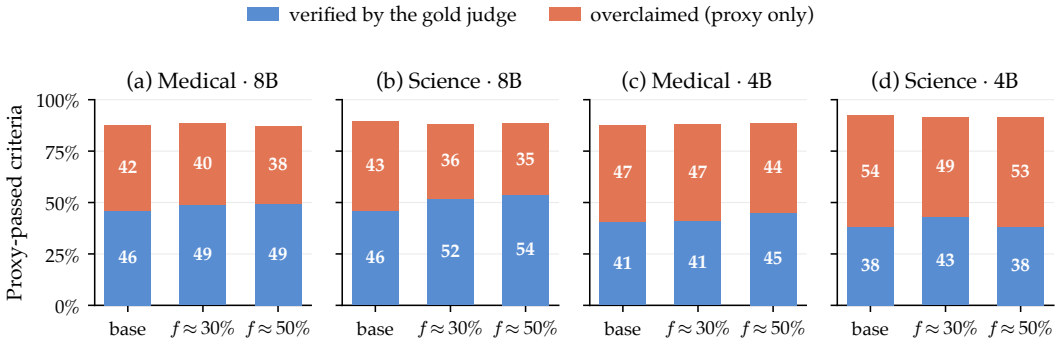


Figure 14: **Same proxy pass rate, different composition.** Both judges grade each run's step-600 responses on matched prompts against every positive-weight criterion. Bar height is the share of criteria the proxy judge accepts, split into the part the gold judge confirms (blue) and the overclaimed part it rejects (terracotta). Table 5 gives the full grid, with exact values, counts, and underclaim.