EAT YOUR PRECISION: PRECISION OBJECTIVE MITIGATES REWARD HACKING IN RUBRIC-BASED RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Rubric-based rewards enable effective reinforcement learning on non-verifiable tasks. However, because these rewards are typically defined as a weighted measure of rubric coverage, models are trained primarily to maximize a *recall* objective. As a result, RL-trained models produce responses that are more complete but also more verbose and contain more unnecessary or factually incorrect content, leading them to be less preferred than responses from the base model. To address this issue, we present the FORK reward, which jointly optimizes recall and a novel precision objective, which requires each unit of a model response to be both necessary and factually correct, as measured by rubric-driven necessity credits and a factual-correctness grader, respectively. Our experimental results show that FORK outperforms existing rubric-based RL methods and achieves Pareto-optimal performance that balances rubric coverage and overall response quality. Compared to the original recall-based reward, FORK substantially improves precision and reduces response length while achieving similar rubric coverage, which leads to significantly higher win rate compared to the base policy (e.g., from 10.6% to 68.0% on RubricHub-Medical and from 22.1% to 74.1% on HealthBench).[1]

1 INTRODUCTION

Reinforcement learning (RL) with verifiable rewards has led to rapid advances in AI for math and coding, where the correctness of the answers can be reliably measured and used as reward signals. In comparison, for non-verifiable tasks such as open-ended generation, the evaluation criteria for model responses can be diverse and sometimes subjective, which makes it difficult to define rewards in RL. To resolve this issue, Gunjal et al. (2026) and Viswanathan et al. (2026) leverage rubric-based RL, which first generates a set of rubrics that model responses should satisfy and then uses the weighted coverage of these pre-defined rubrics as rewards.

However, such rubric-coverage-based reward is easily hacked. As shown in Figure 1, when a model is only optimized towards maximizing the coverage of pre-defined rubrics, we observe that although the model responses become more complete, their factual correctness and necessity decrease, and the responses become more verbose. This makes them less preferred than base policy responses (Mahmoud et al., 2026). This reward hacking behavior is not surprising since rubrics as rewards currently behave like a *recall* objective. The missing part is a *precision* objective: whether the content of a model response is factually correct and necessary.

With this motivation, we propose FORK, which converts rubric-based rewards from a recall score into an F_β score that aggregates recall and precision via their weighted harmonic mean. Given a set of rubrics, to measure the precision, we first divide a model response into a sequence of units and then determine the necessity and factual correctness of each unit by prompting an LLM grader. To improve the accuracy of necessity judgments, which we observe as a bottleneck[2], we develop a credit-based method in which an LLM grader matches rubrics to units, assigns each unit a necessity credit, and determines unit necessity based on a dynamic threshold value.

[1] We will make the source code publicly available.

[2] Necessity judgments of all candidate training graders show only moderate correlation with stronger reference graders, whereas factual correctness judgments show stronger correlation (Table 7).

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.

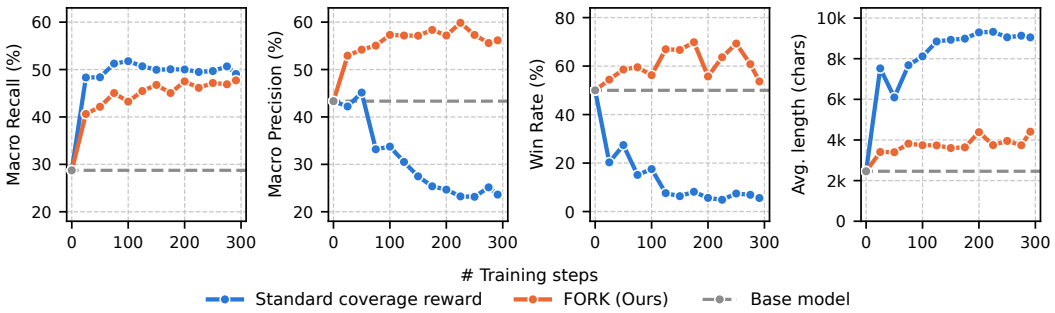


Figure 1: **Reward hacking in rubric-based RL and our mitigation method.** With the standard coverage reward, recall (i.e., weighted coverage of rubrics) quickly saturates, while precision and win rate against a base model continually decreases and responses become more verbose. With a precision objective included in the reward (FORK), the model reaches similar recall while gradually increasing precision and win rate and generating less verbose responses. Here, we train Qwen2.5-7B-Instruct on RubricHub-Medical and evaluate it on an in-domain test set.

We demonstrate the effectiveness of FORK on both in-domain datasets (the test splits of RubricHub-Medical and RubricHub-Science, Li et al. (2026)) and external benchmarks (HealthBench, Arora et al. (2025) and ResearchQA, Yifei et al. (2026)). Compared to the standard coverage reward, FORK significantly improves the factual correctness and necessity of responses (e.g., reducing the response length by a factor of 2.6) while maintaining similar rubric coverage. These improvements also make responses from rubric-based RL-trained models more preferred than base policy responses.

Further analysis supports the effectiveness of FORK. While rubric-coverage rewards (Gunjal et al., 2026; Rezaei et al., 2025; Viswanathan et al., 2026) have a negative effect on satisfaction of absence-based rubrics, our method improves satisfaction of both presence-based and absence-based rubrics, enabling models to add necessary content and suppress irrelevant content at the same time. We also verify that a model trained with our method remains preferred to the base model under different judge prompts and remains more effective than other rubric-based RL methods when evaluated with different LLM graders.

Our key contributions are:

1. We propose FORK, which mitigates reward hacking in rubric-based RL training by incorporating a precision objective into a standard coverage objective.
2. We develop a reliable method for using an LLM grader to measure the precision of free-form responses given a set of rubrics to be satisfied.
3. Experiments show that on multiple public benchmarks, compared to the standard coverage reward, FORK significantly improves precision while maintaining similar rubric coverage, which leads to substantial improvements in the preference score (i.e., win rate compared to the base policy). FORK also outperforms other rubric-based RL methods, achieving Pareto-optimal performance that balances rubric coverage and overall response quality.

2 RELATED WORKS

Rubric-Based Reinforcement Learning. Rubric-based methods decompose response quality into explicit, prompt-specific criteria, providing structured supervision for tasks without readily verifiable outcomes (Gunjal et al., 2026; Viswanathan et al., 2026). Rubrics as Rewards uses expert-guided, instance-specific rubrics as on-policy RL rewards, exploring both explicit aggregation of criterion scores and implicit aggregation by an LLM grader (Gunjal et al., 2026). Reinforcement Learning from Checklist Feedback (RLCF) evaluates instruction-specific requirements using LLM graders and specialized verification programs, and converts aggregated checklist scores into preference pairs for direct preference optimization (Viswanathan et al., 2026). RubricHub scales rubric supervision through coarse-to-fine generation that combines principle-guided synthesis, multi-model aggregation, and difficulty evolution, supporting rubric-based RL (Li et al., 2026). These methods establish rubrics as a practical interface for specifying training objectives. We build on this paradigm while distinguishing coverage of specified requirements from the quality of all generated content.

Rubric Specification and Reward Hacking. Rubric-based optimization remains vulnerable to incomplete reward specifications. Mahmoud et al. (2026) distinguish verifier failure from rubric-design limitations: even with strong verification, rubric-based judges can prefer RL-trained responses that rubric-free judges rate worse overall, with improved completeness but degraded factual correctness and necessity. Several approaches address insufficiently specified requirements. RLCF adds universal criteria against excessive or off-topic content after observing verbose preambles under checklist optimization (Viswanathan et al., 2026). Zhang et al. (2026) refine rubrics by contrasting high-quality, diverse responses, improving discrimination in the high-reward region and delaying over-optimization. OnlineRubrics augments existing criteria through comparisons between current-policy and control-policy responses (Rezaei et al., 2025). DR Tulu maintains search-grounded, evolving rubrics that capture both newly discovered useful content and undesirable behaviors observed in rollouts (Shao et al., 2025). AMARIS further uses persistent training diagnostics to revise criteria and weights in response to recurring failures (Wu et al., 2026). These specification-refinement strategies motivate a complementary question: how should generated content be evaluated when coverage requirements remain incomplete? Our method supplements rubric coverage with unit-level assessments of necessity and factual correctness, rather than relying on criterion expansion alone, and does not require online rubric updates.

Fine-Grained Evaluation and Precision–Recall Rewards. Content-level evaluation provides a complementary perspective on response quality. Min et al. (2023) measure factual precision as the proportion of generated atomic facts supported by a reliable knowledge source. Song et al. (2024) combine sentence-level faithfulness assessment with keyfact–sentence alignment to measure completeness and conciseness, distinguishing coverage of target information from the fraction of generated sentences contributing that information. Related work directly incorporates precision–recall signals into policy optimization. Jhaveri et al. (2025) train clinical note generators with a claim-level F_1 reward combining entailment-based precision against the source dialogue and recall over dialogue-derived claims. For video captioning, Gao et al. (2026) verify atomic claims against video evidence and balance factual precision with reference-claim coverage, assigning attenuated credit to supported but unmatched details and bounding their contribution to recall. Our work adapts this content-level perspective to rubric-based RL: necessity is derived from a unit's contribution to prompt-specific coverage requirements, and precision counts units that satisfy both necessity and factual correctness. Combining this precision with rubric-coverage recall through an F_β reward couples what a response should cover with whether its generated content is warranted and correct.

3 FORK: PRECISION-AWARE RUBRIC REWARD

In this section, we propose FORK, which can mitigate reward hacking in rubric-based RL by incorporating a precision objective (Figure 2). We first provide background on the standard rubric-coverage reward formulation (Section 3.1), describe how to measure the precision of model responses (Section 3.2), and introduce FORK as a new reward formulation for rubric-based RL (Section 3.3).

3.1 PRELIMINARY: RUBRIC-COVERAGE REWARD

Given a prompt x, the current policy π_{θ_t} generates a free-form response $y \sim \pi_{\theta_t}(\cdot \mid x)$. Let

$$\mathcal{R}(x) = \{(r_j, w_j)\}_{j=1}^{M}$$

denote the coverage rubrics associated with x, where r_j specifies a desired element of the response and $w_j > 0$ denotes its importance. For each rubric r_j, an LLM grader produces a binary coverage label

$$s_j(x, y) = \mathbb{I}[y \text{ satisfies } r_j] \in \{0, 1\}.$$

The conventional rubric-coverage reward is the importance-weighted fraction of satisfied rubrics:

$$\mathrm{Rec}(x, y) = \frac{\sum_{j=1}^{M} w_j s_j(x, y)}{\sum_{j=1}^{M} w_j}. \tag{1}$$

We refer to this quantity as *rubric recall* (or simply recall), since it measures how much of the desired content is covered by the response. Rubric recall, however, evaluates mainly whether the requested

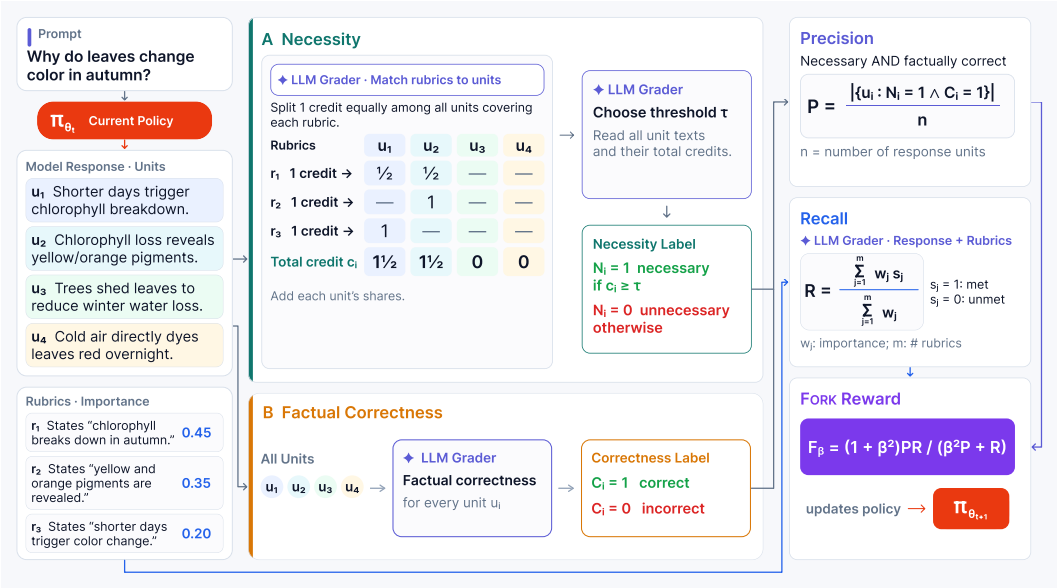


Figure 2: **Training with FORK.** Given a prompt, a policy model samples a response, which is then divided into units. An LLM grader matches rubrics to units to assign each unit a necessity credit and determines a credit threshold for unit necessity. In parallel, the grader assesses the factual correctness of all units. Precision is defined as the fraction of necessary and factually correct units, and recall follows the standard definition of weighted rubric coverage. We update the policy model using their weighted harmonic mean as a reward.

content is present[3]. It does not penalize content that is redundant, unnecessary, or factually incorrect unless specific failure cases are covered by rubrics. Consequently, a response y can obtain a high coverage reward while containing substantial low-quality content. We therefore complement rubric recall with a unit-level notion of precision.

3.2 MEASURING PRECISION OF FREE-FORM RESPONSES

Response Unitization. We first transform the response into an ordered sequence of units:

$$\mathcal{U}(y) = (u_1, \ldots, u_n).$$

For simplicity, we define a unit as approximately a single sentence in the original response[4]. We preserve the original wording and substantive content, making only minimal substitutions needed to make each unit self-contained. For example, a pronoun may be replaced with its antecedent while retaining the remainder of the original sentence. Purely functional or organizational material—such as greetings, section headers, discourse markers (e.g., "first" or "in summary"), and transition-only sentences or phrases—is discarded. Unitization therefore isolates the substantive content of the response without summarizing or rewriting it. We automate this process using an LLM unitizer[5].

Credit-based Necessity. We determine whether each unit contributes necessary content toward the coverage rubrics. For every rubric–unit pair (r_j, u_i), an LLM grader predicts

$$m_{ji} = \mathbb{I}[u_i \text{ contributes to addressing } r_j] \in \{0, 1\}.$$

Let $K_j = \sum_{i=1}^{n} m_{ji}$ be the number of units matched to rubric r_j. We assign each rubric a total unit-credit budget of one and divide this credit equally among its matched units. The credit assigned by rubric r_j to unit u_i is

$$a_{ji} = \begin{cases} \dfrac{m_{ji}}{K_j}, & K_j > 0, \\ 0, & K_j = 0. \end{cases} \tag{2}$$

[3]RubricHub (Li et al., 2026), a widely adopted dataset for rubric-based RL training, consists of 90.2% presence-based rubrics and 8.6% absence-based rubrics (Mahmoud et al., 2026).

[4]We also divide the response into units by line breaks ('\n') and report the results in Section 4.3.

[5]Because response unitization is a trivial task, we use the same model for response unitization and grading for simplicity.

The accumulated necessity credit of u_i is then $c_i = \sum_{j=1}^{M} a_{ji}$. This allocation discounts redundant units: if K_j units address the same rubric, each receives only $1/K_j$ credit from that rubric. At the same time, a unit may accumulate credit from multiple rubrics when it contributes to several desired elements. Given the accumulated necessity credits for all units, an LLM grader determines a threshold $\tau > 0$, which defines the binary necessity label

$$N_i = \mathbb{I}[c_i \geq \tau]. \tag{3}$$

Thus, $N_i = 1$ indicates that u_i contributes sufficient rubric-grounded content, whereas $N_i = 0$ indicates that its contribution is insufficient or redundant.

Factual Correctness. Independently of necessity, an LLM grader assigns each unit a binary factual correctness label

$$C_i = \mathbb{I}[u_i \text{ is factually correct given } x] \in \{0, 1\}. \tag{4}$$

We use a strict unit-level criterion: $C_i = 1$ only when the factual claims expressed by u_i are correct; a unit containing a factual error receives $C_i = 0$.

Response Precision. A unit contributes positively to precision only if it is both necessary and factually correct. We therefore define the precision of the response as

$$\mathrm{Prec}(x, y) = \frac{1}{n} \sum_{i=1}^{n} N_i C_i. \tag{5}$$

3.3 FORK REWARD

We combine response precision and rubric recall using their weighted harmonic mean (i.e., F_β score):

$$r_{\mathrm{FORK}}(x, y) = \begin{cases} \dfrac{(1+\beta^2)\ \mathrm{Prec}(x,y)\,\mathrm{Rec}(x,y)}{\beta^2\,\mathrm{Prec}(x,y) + \mathrm{Rec}(x,y)}, & \mathrm{Prec}(x,y) + \mathrm{Rec}(x,y) > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

The harmonic mean favors responses that jointly cover the desired rubrics and express that content using necessary, factually correct units. In particular, increasing coverage cannot compensate for arbitrarily low precision, and vice versa. Here, $\beta > 1$ gives more weight to recall and $\beta < 1$ gives more weight to precision. The resulting score serves as the reward for updating the policy. The corresponding learning objective is

$$\max_{\theta}\ \mathbb{E}_{x\sim\mathcal{D},\, y\sim\pi_\theta(\cdot|x)}\left[r_{\mathrm{FORK}}(x, y)\right], \tag{7}$$

producing an updated policy $\pi_{\theta_{t+1}}$ from the current policy π_{θ_t}.

4 EXPERIMENTS

Our experiments address four research questions: **(RQ1)** Can FORK improve response precision and overall quality while maintaining comparable rubric coverage? **(RQ2)** Do these improvements generalize to external benchmarks? **(RQ3)** How do precision estimation and precision–recall aggregation affect performance? **(RQ4)** Are the observed improvements robust to changes in evaluation graders and judge prompts?

4.1 SETUP

Base Model and Datasets. We mainly use Qwen2.5-7B-Instruct (Yang et al., 2025) as the base policy model, following previous rubric-based RL work (Gunjal et al., 2026; Rezaei et al., 2025; Mahmoud et al., 2026). We separately train the model on two subsets of RubricHub (Li et al., 2026), Medical and Science. Each example consists of a prompt (or query) x sourced from existing open-ended QA datasets and a set of evaluation rubrics and their weights $\mathcal{R}(x)$ synthesized through multiple steps using LLMs. In our experiments, for the Science subset, we exclude examples sourced from ResearchQA (Yifei et al., 2026) to use this benchmark as a separate evaluation suite. As a result, the Medical subset contains 12,519 training examples and 1,391 test examples, and the Science subset contains 19,806 training examples and 2,201 test examples.

Baselines and Ablations. Based on their reward formulations, we categorize other methods designed to mitigate reward hacking in rubric-based RL into the following three types:

(1) Rubric-Coverage Rewards include *OnlineRubrics* (Rezaei et al., 2025), which adds complementary rubrics to predefined rubrics at every training step by comparing current-policy and control-policy responses; *Universal Rubrics*[6], which adds two generic criteria (necessity and factual correctness) to the predefined rubrics; and *Reweighted Absence-Based Rubrics*, which increases the weights w_j of absence-based rubrics. We also include the *Standard* rubric-coverage reward (Gunjal et al., 2026) for reference.

(2) Preference-Based Rewards include *Holistic Preference*[7], which assigns rewards according to an LLM grader's overall preference based on rubric coverage, necessity, and factual correctness when comparing current-policy and control-policy responses. If a current-policy response is preferred, the reward is 1; if it is less preferred, the reward is 0; and if the responses are tied, the reward is 0.5.

(3) Ablations: FORK Rewards with Precision Variants include *Generic Criteria*, which measures precision as the fraction of necessary and factually correct units, both determined by prompting an LLM grader; *Offline Criteria*, which elicits prompt-specific precision criteria by comparing different model responses before training; and *Online Criteria*, which elicits precision criteria specific to current-policy responses at each training step.

For simplicity, we use base policy responses as control-policy responses. Implementation details for the baselines and ablations we introduce are provided in Appendix A, and their corresponding training prompts are provided in Appendix E.1.

Training Grader Selection. We select an LLM to use as the *training grader* for all rubric-based RL experiments. To do that, we first select five candidate training graders and measure agreements of their judgments with three frontier models. The result shows that Qwen3.6-35B-A3B (Qwen Team, 2026b) shows the strongest correlation (r = 0.714), so we use this model as a universal training grader. Please refer to Appendix B for the detailed grader selection process.

Evaluation. We evaluate in-domain performance on RubricHub test sets and cross-dataset generalization on external benchmarks (i.e., HealthBench (Arora et al., 2025) for the medical domain and ResearchQA (Yifei et al., 2026) for the science domain). We conduct two types of evaluations: (1) rubric-based evaluations measuring macro rubric recall, macro precision, and macro F1, and (2) rubric-free evaluations reporting the overall win rate against the base policy model and average response length (i.e., verbosity). We use GPT-5.4 (Singh et al., 2025), a strong frontier LLM, as a *reference grader* for automating these evaluations. We use the same rubric-free judge prompt as in Mahmoud et al. (2026) and provide newly introduced judge prompts in Appendix E.2. To measure precision, a reference grader directly determines unit necessity and factual correctness given a set of coverage rubrics. We compute the F1 score as the harmonic mean of rubric recall and precision.

Hyperparameters and Implementation Details. We update the policy model using the group relative policy optimization (GRPO) algorithm (Shao et al., 2024) with 16 rollouts per prompt, a context length of 2,000 tokens, and a sampling temperature of 1.0. Models are trained with a batch size of 128, a learning rate of 4.2×10^{-6}, and a constant schedule with 5% linear warmup for 3 epochs on 8 NVIDIA H100 GPUs.

4.2 RESULTS

We report main results on medical domain in Table 1 and science domain in Table 2. We enumerate our main findings in the following:

FORK achieves higher precision than other rubric-based RL methods while maintaining recall at the level of standard coverage reward. Because of this, compared to rubric-coverage rewards, our method improves rubric-based F_1 score by 9.8 pp (from 28.8% to 38.6%) on RubricHub-

[6]We basically follow Viswanathan et al. (2026) but modify generic rubric items to directly address the observed reward hacking.

[7]We adopt the preference-reward formulation from Munos et al. (2024) but use different judging criteria.

Table 1: **Main results on medical domain.** We conduct rubric-based and rubric-free evaluations of our method and other rubric-based RL baselines on the RubricHub-Medical test set and HealthBench. Rubric-based evaluation measures macro rubric recall, macro precision, and the macro F1 score. Precision indicates the fraction of necessary and factually correct response units. Rubric-free evaluation measures win rate against the base policy model and average response length. We use GPT-5.4 as a reference grader for automating these evaluations.

	RubricHub-Medical test set					**HealthBench**	
	Rubric-based			Rubric-free		Rubric-based	Rubric-free
	Recall	Precision	F_1	Win Rate	Length	Recall	Win Rate
Base							
Qwen2.5-7B-Instruct	22.6	46.6	26.3	N/A	2,432	13.4	N/A
Rubric-Coverage Rewards							
Standard	35.7	25.5	26.4	3.1	9,195	25.9	7.1
OnlineRubrics	**38.7**	26.8	28.8	10.6	8,736	25.7	22.1
Universal Rubrics	37.6	27.6	28.4	4.3	8,457	**26.1**	18.5
Reweighted Absence-Based Rubrics	36.4	24.2	25.7	8.5	9,106	25.7	6.3
Preference-Based Rewards							
Holistic Preference	27.5	47.5	31.3	65.4	3,584	22.8	71.6
FORK Rewards w/ Prec. Variants ($\beta = 1$) **(Our Ablations)**							
Generic Criteria	33.8	47.7	36.9	44.0	4,312	24.5	63.4
Offline Criteria	38.0	35.1	33.7	20.7	6,748	25.1	32.6
Online Criteria	36.8	39.3	34.7	14.3	6,699	25.9	14.0
FORK Rewards (Ours)							
$\beta = 1$	32.5	**53.2**	37.5	**68.0**	3,533	25.2	**74.1**
$\beta = 1.25$	35.8	49.0	**38.6**	55.4	4,392	25.1	64.3

Table 2: **Main results on science domain.** We evaluate rubric-based RL methods on the RubricHub-Science test set and ResearchQA as in Table 1.

	RubricHub-Science test set					**ResearchQA**	
	Rubric-based			Rubric-free		Rubric-based	Rubric-free
	Recall	Precision	F_1	Win Rate	Length	Recall	Win Rate
Base							
Qwen2.5-7B-Instruct	27.3	41.8	29.6	N/A	2,673	48.3	N/A
Rubric-Coverage Rewards							
Standard	31.4	33.5	29.2	38.6	4,453	**64.4**	16.1
Universal Rubrics	**31.8**	37.9	31.0	42.5	4,128	62.1	26.6
FORK Rewards (Ours)							
$\beta = 1.25$	30.4	**45.1**	**33.4**	**61.5**	3,038	58.9	**67.1**

Medical. Preference-based rewards show higher precision than rubric-coverage rewards but have significantly degraded recall.

FORK achieves higher rubric-free win rate than other rubric-based RL methods. Compared to rubric-coverage rewards, our method increases overall win rate by 57.4 pp (from 10.6% to 68.0%) on RubricHub-Medical and by 19.0 pp (from 42.5% to 61.5%) on RubricHub-Science. Further analysis reveals that our method shows better response quality in all five dimensions, including completeness, factual correctness, conciseness, relevance, and safety (Table 8).

FORK effectively reduces response length while retaining necessary content. On RubricHub-Medical, compared to the standard coverage reward, our method ($\beta = 1.25$) shortens the response length by a factor of 2.1 but the recall maintains nearly the same, which proves that verbosity issue is significantly reduced. We also qualitatively verify this by manually checking representative examples (Appendix D).

Table 3: **Presence-based and absence-based rubric satisfaction results.** We categorize rubrics in the RubricHub-Medical test set as presence-based or absence-based and report satisfaction rates for rubric-based RL methods.

Rubric Satisfaction Rate	Presence-based	Absence-based
Base		
Qwen2.5-7B-Instruct	20.3	40.4
Rubric-Coverage Rewards		
Standard	35.3	37.2
OnlineRubrics	38.5	36.9
Universal Rubrics	37.3	34.0
FORK Rewards (Ours)		
$\beta = 1.25$	34.9	42.9

Table 4: **Rubric-free evaluation results using a different judge prompt.** We adopt a different judge prompt from Rezaei et al. (2025) and report win rates against the base policy model. **Bold** and underline indicates the best and second best performance.

	RubricHub-M	HealthBench
	Win Rate	Win Rate
Preference-Based Rewards		
Holistic Preference	80.8	**89.4**
FORK Rewards w/ Prec. Variants ($\beta = 1$)		
Generic Criteria	84.6	87.3
Offline Criteria	<u>88.6</u>	80.3
Online Criteria	65.2	53.4
FORK Rewards (Ours)		
$\beta = 1$	84.8	<u>88.6</u>
$\beta = 1.25$	**89.3**	84.6

FORK generalizes effectively to external benchmarks. While maintaining similar level of recall, compared to rubric-coverage rewards, our method increases overall win rate by 52.0 pp (from 22.1% to 74.1%) on HealthBench and by 40.5 pp (from 26.6% to 67.1%) on ResearchQA.

4.3 ANALYSIS

Improvement in satisfaction of both presence- and absence-based rubrics. In RubricHub, most of the coverage reward weight falls on presence-based rubrics rather than absence-based rubrics (Mahmoud et al., 2026). Accordingly, we observe that the standard coverage reward significantly increases the presence-based rubric satisfaction rate (+15.0 pp) while slightly decreasing the absence-based rubric satisfaction rate (-3.2 pp) (Table 3). This implies that the model learns to add both desirable and undesirable content. Rubric-coverage reward baselines produce more desirable content but fail to suppress undesirable content. In contrast, our method nearly matches the presence-based rubric satisfaction rate of the standard coverage reward while improving the absence-based rubric satisfaction rate (+5.7 pp).

Robustness to rubric-free evaluation judge prompts. Since rubric-free win rates depend heavily on LLM judge prompt instructions, we verify whether our method is robust to prompt variations. To do so, we adopt another judge prompt from Rezaei et al. (2025) and verify on the medical domain (Table 4). Since rubric-coverage reward baselines have significantly degraded win rates, we do not include those results. Our method mostly outperforms the other baselines on both evaluation suites, supporting its robustness to different judge prompts.

Isolating the effectiveness of our reward formulation. We use a reference grader for our main evaluation (Section 4.2). However, this evaluation combines the effect of our reward formulation with the effect of transferring to a stronger, unseen grader. To isolate the former, we use the training grader for evaluation (Figure 3). The results show that FORK rewards have the highest win rates among rubric-based RL methods, while β = 1.25 shows Pareto-optimal performance that balances rubric coverage and overall win rate. This supports the effectiveness of our reward formulation compared to the baselines.

Different precision–recall aggregations. To justify using a harmonic mean to aggregate precision and recall, we compare F_1 score aggregation with simple averaging (Table 6). The results show that simple averaging yields slightly higher precision (+2.1 pp) but lower recall (-6.3 pp) than F_1 score aggregation. We observe that higher precision compensates for lower rubric recall during training. We also investigate whether further increasing the weight of recall in F_β score aggregation is beneficial. To do so, we train a model with F_2 score aggregation but observe a detrimental effect on both precision and recall. We leave finding the optimal value of β for future work.

Our unit necessity prediction outperforms prompting-based method. To measure the accuracy of unit necessity prediction, we progressively remove units predicted as unnecessary from a response and measure how much rubric recall score is decreased (Figure 4). Specifically, we randomly sample 40 base model responses from RubricHub-Medical containing 723 units and progressively remove units starting from unnecessary ones and then necessary ones once all unnecessary units are removed. We compare our credit-based method with prompting-based method used in *Generic*

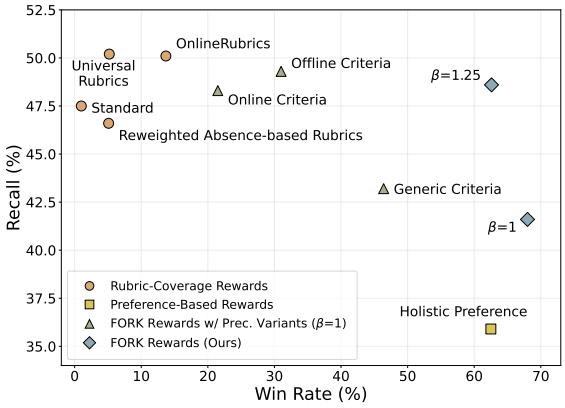


Figure 3: **Recall vs. Win Rate using a training grader.** We report recall against win rate for rubric-based RL methods evaluated with the training grader on RubricHub-Medical.

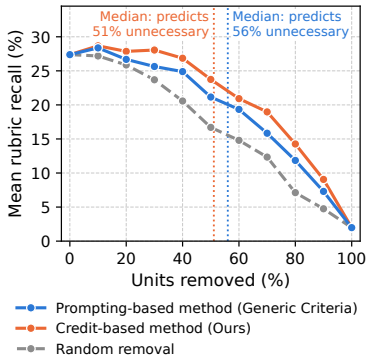


Figure 4: **Accuracy of necessity prediction.** We progressively remove response units starting from unnecessary ones. We use 723 units from 40 base model responses in RubricHub-Medical.

Table 5: **Effectiveness of two different definitions of units.** We train models with *Generic Criteria* baseline on RubricHub-Medical using two definitions of units and compare these with a model trained with standard rubric-coverage reward.

	Recall	Win Rate	Length
Units as sentences (Ours)	33.8	44.0	4,312
Units divided by line changes	35.7	37.8	5,775
Standard coverage reward	35.7	3.1	9,195

Table 6: **Effectiveness of different aggregations of precision and recall.** We compare our F_β aggregation with simple averaging on RubricHub-Medical while using the same method to measure precision.

	Recall	Precision	F_1
Average	26.2	**55.3**	32.8
F_1 score	32.5	53.2	37.5
$F_{1.25}$ score	**35.8**	49.0	**38.6**
F_2 score	35.5	43.6	36.7

Criteria baseline and random removal for reference. Our method shows nearly no decrease in recall until 40% of units are removed (note that in 51% of units are predicted as unnecessary). In contrast, prompting-based method shows gradual decrease in recall as unnecessary units are removed. This supports effectiveness of our unit necessity prediction and explains higher precision and overall winning rate of FORK reward ($\beta = 1$) than *Generic Criteria* baseline (Table 1).

Alternative definition of response units. We also divide model responses into units by line changes and compare it with our sentence-level units (Table 5). For simplicity, we train models with *Generic Criteria* baseline using different definition of units. The result shows that including precision objective using either definition of units both leads to significantly higher winning rate against the base policy model and similar rubric recall score, which indicates that both definitions of units are valid. However, since we observe semantically inconsistent divide in line changes, we adopt units as sentences for consistency.

5 CONCLUSION

In this work, we introduce FORK, a simple method that mitigates reward hacking in rubric-based RL, by adding a complementary precision objective to the original recall-based reward. To address the observed reward hacking behavior, we define a precision objective to require each unit of a model response to be necessary and factually correct. We propose rubric-driven necessity credits to faithfully measure necessity and use an LLM grader to measure factual correctness. Our experimental results show that FORK outperforms other rubric-based RL methods and achieves Pareto-optimal performance that balances recall and overall response quality.

AI USE STATEMENT

In this work, we used generative AI tools for searching related previous works and checking grammar. We have reviewed all AI-assisted work. Specifically, we manually looked at the previous works that the AI tool suggests and verified their relevance. Also, we reviewed grammar-corrected contents so that there is no distortion in the original semantic meaning.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Anthropic. Claude Opus 5 System Card. Technical report, Anthropic, July 2026. URL https://www.anthropic.com/claude-opus-5-system-card.

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

Mingqi Gao, Hongyuan Dong, Yifei Chen, Zhisheng Zhong, Zheng Ruan, Wenjin Hou, Yu Chen, Han Hu, and Yansong Tang. Claim-level rubric rewards for video caption reinforcement learning. *arXiv preprint arXiv:2607.05150*, 2026.

Google DeepMind. Gemini 3.8 Flash Model Card. Technical report, Google DeepMind, September 2026. URL https://deepmind.google/models/model-cards/gemini-3-8-flash/.

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*, volume 2026, pp. 127924–127945, 2026.

Samyak Jhaveri, Praphul Singh, Jangwon Kim, Tara Taghavi, and Krishnaram Kenthapadi. Optimizing long-form clinical text generation with claim-based rewards. *arXiv preprint arXiv:2510.02338*, 2025.

Sunzhu Li, Jiale Zhao, Huimin Ren, Zhenlin Wei, Yang Zhou, Jingwen Yang, Shunyu Liu, Kaike Zhang, and Chen Wei. Rubrichub: A comprehensive and highly discriminative rubric dataset via automated coarse-to-fine generation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 31320–31344, 2026.

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.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 12076–12100, 2023.

Remi Munos, Michal Valko, Daniele Calandriello, Mohammad Gheshlaghi Azar, Mark Rowland, Zhaohan Daniel Guo, Yunhao Tang, Matthieu Geist, Thomas Mesnard, Côme Fiegel, Andrea Michi, Marco Selvi, Sertan Girgin, Nikola Momchev, Olivier Bachem, Daniel J Mankowitz, Doina Precup, and Bilal Piot. Nash learning from human feedback. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 36743–36768. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/munos24a.html.

OpenAI. GPT-6 Astra System Card. Technical report, OpenAI, September 2026. URL https://deploymentsafety.openai.com/gpt-6-astra.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.6-35B-A3B: Agentic coding power, now open to all, April 2026b. URL https://qwen.ai/blog?id=qwen3.6-35b-a3b.

MohammadHossein Rezaei, Robert Vacareanu, Zihao Wang, Clinton Wang, Bing Liu, Yunzhong He, and Afra Feyza Akyürek. Online rubrics elicitation from pairwise comparisons. *arXiv preprint arXiv:2510.07284*, 2025.

Rulin Shao, Akari Asai, Shannon Zejiang Shen, Hamish Ivison, Varsha Kishore, Jingming Zhuo, Xinran Zhao, Molly Park, Samuel G Finlayson, David Sontag, et al. Dr tulu: Reinforcement learning with evolving rubrics for deep research. *arXiv preprint arXiv:2511.19399*, 2025.

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

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Hwanjun Song, Hang Su, Igor Shalyminov, Jason Cai, and Saab Mansour. Finesure: Fine-grained summarization evaluation using llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 906–922, 2024.

Vijay Viswanathan, Yanchao Sun, Xiang Kong, Meng Cao, Graham Neubig, and Sherry Wu. Checklists are better than reward models for aligning language models. *Advances in Neural Information Processing Systems*, 38:114728–114754, 2026.

Peilin Wu, Xinlu Zhang, Kun Wan, Wentian Zhao, Gang Wu, Xinya Du, and Zhiyu Chen. Amaris: A memory-augmented rubric improvement system for rubric-based reinforcement learning. *arXiv preprint arXiv:2605.18592*, 2026.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 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.

Junkai Zhang, Zihao Wang, Lin Gui, Swarnashree Mysore Sathyendra, Jaehwan Jeong, Victor Veitch, Wei Wang, Yunzhong He, Bing Liu, and Lifeng Jin. Chasing the tail: Effective rubric-based reward modeling for large language model post-training. In *International Conference on Learning Representations*, volume 2026, pp. 133430–133457, 2026.

A Detailed Description of Other Rubric-based RL Methods

In Section 4.1, we briefly describe which baselines and ablations we adopt. Since there are newly introduced methods, in this section, we further describe how those are implemented in detail.

A.1 Rubric-Coverage Rewards

Universal Rubrics adds two generic rubrics corresponds to necessity and factual correctness to a predefined set of rubrics. The verbatim rubrics are: (1) "The model response should be *necessary*: every part is needed, with nothing repeated, padded, tangential, or included only to sound thorough.", (2) "The model response should be *factually correct*: every claim is correct and supported, with nothing fabricated, misleading, or overconfident.".

Reweighted Absence-Based Rubrics first determines absence-based rubrics. If a rubric contains both presence-based component and absence-based component, we consider it as an absence-based. Then, given these absence-based rubrics and their original weights, an LLM increases their weights. The prompt is provided in Figure 10–11.

A.2 Preference-Based Rewards

Holistic Preference let an LLM grader to decide its preference of current policy responses against base policy responses and uses as a reward. The prompt is provided in Figure 12–13.

A.3 Ablations: Fork Rewards with Precision Variants

These rewards use the same structure of reward as our method with F_1 aggregation, but use different methods of measuring precision.

Generic Criteria replaces our credit-based necessity prediction to a simple prompting method. Therefore, an LLM grader is given a set of response units and corresponding rubrics and determines each unit's necessity and factual correctness, respectively. The prompt is provided in Figure 14.

Offline Criteria generates prompt-specific precision criteria and use those for measuring precision, instead of using two generic criteria, necessity and factual correctness. To generate those criteria, an LLM grader is given two responses, one from a base policy model and another from a RL-trained policy model with standard coverage reward, and then generates precision criteria by comparing those two responses. During online RL training, these precision criteria are used for every training step. The prompt is provided in Figure 15–16.

Online Criteria also generates prompt-specific precision criteria but updates those at every training step. To do that, an LLM grader is given a current policy response and a set of rubrics, and generates online precision criteria by evaluating the response quality. During online RL training, these precision criteria are updated for every training step. The prompt is provided in Figure 17–18.

B Detailed Training Grader Selection Process

We select an LLM to use as the *training grader* for all rubric-based RL experiments. This is especially important for mitigating misaligned decisions by the training grader, which have been identified as another mode of reward hacking in rubric-based RL (Mahmoud et al., 2026). We first select candidate training graders as follows: gpt-oss-20b, gpt-oss-120b (Agarwal et al., 2025), GPT-4.1 Mini (Achiam et al., 2023), Qwen3.5-122B-A10B (Qwen Team, 2026a), and Qwen3.6-35B-A3B (Qwen Team, 2026b). Then, using 40 randomly sampled base policy responses from RubricHub-Medical, we measure each training grader's ability to make five types of judgments required by our method and the baselines: rubric satisfaction, holistic preference, necessity, factual correctness, and necessity credit thresholding. To do this, we collect judgments from three frontier LLMs, GPT-6 Astra (OpenAI, 2026), Gemini 3.8 Flash (Google DeepMind, 2026), and Claude Opus 5 (Anthropic, 2026), take their majority judgments as reference judgments, and measure Pearson correlations between candidate and reference judgments.

When the correlations for the five judgment types are averaged, **Qwen3.6-35B-A3B** shows a strong correlation ($r = 0.714$) and outperforms the other candidate graders (Table 7). We select this model as the universal training grader in all experiments.

Table 7: **Training grader selection results.** We select five candidate training graders and measure their ability to make five types of judgments required for rubric-based RL methods. We use 40 randomly sampled base policy responses from RubricHub-Medical and report Pearson correlations between candidate and reference judgments.

	Rubric Sat.	**Holistic Pref.**	**Necessity**	**Factual Corr.**	**Threshold**	**Avg.**
gpt-oss-20b	0.591	0.796	0.348	0.350	0.679	0.553
gpt-oss-120b	0.606	0.825	0.563	0.677	0.688	0.672
GPT-4.1 Mini	0.639	0.846	0.447	0.579	0.711	0.644
Qwen3.5-122B-A10B	0.735	0.849	0.575	0.542	0.780	0.696
Qwen3.6-35B-A3B	0.719	0.853	0.534	0.703	0.760	**0.714**

C Further Analysis on Rubric-free Evaluations

In Section 4.2, we report that our method shows higher rubric-free win rate than other rubric-based RL methods. In this section, we analyze the LLM grader's decisions by comparing five quality dimension scores of our methods and rubric-coverage rewards (Table 8). Note that each quality dimensions are scored in a 1–7 Likert scale following Mahmoud et al. (2026).

The results show that our method has higher absolute scores than rubric-coverage rewards in all five quality dimensions, including completeness (+0.05), factual correctness (+0.76), conciseness (+2.64), relevance (+1.30), and safety (+1.67). The highest gain is at conciseness. It is interesting that safety shows the second highest gain even though our method does not learn to generate safe responses. This means that necessity and factual correctness aligns well with safety, which additionally supports effectiveness of our method. Compared to the base model, our method shows higher completeness and safety scores while showing similar factual correctness and relevance scores, which contributes to a higher overall quality score.

Table 8: **Rubric-free evaluation results across response quality dimensions.** We report each quality dimension scores of our method and rubric-coverage rewards in a 1–7 Likert scale following Mahmoud et al. (2026). We also add a base model for reference. The overall score is determined implicitly by the reference grader (i.e., GPT-5.4).

	Completeness	Factual Corr.	Conciseness	Relevance	Safety	**Overall**
Base						
Qwen2.5-7B-Instruct	4.83	**4.90**	**5.58**	**5.95**	5.46	4.94
Rubric-Coverage Rewards						
Standard	5.34	3.77	1.14	4.05	4.98	3.29
OnlineRubrics	5.68	4.05	1.79	4.49	5.39	3.95
Universal Rubrics	5.64	3.76	1.85	4.57	5.11	3.79
FORK Rewards (Ours)						
$\beta = 1$	**5.73**	4.81	4.49	5.87	**6.06**	**5.20**

D Qualitative Analysis of Model Responses

On RubricHub-Medical, FORK ($\beta = 1.25$) shortens model responses by a factor of 2.1 relative to the standard coverage reward while keeping recall nearly the same (Table 1), supporting that verbosity issue is reduced significantly. To qualitatively verify this, we first select two pairs of examples from medical and science domain, where each pair consists of model responses from the standard coverage reward and the proposed FORK reward. Then, we highlight necessary units using a reference grader (i.e., GPT-5.4).

We confirm that our method mostly generates necessary contents whereas the standard coverage reward does not. Also, while having similar amount of necessary contents, our method generates significantly shorter, more concise responses.

One thing we observe in the medical example is that the model trained with the standard coverage reward first generates an initial response, add another response, and repeat this process (Each response ends with: "If you have any more questions or need further assistance, feel free to ask. Stay Well! Qwen"). Our model does not show this behavior.

The science example shows a different failure: after computing the work correctly, the standard coverage reward adds a verification step that contradicts its own result. The reference grader labels the whole step unnecessary, so none of it is highlighted. The response then concludes that the assumption in the verification step is incorrect, and thus maintain the original answer. Our model does not show this behavior.

D.1 MEDICAL: A CONSUMER HEALTH QUESTION

Prompt

Do allergies cause coughing? I had a bad cough all winter, and it went away for a couple months when the weather got warmer. It just started again since the weather has gotten very hot (90 degrees) and rainy. Is there a cough medicine for coughs due to allergies?

FORK (ours)

Yes, allergies can indeed cause coughing, and your symptoms align with this possibility. Allergies can trigger a cough through several mechanisms, including:

1. **Postnasal Drip**: When allergens like pollen, dust mites, or mold irritate the nasal passages, they can cause increased mucus production. This excess mucus can drip down the back of the throat (postnasal drip), stimulating the cough reflex, especially at night or when lying down.
2. **Inflammation and Irritation**: Allergic reactions can cause inflammation in the airways, which can lead to a dry, hacking cough, especially if the airways are sensitive to the allergens.
3. **Bronchospasm**: In some cases, allergies can cause the airways to narrow, leading to wheezing and coughing, which is more common in individuals with asthma that is triggered by allergies.

The pattern you described—coughing worsening in the winter and returning with hot, rainy weather—suggests that your cough may be related to seasonal allergies, possibly to mold, pollen, or other outdoor allergens that are more prevalent in these conditions, or to indoor allergens that may be more active in warmer, humid weather (like dust mites, mold, or pet dander).

Cough Medicines for Allergy-Induced Cough

There are several types of cough medicines that can help manage allergy-induced cough, but it's important to note that they should be used under the guidance of a healthcare provider, especially if you have underlying health conditions or are taking other medications. Here are some options:

1. **Antihistamines**: These can help reduce the symptoms of allergies, including sneezing, itching, and runny nose, which can contribute to coughing. Common over-the-counter antihistamines include diphenhydramine (Benadryl) and cetirizine (Zyrtec). However, they can cause drowsiness, so it's best to use them before bedtime or to avoid activities that require alertness, like driving.
2. **Decongestants**: These can help relieve nasal congestion, which can contribute to postnasal drip and cough, but they do not directly treat the cough itself. Pseudoephedrine (Sudafed) is a common decongestant, but it can have side effects like increased heart rate and blood pressure, so it's not recommended for people with high blood pressure, heart disease, or prostate issues.
3. **Cough Suppressants**: These are used to suppress a cough that is dry and unproductive, and they include dextromethorphan (found in many cough syrups and cold medicines). However, if your cough is productive (producing phlegm), a cough suppressant may not be the best choice, as it can prevent you from clearing the mucus that your body is trying to expel.
4. **Expectorants**: These help loosen mucus and make it easier to cough up, and they include guaifenesin (found in Mucinex). They can be helpful if your cough is productive, but they do not address the underlying allergy.
5. **Nasal Corticosteroids**: These are often more effective for long-term allergy management and can help reduce inflammation in the nasal passages, which can contribute to coughing. Examples include fluticasone (Flonase) and mometasone (Nasonex), which are available by prescription.

Recommendations

Given the pattern of your cough, it might be beneficial to use a combination of an antihistamine and a decongestant, or an antihistamine and an expectorant, to manage your symptoms. However, since you've had a persistent cough for an extended period, it's important to consult a healthcare provider to rule out other potential causes, such as a respiratory infection, asthma, or acid reflux, which can also cause chronic cough.

Here are some steps you can take:

1. **Identify Triggers**: Keep track of when your cough occurs (e.g., during certain times of the day, after being outside, or when the weather changes) to help identify potential allergens.
2. **Environmental Control**: Use air conditioning to keep indoor humidity low (below 50%), use dehumidifiers, and regularly clean or replace air filters to reduce mold and dust mites.
3. **Allergy Testing**: Consider getting allergy testing to identify specific triggers, which can help in tailoring your treatment plan, whether it's through environmental control, allergy shots, or targeted medications.
4. **Consult a Healthcare Provider**: Speak with a healthcare provider, such as an allergist or primary care physician, to discuss your symptoms, undergo a physical examination, and possibly undergo allergy testing, and to get a prescription for appropriate medications, which might include a combination of antihistamines, decongestants, and/or nasal corticosteroids, as well as advice on managing your cough, such as using a humidifier, staying hydrated, and avoiding irritants like cigarette smoke and air pollution.

If your cough is severe, accompanied by shortness of breath, wheezing, fever, or blood in your phlegm, or if it persists for more than 8 weeks, it's crucial to seek medical attention, as these symptoms could indicate a more serious condition that requires specific treatment, such as bronchitis, asthma, or even a lung infection.

Standard coverage reward

Yes, allergies can indeed cause coughing. Allergies trigger the release of histamines and other chemicals in the body, which can lead to inflammation and increased mucus production in the airways. This can result in symptoms like coughing, sneezing, and a runny nose. The pattern you're describing—coughing that worsens in certain weather conditions—suggests that you might be experiencing seasonal allergies, also known as hay fever.

Here are a few points to consider:

1. **Seasonal Allergies and Coughing**: Allergies to pollen, mold, and other environmental triggers can worsen in spring and summer when these allergens are more prevalent. The change in weather from winter to spring and then to summer can also affect the types and levels of allergens in the air.
2. **Coughing Mechanism**: Coughing can be a response to irritation in the airways, such as from allergens, dust, or mucus. Post-nasal drip, where excess mucus from the nose drips down the back of the throat, can also cause a cough.
3. **Treatment Options**: There are several over-the-counter (OTC) medications that can help manage coughs and allergy symptoms. Here are a few options:
   - **Antihistamines**: These can help reduce sneezing, itching, and runny nose, which can in turn reduce coughing. Examples include cetirizine (Zyrtec), loratadine (Claritin), and fexofenadine (Allegra). They are generally safe for most people, but check with your pharmacist or doctor if you have any concerns, especially if you are taking other medications.
   - **Decongestants**: These can help reduce nasal congestion, which can also help with coughing. However, they should be used with caution, as they can cause rebound congestion if used for more than a few days. They are not recommended for people with high blood pressure or certain heart conditions.
   - **Expectorants**: These can help loosen mucus, making it easier to cough up. Guaifenesin (Mucinex) is an example of an expectorant. However, they are less effective for coughs caused by allergies.
   - **Cough Suppressants**: These are used to suppress a dry, hacking cough. Dextromethorphan (found in many cough syrups) can be effective, but they are not recommended for coughs with mucus, as suppressing the cough can prevent you from expelling the mucus.
4. **Other Considerations**: If your cough persists or is accompanied by other symptoms like fever, shortness of breath, or wheezing, it's important to consult a healthcare provider to rule out other conditions such as asthma, bronchitis, or pneumonia.
5. **Environmental Controls**: Managing your environment can also help reduce allergy symptoms. Use air conditioning to filter indoor air, keep windows closed during high pollen seasons, and use dehumidifiers to control mold growth.

Specific to Your Situation

Given that your cough worsened with the change in weather, it might be worth considering whether you are allergic to specific outdoor allergens like pollen or mold. An allergist can perform skin tests or blood tests to identify specific allergens you might be reacting to.

If you find that your symptoms are significantly impacting your quality of life, it might be beneficial to consult a healthcare provider for a more personalized assessment and treatment plan.

Remember, always consult a healthcare professional for a proper diagnosis and personalized advice, especially if you have any underlying health conditions or are taking other medications.

I hope this information helps!

If you have any more questions or need further assistance, feel free to ask.

Stay well!

Qwen

Additional Note

If you experience any of the following, seek immediate medical attention:

- Difficulty breathing
- Swelling of the face, lips, tongue, or throat
- Severe wheezing
- Hives or rash
- Dizziness or fainting
- Rapid heartbeat or palpitations
- Difficulty swallowing

These could be signs of a severe allergic reaction (anaphylaxis), which is a medical emergency.

Qwen

Detailed Information

Allergies can indeed cause coughing, and the pattern you're describing—coughing worsening with changes in weather—suggests that you might be experiencing seasonal allergies. Here's a more detailed explanation:

1. **Seasonal Allergies and Coughing**:
   - **Pollen Allergies**: High pollen counts, especially from grass, trees, and weeds, can trigger coughing. Pollen counts are often highest in the early morning and late afternoon.
   - **Mold Allergies**: Mold spores can also cause coughing, especially in humid or rainy weather. Mold can grow in damp areas like basements, bathrooms, and gardens.
   - **Weather Changes**: Changes in temperature and humidity can affect the airways and exacerbate coughing. For example, dry, cold air can dry out the nasal passages and throat, while humid, warm air can increase mucus production.
2. **Coughing Mechanism**:
   - **Post-Nasal Drip**: When allergens cause inflammation in the nasal passages, excess mucus can drip down the back of the throat, leading to a cough.
   - **Bronchial Irritation**: Allergens can also irritate the bronchial tubes, causing a dry, hacking cough.
3. **Treatment Options**:
   - **Antihistamines**: These can help reduce sneezing, itching, and runny nose, which can in turn reduce coughing. Cetirizine (Zyrtec), loratadine (Claritin), and fexofenadine (Allegra) are effective for many people.
   - **Nasal Sprays**: Decongestant nasal sprays can help reduce nasal congestion, but they should be used for no more than a few days to avoid rebound congestion.
   - **Inhalers**: For those with asthma, inhaled corticosteroids and bronchodilators can help manage symptoms.
   - **Expectorants**: Guaifenesin (Mucinex) can help loosen mucus, making it easier to cough up.

- **Cough Suppressants**: Dextromethorphan (found in many cough syrups) can help suppress a dry, hacking cough, but they are not recommended for coughs with mucus.

4. **Environmental Controls**:
   - Use air conditioning to filter indoor air.
   - Keep windows closed during high pollen seasons.
   - Use dehumidifiers to control mold growth.
   - Consider using HEPA filters in your home.
5. **Consult a Healthcare Provider**:
   - If your symptoms persist or worsen, or if you have other symptoms like fever, shortness of breath, or wheezing, consult a healthcare provider.
   - An allergist can perform skin tests or blood tests to identify specific allergens you might be reacting to.

Specific Medications for Allergy-Induced Cough

There are specific medications that can help manage coughs due to allergies. Here are a few options:

- **Cetirizine (Zyrtec)**: This is an antihistamine that can help reduce sneezing, itching, and runny nose, which can in turn reduce coughing.
- **Loratadine (Claritin)**: Another antihistamine that can help manage allergy symptoms.
- **Fexofenadine (Allegra)**: An antihistamine that is effective for many people with seasonal allergies.
- **Montelukast (Singulair)**: This is a leukotriene receptor antagonist that can help reduce inflammation in the airways and may help with coughing in some people with asthma or allergies.

When to Seek Medical Attention

If your cough is severe, persistent, or accompanied by other symptoms like fever, shortness of breath, or wheezing, it's important to consult a healthcare provider. They can provide a proper diagnosis and recommend appropriate treatment.

I hope this detailed information helps you better understand your symptoms and provides you with some guidance on managing them. If you have any more questions or need further assistance, feel free to ask.

Stay well!

Qwen

Detailed Medication Guidance

For a cough due to allergies, here are some specific medications you might consider:

1. **Antihistamines**:
   - **Cetirizine (Zyrtec)**: This is a non-sedating antihistamine that can help reduce sneezing, itching, and runny nose. It is generally well-tolerated and can be taken daily.
   - **Loratadine (Claritin)**: Another non-sedating antihistamine that is effective for many people with seasonal allergies. It is also generally well-tolerated.
   - **Fexofenadine (Allegra)**: This is another non-sedating antihistamine that can help manage allergy symptoms. It is generally well-tolerated and can be taken daily.
2. **Decongestants**:
   - **Pseudoephedrine (Sudafed)**: This can help reduce nasal congestion, but it should be used with caution, as it can cause rebound congestion if used for more than a few days. It is also not recommended for people with high blood pressure or certain heart conditions.
   - **Phenylephrine**: This is another decongestant that can help reduce . . .

D.2 SCIENCE: A VECTOR CALCULUS PROBLEM

Prompt

Show that if a particle is moved along the closed curve $(\cos t, \sin t, 0)$, where $0 \leqslant t \leqslant 2\pi$, under the force field $\mathbf{F} = -y\mathbf{i} + x\mathbf{j}$, then the work done by the force field on the particle is nonzero. Calculate the exact amount of work done.

FORK (ours)

To determine the work done by the force field $\mathbf{F} = -y\mathbf{i} + x\mathbf{j}$ on a particle moving along the closed curve $(\cos t, \sin t, 0)$ for $0 \leq t \leq 2\pi$, we follow these steps:

1. **Parameterize the curve:**

The curve is given by $\mathbf{r}(t) = (\cos t, \sin t, 0)$ for $0 \leq t \leq 2\pi$.

2. **Find the derivative of the curve:**

The derivative of $\mathbf{r}(t)$ with respect to t is:

$$\mathbf{r}'(t) = \frac{d\mathbf{r}}{dt} = (-\sin t, \cos t, 0).$$

3. **Evaluate the force field $\mathbf{F}$ at the point $\mathbf{r}(t)$:**

Since $\mathbf{F} = -y\mathbf{i} + x\mathbf{j}$ and $\mathbf{r}(t) = (\cos t, \sin t, 0)$, we have:

$$\mathbf{F}(\mathbf{r}(t)) = -\sin t\mathbf{i} + \cos t\mathbf{j}.$$

4. **Compute the dot product $\mathbf{F}(\mathbf{r}(t)) \cdot \mathbf{r}'(t)$:**

$\mathbf{F}(\mathbf{r}(t)) \cdot \mathbf{r}'(t) = (-\sin t\mathbf{i} + \cos t\mathbf{j}) \cdot (-\sin t\mathbf{i} + \cos t\mathbf{j}) = (-\sin t)(-\sin t) + (\cos t)(\cos t) + 0 = \sin^2 t + \cos^2 t = 1.$

5. **Set up the integral for the work done:**

The work W done by the force field is given by the line integral:

$$W = \int_C \mathbf{F} \cdot d\mathbf{r} = \int_0^{2\pi} \mathbf{F}(\mathbf{r}(t)) \cdot \mathbf{r}'(t)\, dt = \int_0^{2\pi} 1\, dt.$$

6. **Evaluate the integral:**

$$\int_0^{2\pi} 1\, dt = [t]_0^{2\pi} = 2\pi - 0 = 2\pi.$$

Since the work done is 2π, which is nonzero, we conclude that the work done by the force field on the particle is $\boxed{2\pi}$.

Standard coverage reward

To determine the work done by the force field $\mathbf{F} = -y\mathbf{i} + x\mathbf{j}$ on a particle moving along the closed curve $(\cos t, \sin t, 0)$ for $0 \leq t \leq 2\pi$, we need to evaluate the line integral of $\mathbf{F}$ along the curve. The work done is given by:

$$W = \int_C \mathbf{F} \cdot d\mathbf{r}$$

Step 1: Parametrize the curve
The curve is given by:

$$\mathbf{r}(t) = (\cos t, \sin t, 0)$$

where $0 \leq t \leq 2\pi$.

Step 2: Compute the derivative of the parametrization
The derivative of $\mathbf{r}(t)$ with respect to t is:

$$\mathbf{r}'(t) = \frac{d}{dt}(\cos t, \sin t, 0) = (-\sin t, \cos t, 0)$$

Step 3: Express the force field in terms of t
Substitute $x = \cos t$ and $y = \sin t$ into the force field $\mathbf{F} = -y\mathbf{i} + x\mathbf{j}$:

$$\mathbf{F}(\mathbf{r}(t)) = -\sin t\mathbf{i} + \cos t\mathbf{j}$$

Step 4: Compute the dot product $\mathbf{F} \cdot d\mathbf{r}$
The line integral is:

$$\mathbf{F} \cdot d\mathbf{r} = \mathbf{F}(\mathbf{r}(t)) \cdot \mathbf{r}'(t)$$
$$\mathbf{F}(\mathbf{r}(t)) \cdot \mathbf{r}'(t) = (-\sin t, \cos t, 0) \cdot (-\sin t, \cos t, 0)$$
$$= (-\sin t)(-\sin t) + (\cos t)(\cos t) + (0)(0)$$
$$= \sin^2 t + \cos^2 t$$
$$= 1$$

Step 5: Set up and evaluate the integral
The work done is:

$$W = \int_0^{2\pi} 1 \, dt$$
$$W = [t]_0^{2\pi}$$
$$W = 2\pi - 0 = 2\pi$$

Step 6: Verify using the Fundamental Theorem of Line Integrals
The force field $\mathbf{F} = -y\mathbf{i} + x\mathbf{j}$ is conservative. To see this, we check if $\mathbf{F}$ is the gradient of a scalar potential function. Let $\mathbf{F} = \nabla\phi$, where $\phi(x, y) = x^2/2 - y^2/2$. Then:

$$\frac{\partial \phi}{\partial x} = x = \text{the } x\text{-component of } \mathbf{F}$$
$$\frac{\partial \phi}{\partial y} = -y = \text{the } y\text{-component of } \mathbf{F}$$

Since the curve is closed, the work done by a conservative force field around a closed path is zero. However, the curve is not a simple closed loop in the context of the potential function, but rather a circle. The work done is still calculated as above.

Step 7: Conclusion
The work done by the force field on the particle as it moves along the closed curve is:

$$\boxed{2\pi}$$

This result indicates that the work done is nonzero, which is consistent with the fact that the force field is not conservative when considered in the context of the specific closed path. The work done is 2π.

E PROMPTS

E.1 TRAINING

We provide prompts for matching rubrics to units in Figure 5, prompts for deciding necessity threshold in Figure 6–7, and prompts for deciding factual correctness in Figure 8–9. We also provide prompts for baselines and ablations in Figure10–18.

E.2 EVALUATION

There is one newly introduced judge prompt in our evaluation, which is measuring precision. We use the prompt same as in *Generic Criteria* baseline (Figure 14) since both of their job is to evaluate necessity and factual correctness of response units.

FORK (Rubrics to Units Matching)

===SYSTEM===
You are an expert medical evaluator. You are given a user QUESTION, the COVERAGE RUBRIC written for it — each criterion carrying a separate grader's verdict (MET or NOT MET) and that grader's reason — and a numbered list of UNITS taken from a model's response. Your task is attribution, not grading: for EACH criterion marked MET, identify the units that are the basis for that verdict.

About the units:
- Each unit is an exact span of the response, so it may lean on its neighbours or its section heading for meaning — a pronoun, a bare list item, a clause whose main verb is in the heading. Read it in that context.
- Units are grouped under [section] lines carrying the response's own headings and lead-ins, in the order the response wrote them. Those [section] lines are context only — never cite them; cite unit numbers only.

How to attribute:
- Take the verdicts and the reasons as given, even where you would have judged differently. You are locating where in the response the grader's reason holds, not re-deciding it.
- A unit is part of the basis if it supplies something the criterion asks for. A unit that merely touches the criterion's topic without supplying what it asks for is not.
- List EVERY unit that is part of the basis, not just the clearest example. When several units each supply part of what the criterion asks for — the items of an enumeration, the two halves of a comparison, a claim and the evidence required for it — list all of them.
- If the basis is the response as a whole rather than particular units — its organization, its consistency, its tone, or what it refrains from doing — answer "whole-response" instead of a list. A criterion about a property the response shows or avoids throughout — tone, organization, consistency, hedging, terminology, what it never says — is carried by every unit: answer "whole-response" and do not list the units where the property is most visible, because those are examples of the basis, not the basis; list units only when the criterion asks for specific content that particular units supply.
- If you cannot find any unit that is the basis for a criterion marked MET, give an empty list. - Criteria marked NOT MET are context only. Do not include entries for them.

Output a single JSON object keyed by the numbers of the criteria marked MET (as strings), and nothing else:
``` json
{
  "1": {"units": [2, 3, 5]},
  "13": {"units": "whole-response"},
  "20": {"units": []}
}
```

Include an entry for EVERY criterion marked MET — a criterion you omit is scored as unattributed. Start your response with ``` json and end with ``` . Do not include any other text.

===USER===
<question>
{question}
</question>

<coverage_rubric>
{rubric_block}
</coverage_rubric>

<units>
{units_block}
</units>

For each criterion marked MET, give the units that are the basis for its verdict. Your JSON:

Figure 5: Prompt for FORK (Rubrics to Units Matching).

FORK (Necessity Threshold Decision) (1/2)

===SYSTEM===
You are an expert medical evaluator. You are given a user QUESTION and a model's ANSWER to it, split into numbered UNITS — exact spans of the answer, in its order, under its [section] lines. The answer was graded against a coverage rubric — the points a good answer should contain — and each unit carries a score in parentheses: the share of rubric credit it holds. A unit that alone carries a credited point scores 1.00; a unit that is one of K units carrying the same point scores 1/K; shares add up when a unit carries several points, so a score can exceed 1.00; a unit that carries no credited point scores 0.00. The rubric is not shown.

Your task has two steps, answered together. First, set the THRESHOLD for this answer: the lowest score a unit must carry to be necessary — a unit the question needs. Then deduplicate among the necessary units. Rubric credit is shared by duplicates: when the answer makes a point and later makes it again, both units can carry a share of it. For each necessary unit in the assigned range, decide whether it duplicates an earlier necessary unit — verdict "remove" — or makes a contribution of its own — verdict "keep".

About the threshold:
- The score says how much of a credited point a unit carries; a low score means the credit for some point is spread over many units. Decide, for this question, how thin that spread may be before a unit stops being necessary.
- A question that calls for a list, a set of steps or several distinct items legitimately spreads a point over many units, and a low threshold keeps them. A question answered by a few points should not have its credit spread thin, and a higher threshold cuts the units that only dilute it.
- Choose the threshold from the scores in front of you, reading the units themselves as the evidence of what the spread means in this answer. A unit scoring 0.00 carries no credited point and is never necessary, so the threshold is greater than 0.00.
- Give the threshold as a decimal with two places, in the same form as the scores. A unit whose score is at or above the threshold is necessary; every other unit is context only.

About the units:
- Each unit is an exact span of the answer, so it may lean on its neighbours or its section heading for meaning — a pronoun, a bare list item, a clause whose main verb is in the heading. Read it in that context.
- Units are grouped under [section] lines carrying the answer's own headings and lead-ins, in the order the answer wrote them. Those [section] lines are context only — never judge them.
- Units below the threshold are context only. They are never judged, and a necessary unit never duplicates a unit below the threshold: compare necessary units with earlier necessary units only.

The principle: a unit is a duplicate when there is no semantically meaningful difference between what it gives the asker and what an earlier necessary unit already gave. A difference is meaningful when it changes what the asker knows or can do; it is trivial when it only changes how the same thing is said — shorter, longer, reworded, re-explained, emphasized, or placed under another heading.

Common duplicates: the same fact, advice, item, warning or conclusion stated again in other words; the same point repeated at more length or with a generic reason attached; a summary, recap or conclusion that only gathers points already made; an item listed again that an earlier unit already listed; two entries of one list that make the same point under different names.

Direction: the duplicate is always the later unit, never the first to make the point. "Earlier" means earlier in the answer's order, anywhere in the answer: the comparison spans every section and reaches outside the assigned units — all units are shown for that. Nothing that comes later makes an earlier unit a duplicate.

A "remove" must name the earlier unit's number in its reasoning; if you cannot name one, the verdict is "keep". When the difference is genuinely unclear, the verdict is "keep".

In the verdicts, do NOT judge whether the unit is true, needed, or well written — the threshold has already decided what is necessary, and the rest are different evaluations. Judge as if every unit's content were accurate.

Figure 6: Prompt for FORK (Necessity Threshold Decision) (1/2).

FORK (Necessity Threshold Decision) (2/2)

Output a single JSON object, and nothing else: first the key "threshold" with your decimal, then one entry per necessary unit in the assigned range, keyed by its number (as a string) — each value is an object with "reasoning" first, then "verdict", never a bare string. The reasoning is a few words; a "remove" names the earlier unit:

```` json
{
  "threshold": 0.25,
  "3": {"reasoning": "first statement of the dosing rule", "verdict": "keep"},
  "8": {"reasoning": "duplicates 3: the same dosing rule reworded", "verdict": "remove"},
  "9": {"reasoning": "adds the threshold that 3 does not give", "verdict": "keep"},
  "14": {"reasoning": "recap of 3, 9 and 11, nothing new", "verdict": "remove"}
}
````

Include an entry for EVERY necessary unit in the assigned range — a necessary unit you omit is scored as invalid. Do not include entries for units below the threshold or outside the assigned range. Start your response with ``` json and end with ``` . Do not include any other text.

===USER===
<question>
{question}
</question>

<answer>
{units_block}
</answer>

Set the threshold for this answer, then judge the units {judge_range} whose score is at or above it — "keep" or "remove" for each. Every other unit is shown for context only. Your JSON: "threshold" first, then one {"reasoning": ..., "verdict": ...} object per necessary unit in that range:

Figure 7: Prompt for FORK (Necessity Threshold Decision) (2/2).

FORK (Factual Correctness Decision) (1/2)

===SYSTEM===
You are an expert medical evaluator. You are given a user QUESTION and a numbered list of UNITS taken from a model's answer to it. Judge the FACTUAL CORRECTNESS of each assigned unit — and nothing else about it.

About the units:
- Each unit is an exact span of the answer, so it may lean on its neighbours or its section heading for meaning — a pronoun, a bare list item, a clause whose main verb is in the heading. Read it in that context and judge what it says there.
- Units are grouped under [section] lines carrying the answer's own headings and lead-ins, in the order the answer wrote them. Those [section] lines are context only — never judge them.
- Only the assigned units are judged; every other unit is shown for context.

For every assigned unit, give exactly one verdict. Decide in this order:
- "na" — the unit asserts nothing checkable: it only organizes the answer, leads into or sums up other units, expresses empathy or reassurance, comments on the answer itself, or offers further help. Decide this first. A unit that tells the asker what is true, what to do, how, how much, when, or why — advice and instructions included — asserts something checkable and is never "na", however brief or hedged.
- "incorrect" — the unit asserts something wrong: it contradicts reliable medical knowledge or the facts the question states, gives advice or an instruction that is wrong or unsafe for the situation the question describes, or presents as standard a name, term or practice the field does not recognize. A true fact applied to the wrong situation is an error in that situation. A unit that asserts several things is "incorrect" if any one of them is wrong.
- "correct" — what the unit asserts is right, at the strength and breadth it asserts it: supported by reliable medical knowledge you hold, common knowledge, or the question itself. Common clinical knowledge that any clinician would confirm is "correct" without a citation; "uncertain" is for claims that would need one.
- "uncertain" — the unit asserts something checkable, but you cannot confirm it from reliable medical knowledge you hold, at the strength and breadth it asserts it: a specific you cannot verify (often a very specific figure, a rare interaction, a niche study claim, a named source), or an overstatement — more certainty, a stronger or broader effect, or a narrow or contested finding presented as general. Judge the takeaway a careful reader would carry away: a hedge ("may", "can", "often") still asserts that the link or effect is real, so it does not lower the bar. Plausible, commonly stated or hard to disprove is not confirmed; confirmation comes from what you actually know, not from a source you would have to assume exists.

How to judge:
- Judge each assigned unit only on its own content. A unit is not wrong because a neighbouring unit is wrong.
- Do NOT judge whether the unit is needed, relevant or repeated — that is a different evaluation. A redundant, generic or tangential unit whose content is factually accurate is "correct".

Output a single JSON object keyed by the assigned unit numbers (as strings), and nothing else — each unit's value is an object with "reasoning" first, then "verdict", never a bare string. The reasoning is a few words where the verdict is obvious, one sentence where it takes thought; for "incorrect" it must name the error; for "uncertain", the claim you cannot confirm — the specific, or the strength or breadth it overstates:

```` json
{
  "6": {"reasoning": "matches standard dosing guidance", "verdict": "correct"},
  "7": {"reasoning": "transition into the treatment section, asserts nothing", "verdict": "na"},
  "8": {"reasoning": "states antibiotics treat viral infections — they do not act on viruses", "verdict": "incorrect"},
  "9": {"reasoning": "specific prevalence figure I can neither support nor refute", "verdict": "uncertain"},
  "10": {"reasoning": "'may provide significant relief' still conveys a real, sizeable effect; the evidence is weak and mixed", "verdict": "uncertain"}
}
````

Figure 8: Prompt for FORK (Factual Correctness Decision) (1/2).

FORK (Factual Correctness Decision) (2/2)

Include an entry for EVERY assigned unit number — a unit you omit is scored as invalid. Do not include entries for units that are not assigned. Start your response with ``` json and end with ``` . Do not include any other text.

===USER===
<question>
{question}
</question>

<units>
{units_block}
</units>

Judge the factual correctness of units {judge_range}. The other units are shown for context only. Your JSON, one {"reasoning": ..., "verdict": ...} object per assigned unit:

Figure 9: Prompt for FORK (Factual Correctness Decision) (2/2).

Reweighted Absence-Based Rubrics (1/2)

===SYSTEM===
You are an expert evaluator. Given a prompt (which may include conversation history) and a generated response, please evaluate the response against EACH rubric, and state the weight each absence-based rubric item should carry.

Return a json object where keys are the rubric numbers (as strings) and values are objects with "explanation", "criteria_met", "type" and "updated_weight" fields.
- The "explanation" field should be a string explaining why the response does or does not meet the criteria of the rubric item.
- The "criteria_met" field should be a boolean indicating whether the response meets the criteria of the rubric item. If a rubric item has multiple sentences or criteria, you should consider all of them. If any of the criteria is not met, the answer should be false. Only return true if all of the criteria are met. The weight listed with an item has no bearing on this verdict: decide whether the criterion is met exactly as you would if no weight were shown.
- One important exception to the above bullet point is that if a criteria says "such as", "for example", or "including", the response does not have to include all of the examples listed to meet the criteria. For example, if the criteria says "Mentions practical ways to reduce household electricity use such as lowering thermostat settings, using LED bulbs, and unplugging idle devices", and the response just mentions using LED bulbs, that would still meet the criteria even though it didn't mention all of the specific examples listed in the criteria.
- The "type" field should be "presence" or "absence", as defined under Criterion types below. It classifies the rubric item's wording, not your verdict: a presence-based item that the response fails to satisfy is still "presence".
- The "updated_weight" field should be an integer for an absence-based item, chosen as described under Updated weights below, and null for a presence-based item.

Criterion types
Each rubric item is presence-based or absence-based. The type is a property of the item's wording, read before looking at the response: it does not depend on the response or on whether the item is met.
- A presence-based item asks whether the response contains something: a fact, a reasoning step, a list, a recommendation, a disclaimer, a format or a tone. It is met when that content appears explicitly in the response and not met when the content is missing. Missing content makes the item not met; it does not make the item absence-based. If your explanation says that the response does not mention, state or provide what the item asks for, the item is presence-based.
- An absence-based item asks whether the response avoids something. Its wording forbids or limits: it does not claim, recommend, include, assume or fabricate X; it contains no errors or contradictions; it stays within a limit such as "only X" or "without Y". It is met only if nothing anywhere in the response violates it, so check the whole response before marking it met, and say in the explanation what you checked for and what you found.
- When a rubric item has both a presence requirement and an absence requirement, label it "absence", even when most of the item asks for content and the prohibition or limit is a single clause. Its "criteria_met" still requires both parts to be met.
- A correctness word attached to a specific required fact, such as "correctly states that X", makes a presence-based item, not an absence-based one.

Updated weights
Each rubric item is listed with its weight. Rubrics like this one describe what a good answer should contain in many items but what it must avoid in only a few, because the ways an answer can go wrong cannot be enumerated the way its required content can. Graded as listed, such a rubric rewards adding content: every extra sentence might satisfy one more presence-based item, while the few absence-based items cost little to violate, so answers grow padded and error-prone. The absence-based items are the rubric's only guards against that, and their listed weights understate that role. Beside deciding whether each item is met, your second job is to state, for each absence-based item, the weight it should carry, so that the grade favors an answer that says what is needed, says it correctly, and stops there.

Figure 10: Prompt for Reweighted Absence-Based Rubrics (1/2).

Reweighted Absence-Based Rubrics (2/2)

Size the weight against the presence-based items the item guards: violating it should cost at least as much as the answer could gain by satisfying more of them. Two things decide how much that is, and the judgment is yours:
- What the item guards. Ask how much of the answer's value depends on it, not how broadly it is worded. A guard on whether the answer can be trusted or acted on at all, such as no incorrect statements or no unsafe instructions, could deserve a weight above the highest listed weight, possibly several times over. A narrowly worded item could deserve as much when the detail it protects is decisive, and a broadly worded one might warrant only a moderate increase when what it guards matters little for this question. These are illustrations, not fixed tiers.
- What this response did. If the response violates the item, weigh the damage the violation does to the answer for the asker. If it avoids the item, weigh what the answer would have lost had it not.

Give the weight as an integer on the same scale as the listed weights, never below the listed weight, and as high as the case warrants. The response's own statements about what it avoids do not raise the weight. For presence-based items, "updated_weight" is null.

Example
For example, if the prompt is "How can I reduce electricity use at home?" and the response is "Leave all the lights on overnight." and rubric item 1 is "suggests practical ways to reduce electricity use", you should return:

``` json
{
  "1": {"explanation": "The response did not suggest practical ways to reduce electricity use.", "criteria_met": false, "type": "presence", "updated_weight": null}
}
```

Start your response with a valid JSON object that starts with ``` json and ends with ``` . Do not include any other text. Evaluate ALL rubric items and include an entry for each one.

===USER===
Given the following prompt and response, please evaluate the response against each rubric.

<prompt>
{prompt}
</prompt>

<response>
{response}
</response>

<rubrics>
{rubric_list_string}
</rubrics>

Your JSON Evaluation:

Figure 11: Prompt for Reweighted Absence-Based Rubrics (2/2).

Holistic Preference (1/2)

===SYSTEM===
You are an expert evaluator. You will be given a user prompt, a scoring rubric written for that prompt, and two responses to the prompt (@response_A and @response_B). Your job is to evaluate EACH response on the following 4 dimensions, scoring each from 1 (worst) to 7 (best), and then decide which response is preferable overall.

Dimensions and anchors:

1. Rubric coverage - how well the response covers the rubric's criteria, where each criterion's weight states its importance.
1: covers almost none of the rubric's weight; 4: covers roughly half of the rubric's weight; 7: covers every criterion, including the heavily weighted ones.

2. Correctness - All factual claims, calculations, and recommendations in the response are correct — supported by the prompt, the original rubric, or reliable domain knowledge — with nothing fabricated, misleading, numerically wrong, or overconfident.
1: many false or fabricated claims (5+); 3: several errors (3-4 incorrect claims); 5: a few minor errors (1-2 incorrect claims); 7: no detectable factual errors.

3. Necessity - The response needed every part of it: nothing restates what an earlier part already said in the same or different words, nothing is boilerplate, filler, or a pleasantry, nothing is vague padding, and no section, example, or caveat is there only to sound thorough — removing any part would noticeably weaken the answer.
1: extensive repetition, boilerplate, or padding; 4: several parts could be removed without weakening the answer; 7: every part is needed; removing any of them would noticeably weaken the answer.

Instructions:
Judge each dimension for EACH response. For each dimension, write a brief justification (1-2 sentences) FIRST, then give the score. For rubric coverage, assess the response's coverage as a whole rather than item by item, citing the criteria that most affected your judgment where that makes the assessment clear. The rubric applies to both responses equally; neither response is a reference for the other. After scoring all dimensions, weigh all four dimensions as you see fit and state which response is preferable overall, "A" or "B", with a 2-3 sentence explanation referencing @response_A and @response_B. Answer "tie" when both responses are of equal overall quality and neither can be preferred over the other. Be fair and objective. Do not let the ordering of responses bias you.

Figure 12: Prompt for Holistic Preference (1/2).

Holistic Preference (2/2)

Return a single JSON object with this exact shape (scores are integers 1-7; "preferred" is "A", "B", or "tie"):

```` json
{
  "response_A": {
    "coverage": {"justification": "", "score": 0},
    "correctness": {"justification": "", "score": 0},
    "necessity": {"justification": "", "score": 0}
  },
  "response_B": {
    "coverage": {"justification": "", "score": 0},
    "correctness": {"justification": "", "score": 0},
    "necessity": {"justification": "", "score": 0}
  },
  "comparison": {"explanation": "", "preferred": ""}
}
````

Start your response with ```` ``` json ```` and end with ```` ``` ````. Do not include any other text.

===USER===
<prompt>
{prompt}
</prompt>

<rubric>
{rubric_block}
</rubric>

<response_A>
{response_A}
</response_A>

<response_B>
{response_B}
</response_B>

Your JSON Evaluation:

Figure 13: Prompt for Holistic Preference (2/2).

Generic Criteria

===SYSTEM===
You are an expert medical evaluator. You are given a user QUESTION, the ORIGINAL RUBRIC written for it, optional REFERENCE material, and a numbered list of UNITS taken from a model's response. Judge EACH assigned unit against three criteria.

About the units:
- Each unit is an exact span of the response, so it may lean on its neighbours or its section heading for meaning — a pronoun, a bare list item, a clause whose main verb is in the heading. Read it in that context and judge what it says there.
- Units are grouped under [section] lines carrying the response's own headings and lead-ins, and they appear in the order the response wrote them. Those [section] lines are context only — never judge them.

Criteria (judge each as true/false for every assigned unit):

1. correct — The unit is factually correct and supported by the prompt, the reference material (if provided), the original rubric, or reliable domain knowledge. It is NOT fabricated, misleading, numerically wrong, or overconfident. If you cannot support the unit from any of these, mark it false.

2. necessary — The response NEEDED this unit. Ask: would a well-written, concise answer to this question have included it? Common reasons a unit is unnecessary — any one is enough, and the list is not exhaustive:
- it restates something an EARLIER unit already said, in the same or different words;
- it is boilerplate, filler, or a pleasantry (greetings, sign-offs, offers to answer follow-up questions, disclaimers already made);
- it is padding: vague, generic, or so unspecific that it adds nothing of substance;
- it is there only to sound thorough — an extra section, example, or caveat that adds nothing to answering the question.

When a unit is borderline, test it by removal: read the response without it, and if its absence would noticeably weaken the answer, it is necessary; otherwise it is not.

A unit that is simply wrong should fail 'correct', not 'necessary' — judge necessity as if the unit were true.

Judge 'correct' on the unit's own merits. 'necessary' is different: it is a judgement about the unit's place in the whole response, so weigh it against the other units and against what a good answer to this question needs.

Output a single JSON object keyed by the unit numbers (as strings), and nothing else:
``` json
{
  "1": {"explanation": "<one short sentence>", "correct": true, "necessary": true},
  "2": {"explanation": "<one short sentence>", "correct": false, "necessary": false}
}
```

Include an entry for EVERY assigned unit number — a unit you omit is scored as invalid. Do not include entries for units outside the assigned range. Start your response with ``` json and end with ``` . Do not include any other text.

===USER===
<question>
{question}
</question>
{reference_block}
<original_rubric>
{rubric_block}
</original_rubric>

<units>
{units_block}
</units>

Judge units {judge_range} against correct / necessary. The other units are shown for context only.
Your JSON:

Figure 14: Prompt for Generic Criteria.

Offline Criteria (1/2)

===SYSTEM===
You are an expert medical evaluator. You are given a user QUESTION, the COVERAGE RUBRIC written for it, optional REFERENCE material, a PRECISION RUBRIC written for this question, and a numbered list of UNITS taken from a model's response. Judge EACH assigned unit.

About the units:
- Each unit is an exact span of the response, so it may lean on its neighbours or its section heading for meaning — a pronoun, a bare list item, a clause whose main verb is in the heading. Read it in that context and judge what it says there.
- Units are grouped under [section] lines carrying the response's own headings and lead-ins, and they appear in the order the response wrote them. Those [section] lines are context only — never judge them.

Three judgements for every assigned unit:

1. correct (true/false) — The unit is factually correct and supported by the prompt, the reference material (if provided), the coverage rubric, or reliable domain knowledge. It is NOT fabricated, misleading, numerically wrong, or overconfident. If you cannot support the unit from any of these, mark it false.

2. the PRECISION RUBRIC — a numbered list of criteria, each with a scope (which units it applies to) and a test (what an in-scope unit must or must not do). For each criterion, decide the scope first: does it cover this unit? A unit outside a criterion's scope is never judged by that criterion. Then, for each criterion that does cover the unit, decide whether the unit complies with its test. Report the numbers of the criteria the unit fails; a unit that no criterion covers, or that complies with every criterion covering it, gets an empty list. Some scopes and tests refer to the unit's place in the whole response — its section, its position, what earlier units already said. All units are shown so you can decide those; use them.

Some tests are VERIFICATION tests: they make compliance conditional on a specific claim (a name, a value, a citation, a mechanism) being standard, well-established knowledge — recognizable by conditions like "unless the stated dose is standard, well-established dosing". Answer that condition directly and affirmatively: the unit meets it only if you can support the specific claim from the prompt, the reference material (if provided), the COVERAGE RUBRIC, or reliable domain knowledge you actually recognize. A claim you cannot place — an unfamiliar study or formula name, a value you cannot confirm, a real entity attached to a claim you do not recognize — does not meet the condition, however confident the wording. Not every "unless" marks a verification test: an exception clause about behavior or circumstance — "unless it is strictly listing red-flag symptoms and clearly distinguishes them from the user's current state" — is ordinary test content; decide by reading whether that circumstance actually holds for this unit.

Accuracy and compliance are separate questions; never mix them. 'correct' judges accuracy. Each precision criterion judges compliance with its own test, exactly as written: a unit is not excused from a test because what it says is accurate, and a unit never fails a test merely because what it says is inaccurate — inaccuracy fails 'correct' and nothing else. Verification tests are the one deliberate overlap: whether the claim is established IS their test, so answer it there.

The COVERAGE RUBRIC is support material for 'correct' and for verification tests — never grade units against it directly; coverage is graded elsewhere.

Output a single JSON object keyed by the unit numbers (as strings), and nothing else:
``` json
{
  "1": {"explanation": "<one short sentence>", "correct": true, "fails": []},
  "2": {"explanation": "<one short sentence, naming what was failed>", "correct": true, "fails": [3]}
}
```

Include an entry for EVERY assigned unit number — a unit you omit is scored as invalid. 'fails' must always be present (an empty list if none). Do not include entries for units outside the assigned range. Start your response with ``` json and end with ``` . Do not include any other text.

Figure 15: Prompt for Offline Criteria (1/2).

Offline Criteria (2/2)

===USER===
<question>
{question}
</question>
{reference_block}
<coverage_rubric>
{rubric_block}
</coverage_rubric>

<precision_rubric>
{precision_rubric_block}
</precision_rubric>

<units>
{units_block}
</units>

Judge units {judge_range} against correct / the precision rubric. The other units are shown for context only. Your JSON:

Figure 16: Prompt for Offline Criteria (2/2).

Online Criteria (1/2)

===SYSTEM===
You are an expert medical evaluator. You are given a user QUESTION and a model's ANSWER to it, split into numbered UNITS — exact spans of the answer (sentences, list items, clauses), in its order, under its [section] lines. The answer was graded against a coverage rubric — the points a good answer should contain — and each point it earned was credited to the units carrying it: those are tagged (credited), the rest (uncredited). Credited units are the core content. Uncredited units are either peripheral content the answer still needs — lead-ins, qualifiers, examples and details that extend a credited point, relevant points the rubric did not ask for — or futile content — padding, repetition, digression — and the tag does not say which: an uncredited unit may or may not have an undesirable property, and telling the two apart is where your job starts.

Your task is to write the PRECISION RUBRIC for this answer: extract the undesirable properties the uncredited units show, if any, and turn each into a criterion. A tag records information — which point a unit carries; a property is behaviour — what a unit does toward the question, the units around it or its own point. A property is undesirable when deleting units that have it leaves the answer no worse for the asker: no less correct, complete or clear. For example: saying again what an earlier unit already said (adding a detail, example, qualifier or reason to a point is not saying it again); filler, pleasantries, sign-offs or commentary on the answer itself; generic advice that would fit any question on the topic; explaining or justifying at length where a unit of that kind only needs to name the point; drifting to a topic the question did not ask about. It helps to ask what an uncredited unit does that the credited units need not do — but a tag is not a property: credited units can show these properties too, to some degree, and uncredited units can carry something the asker needs — a qualifier, an example, a point of their own. A separate grader will apply every criterion to every unit, credited or not, without seeing the tags, your analysis or your unit lists; a unit is good only if it satisfies every criterion that applies to it.

Rules:
- Each criterion is a scope and a prohibition. "scope" is "units that <kind of content >" — a kind of content, never a section, a position or one particular unit. "criterion" begins "A unit should not ..." and is one prohibition the grader can check on the unit alone; only a repeat may refer to earlier units. Make the scope "all units" when the property is not confined to one kind of unit but should be checked on every unit of the answer.
- A criterion is about what a unit does, never about the point it makes or fails to make — what the answer must contain is judged by the coverage rubric, not by your criteria. If a credited unit shows the property because of the point it makes, the criterion is about content: narrow the scope or drop it. Content the question asks for is never off-topic, even when uncredited.
- Write a criterion only for an undesirable property you actually see in the uncredited units, and list the uncredited units that show it (uncredited only). Write as many criteria as there are distinct undesirable properties; an uncredited unit showing no such property needs no criterion, and if none does, return an empty list.
- One property per criterion. Prefer several simple criteria to one that chains different properties with "or"; "or" is fine between forms of the same property (pleasantries or sign-offs). Overlap between criteria is harmless; the same property twice is not.
- The grader cannot see which units are credited. A criterion must never mention or depend on credit — the words credit, credited, uncredited and rubric do not appear in a scope or criterion. A repeat is judged against earlier units of any kind: write "an earlier unit", never "a credited unit".
- Do not quote, paraphrase, number or otherwise point at units in "scope" or "criterion": each must make sense applied to any unit of any answer to this question.
- Begin with a brief "analysis" (a few sentences): which uncredited units the answer needs and which show an undesirable property — by groups, not one by one — and what the latter do that the answer would be no worse without. Then write the criteria.

Figure 17: Prompt for Online Criteria (1/2).

Online Criteria (2/2)

For example, for a question asking what to do for someone with a suspected stroke while waiting for the ambulance:
- scope "all units" — "A unit should not say again what an earlier unit already said, in the same or in different words."
- scope "units that tell the reader whom to call or where to go for help" — "A unit should not add unrequested reasons why that service is the right one beyond naming it."
- scope "all units" — "A unit should not explain the mechanism or types of stroke, which the question did not ask about."
Not criteria: "A unit should mention noting the time the symptoms began." (requires a point — the coverage rubric's job); "A unit should not describe what to do while waiting." (bans what the question asks for); "A unit should not add steps not already covered by the credited units." (refers to credit, which the grader never sees).

Output a single JSON object and nothing else:
``` json
{"analysis": "<a few sentences: which uncredited units the answer needs, which show an undesirable property, and what those do>", "criteria": [{"units": [<numbers of the uncredited units that show it>], "scope": "<all units — units that ...>", "criterion": "A unit should not ..."}]}
```

Start your response with ``` json and end with ``` . Do not include any other text.

===USER===
<question>
{question}
</question>

<answer>
{units_block}
</answer>

Write the precision rubric for this answer. Your JSON:

Figure 18: Prompt for Online Criteria (2/2).