

RUBRICWORLD: World Model-Guided Rubric Optimization for LLM Graders

Hongji Pu¹, Hang Li², Yucheng Chu², Jiliang Tang²

¹University of Illinois Urbana-Champaign, Urbana, IL, USA

²Michigan State University, East Lansing, MI, USA

Abstract

Rubric optimization improves LLM grading by revising scoring instructions while keeping the grader’s parameters fixed. Existing optimizers often judge edits by aggregate validation performance, limiting the supervision they retain about which errors an edit corrects or introduces. Testing each candidate also requires rescored validation answers, which limits how many edits can be explored within a fixed budget. We propose RUBRICWORLD, a framework that learns from previous edit evaluations to guide further search. Its world model predicts an edit’s correction and degradation rates at score boundaries, conditioned on the current rubric state. The training signal comes from paired grading results: scores for the same answers before and after an edit. Predictions select one candidate for direct evaluation, and held-out confidence bounds determine whether it corrects enough target errors while keeping new errors within prespecified limits. Measured effects update the model even when the edit is rejected. Across EIR (National), ASAP2.0, and ASAP-SAS with three frozen LLM graders, RUBRICWORLD achieves the highest score in 20 of 27 metric comparisons. On ASAP2.0, it raises `gpt-5-mini` accuracy from 18.20% with the original rubric to 46.70%, exceeding the strongest baseline, APO, by 22.60 percentage points.

1 Introduction

Automatic scoring has evolved from essay graders based on hand-crafted features to learned scoring systems (Page, 1966; Attali and Burstein, 2006; Taghipour and Ng, 2016). Large language models (LLMs) can score open-ended answers using natural-language rubrics, examples, and scoring criteria (Mansour et al., 2024; Henkel et al., 2024; Hashemi et al., 2024; Zhou et al., 2025). Vague or incomplete rules can cause grading errors. Revising the rubric can improve grading without re-training the model (Chu et al., 2025, 2026), while

keeping the changes available for teacher review (Henkel et al., 2025). Testing each revision, however, consumes model calls and tokens. Effective optimization therefore depends on how much the optimizer learns from each evaluation.

Existing rubric and prompt optimizers generate revisions from grading errors, textual critiques, or evaluation traces (Pryzant et al., 2023; Chu et al., 2025, 2026; Agrawal et al., 2026). They test revisions on labeled examples and use the results to guide search. However, they do not explicitly learn from past edits to predict which errors a new edit will correct or introduce under the current rubric. When candidate quality is summarized by validation accuracy, corrections can outweigh and conceal newly introduced errors. For example, raising the requirements for score 4 may correctly lower over-scored answers from 4 to 3, but also lower answers that deserve 4. The same net accuracy gain could reflect a few corrections or many corrections offset by many new errors. Accuracy alone gives limited guidance on how to preserve the corrections while avoiding the new errors.

Finding a suitable revision is also costly. Several edits may address the same problem, and testing each requires the grader to rescore validation answers (Yang et al., 2024; Opsahl-Ong et al., 2024). This limits the number of candidates explored within a fixed budget. Moreover, earlier revisions alter the rubric under which later edits are evaluated: adding or strengthening one rule can change another edit’s effect (Sclar et al., 2024). Reusing past evaluations therefore requires accounting for the current rubric, not just the candidate text.

To address these problems, we propose RUBRICWORLD, a world model-guided framework for rubric optimization. Here, a world model predicts the grading effect of an edit from the current rubric state and the edit itself. This lets the optimizer reuse earlier evaluations while accounting for the rubric changes made since those

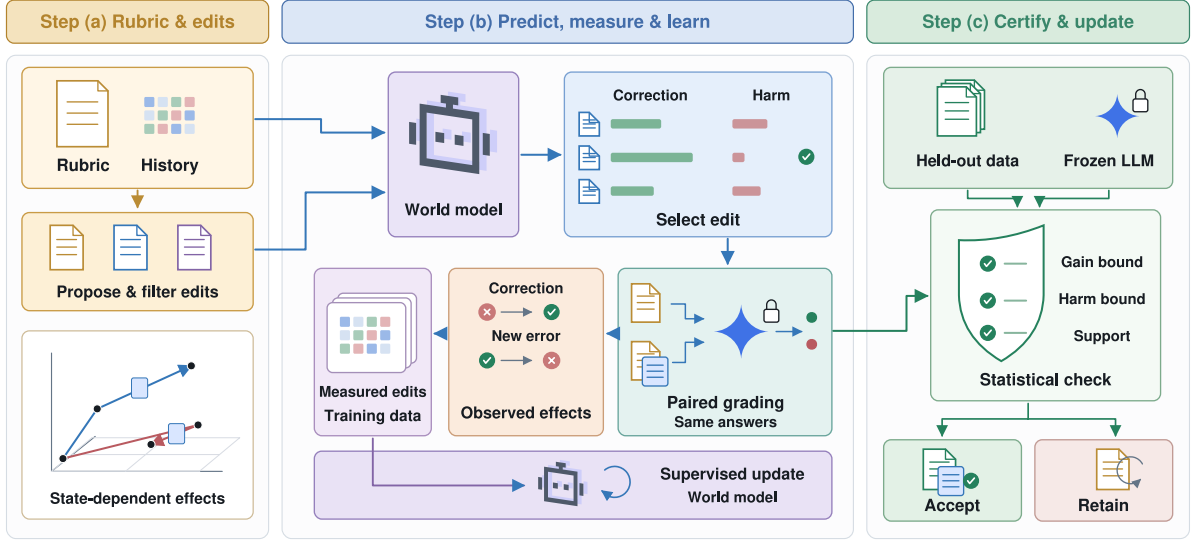


Figure 1: **Overview of RUBRICWORLD.** (a) The current rubric and observed grading errors guide the generation of edits with a limited scope. (b) The world model predicts corrections and degradations for each candidate and guides the choice of an edit for evaluation. Paired grading decisions before and after the edit provide learning feedback, including for edits that are later rejected. (c) Confidence bounds on held-out answers determine whether the edit is accepted. The LLM grader’s parameters remain fixed throughout.

evaluations. It can then screen proposed edits before spending grader calls on them.

We train the model on *paired grading results*: the scores assigned to the same answers before and after an edit, compared with their human reference scores. To describe what changed, we express each score as decisions at ordinal score boundaries (McCullagh, 1980; Cao et al., 2020). For example, a score of 4 lies above the $1|2$, $2|3$, and $3|4$ boundaries. At each boundary, the paired results reveal which incorrect decisions became correct and which correct decisions became incorrect. Together, these correction and degradation rates define the *edit effect*, including effects away from the edit’s intended boundary. The world model learns these rates from the edit and its interaction with the current rubric state.

In each round, RUBRICWORLD uses the predicted effects to choose one candidate for direct evaluation, reducing the number of candidates that require rescoring answers. Separate held-out answers determine whether to accept the edit: confidence bounds must support sufficient target correction and limited degradation at the boundaries chosen for certification. The measured training-set effect is retained even when the edit is rejected, so that unsuccessful attempts also inform later selections.

Our contributions are:

- We use paired grading results to measure an edit’s corrections and degradations at each ordinal score boundary.
- We introduce RUBRICWORLD, which learns to predict these effects under the current rubric and uses them to select candidates, with independent held-out checks for rubric updates.
- We evaluate grading quality, optimization cost, and accumulated errors on EIR (National), ASAP2.0, and ASAP-SAS with three frozen LLM graders.

2 Related Work

Rubric-Guided LLM Evaluation. G-Eval uses task-specific evaluation steps to guide LLM scoring, while Prometheus learns fine-grained evaluation from supplied rubrics (Liu et al., 2023; Kim et al., 2024a). LLM-Rubric further combines and calibrates criterion-level judgments for more structured evaluation (Hashemi et al., 2024). These methods primarily study how a given scoring rubric is applied or calibrated, rather than how it should be revised repeatedly from observed grading errors (Kim et al., 2024a; Hashemi et al., 2024).

Feedback-Driven Prompt and Rubric Optimization. APO and GEPA improve instructions using textual feedback or evaluation traces, while GradeOpt derives grading guidelines from misgraded answers (Pryzant et al., 2023; Agrawal et al.,

2026; Chu et al., 2025). PromptAgent formulates prompt refinement as sequential planning, whereas MIPRO learns a surrogate objective to guide candidate search (Wang et al., 2024; Opsahl-Ong et al., 2024). Recent optimizers also introduce preservation or degradation constraints to reduce losses on previously correct behavior (Xu et al., 2026; Satheesha et al., 2026). These approaches improve candidate generation, search, or update selection, but do not explicitly learn the paired edit effect conditioned on the current rubric state; this matters because earlier revisions change the state under which later edits are applied (Wang et al., 2024; Opsahl-Ong et al., 2024).

World Model Learning. Recent world-model methods learn action-conditioned dynamics and use predicted outcomes to guide subsequent planning or control (Hansen et al., 2024; Hafner et al., 2025). RUBRICWORLD adopts this predictive view at the level of a single rubric edit: rather than predicting only a scalar prompt score, it models paired corrections and degradations across score boundaries under the current rubric state. The model predicts the effect of one edit at a time. These predictions guide candidate selection, and separate held-out evidence determines whether to accept the edit.

3 Methodology

RUBRICWORLD is a framework for improving a frozen LLM grader by editing its rubric. It has three steps: generate candidate edits, predict their effects, and check the selected edit before accepting it (Figure 1). Each round, the generator proposes edits for a group of observed grading errors. The world model predicts which errors each edit will correct or introduce under the current rubric and selects one candidate for direct evaluation. Independent held-out answers determine whether to accept it. Its measured training-set effect updates the world model even if the rubric is retained.

3.1 Measuring an Edit’s Effect

An edit can correct some decisions while making others wrong. We measure these two effects separately at each score boundary. Let f be the frozen LLM grader, R_t the rubric at round t , and $F_f(x; R)$ its score for answer x under rubric R . For answer x_i with human score y_i , the current prediction is $\hat{y}_i^{(t)} = F_f(x_i; R_t)$. Applying edit p gives the temporary rubric $R_t \oplus p$ and score $\hat{y}_i^p = F_f(x_i; R_t \oplus p)$;

$\oplus$ denotes the text modification. Grading the same answers under both rubrics gives the paired observations.

For $K = y_{\max} - y_{\min} + 1$ consecutive integer scores, the set $\mathcal{B} = \{y_{\min}, \dots, y_{\max} - 1\}$ contains the $K - 1$ boundaries, where b separates scores b and $b + 1$ (McCullagh, 1980; Cao et al., 2020). At each boundary, the question is whether an answer deserves a score above b . The human-score label is $z_{i,b} = \mathbb{I}(y_i > b)$, and the grader’s before- and after-edit decisions are $\hat{z}_{i,b}^{(t)} = \mathbb{I}(\hat{y}_i^{(t)} > b)$ and $\hat{z}_{i,b}^p = \mathbb{I}(\hat{y}_i^p > b)$, where $\mathbb{I}$ is the indicator function.

Let $\mathcal{F}_{t,b}$ contain the initially incorrect decisions in the evaluated batch, and $\mathcal{A}_{t,b}$ the initially correct ones. Correction is the fraction of $\mathcal{F}_{t,b}$ made correct by the edit; degradation is the fraction of $\mathcal{A}_{t,b}$ made incorrect:

$$C_{t,b}(p) = \frac{\sum_{i \in \mathcal{F}_{t,b}} \mathbb{I}(\hat{z}_{i,b}^p = z_{i,b})}{|\mathcal{F}_{t,b}|}, \quad (1)$$

$$D_{t,b}(p) = \frac{\sum_{i \in \mathcal{A}_{t,b}} \mathbb{I}(\hat{z}_{i,b}^p \neq z_{i,b})}{|\mathcal{A}_{t,b}|}. \quad (2)$$

For example, correcting 6 of 20 initially wrong decisions gives $C = 0.30$, while making 2 of 80 initially correct decisions wrong gives $D = 0.025$. A net accuracy change would combine these effects.

The edit-effect vector $\mathbf{e}_t(p) = [C_t(p); D_t(p)]$ collects these rates in increasing boundary order and has $d_e = 2(K - 1)$ coordinates. It records effects at both the target boundary and other boundaries. If a denominator is empty, its rate is undefined and excluded from training by a binary mask $\mathbf{m}_t(p)$; it is not recorded as zero. We use only answer records with complete human, before-edit, and after-edit scores (Appendix H.3).

Let $\mathcal{P}_t$ be the candidate set, b_t^* the target boundary, and ϵ_b the maximum tolerated degradation at boundary b . The round-level objective is

$$\begin{aligned} & \underset{p \in \mathcal{P}_t}{\text{maximize}} && C_{t,b_t^*}(p) \\ & \text{subject to} && D_{t,b}(p) \leq \epsilon_b, \quad b \in \mathcal{B}. \end{aligned} \quad (3)$$

This rewards target corrections while limiting new errors separately at each boundary. It does not directly maximize exact-score accuracy (ACC) or quadratic weighted kappa (QWK) (Cohen, 1968).

3.2 Data Use and Candidate Generation

Data requirements. Each example contains an item, an answer, and its human score. RUBRICWORLD requires three disjoint labeled sets:

$\mathcal{D}_{\text{dyn,train}}$ for candidate generation, paired-effect measurement, and world-model training; $\mathcal{D}_{\text{dyn,val}}$ for tuning; and $\mathcal{D}_{\text{cert}}$ for independent acceptance checks. Certification requires a fresh, disjoint subset $\mathcal{D}_{\text{cert}}^{(t)}$ at each round to prevent adaptation to certification answers (Dwork et al., 2015). $\mathcal{D}_{\text{test}}$ is reserved for final evaluation. Section 4.2 describes their construction.

Error groups and candidates. At each boundary, over-scoring means that the grader places an answer above the boundary when its human score does not; under-scoring is the reverse. Thus there are at most $2(K - 1)$ boundary-direction groups. We further separate errors present under the initial rubric from errors introduced by accepted edits, giving at most $4(K - 1)$ groups. An answer can occur in several groups if its score crosses several boundaries. Empty groups are omitted.

The routing rule selects one nonempty group from the training errors, fixing the target boundary b_t^* and correction direction before candidate generation (Appendix G.4). Given this group, the current rubric, and relevant correctly graded training answers, the generator proposes $M = 5$ local edits. Each edit tightens, loosens, or clarifies the chosen score distinction. We remove duplicate rules, contradictions, unrelated criteria, and rules tied to a particular answer’s wording. The remaining candidates form $\mathcal{P}_t$; each retains its target, direction, supporting examples, and proposed text change (Appendix G.5). If no training errors remain, $\mathcal{P}_t = \emptyset$. Appendix C.2 gives an illustrative edit.

3.3 Predicting Edit Effects

Measuring every candidate’s effects requires repeated grader calls. The world model predicts those effects first, so only the selected candidate needs direct evaluation. Its input is the edit p and state $s_t = (R_t, \mathcal{H}_t, \mathbf{v}_t)$. Here, $\mathcal{H}_t$ stores previously measured edit effects, and $\mathbf{v}_t$ summarizes the grader’s current scores and boundary correctness on training answers, together with answer features. These inputs describe what has already been tried and which errors remain. Only R_t is supplied to the grader as its scoring rubric; the history and summary are inputs to the world model.

Text embeddings. A general-purpose text encoder embeds the proposed edit together with its target boundary and correction direction, giving $a_p = \phi(p) \in \mathbb{R}^{d_a}$. The state encoder combines

an embedding of the rubric text with the measured history and training-answer summary to produce $h_t = \psi(s_t) \in \mathbb{R}^{d_h}$. Text embeddings represent what the rules say; paired grading observations teach the predictor how those rules affect the grader. The encoders operate separately from the grader and require no access to its weights, so the same interface supports API-based graders. Optional score-probability features are used only when available; details are in Appendix I.2.

Edit and state interaction. With trainable parameters θ , the predictor learns an edit’s general effect and an adjustment for the current rubric state:

$$\begin{aligned}\eta_\theta(s_t, p) &= W_a a_p + c + U[(A a_p) \odot (B h_t)], \\ \mu_\theta(s_t, p) &= \sigma(\eta_\theta(s_t, p)).\end{aligned}\tag{4}$$

The learned map W_a predicts from the edit alone, with bias c . The learned maps A and B project the edit and state to the same width; element-wise multiplication $\odot$ combines them, and U maps the result to an adjustment for each correction or degradation rate (Rendle, 2010). The sigmoid σ keeps predicted rates in $[0, 1]$. This interaction allows the same edit to receive different predictions under different rubrics. Matrix dimensions are given in Appendix I.3.

Learning from measured edits. Each record $(s, p, \mathbf{e}, \mathbf{m})$ in $\mathcal{T}_t$ contains the state, edit, measured rates, and validity mask. We minimize

$$\mathcal{L}(\theta) = \sum_{(s, p, \mathbf{e}, \mathbf{m}) \in \mathcal{T}_t} \sum_{\substack{1 \leq k \leq d_e \\ m_k = 1}} \ell(\mu_{\theta, k}(s, p), e_k), \tag{5}$$

where ℓ is the scalar regression loss and k indexes a rate. The mask excludes undefined rates. Initial records $\mathcal{T}_0$ come from edits evaluated on training answers before model-guided selection. We fit five independently seeded models and update them as new edits are measured (Appendix I).

3.4 Selecting and Evaluating a Candidate

The five models provide both a mean prediction and their disagreement about each candidate (Lakshminarayanan et al., 2017). For candidate p_t^j and boundary b , let $\mu_{t,j,b}^C$ and $\mu_{t,j,b}^D$ be the mean predicted correction and degradation rates, and $\sigma_{t,j,b}^C$ and $\sigma_{t,j,b}^D$ their sample standard deviations across models. We penalize disagreement to avoid choosing an edit only because of an uncertain favorable

prediction. With penalty weight $\beta \geq 0$,

$$\begin{aligned}\tilde{C}_{t,j} &= \mu_{t,j,b_t^*}^C - \beta \sigma_{t,j,b_t^*}^C, \\ \tilde{D}_{t,j,b} &= \mu_{t,j,b}^D + \beta \sigma_{t,j,b}^D.\end{aligned}\quad (6)$$

These are screening scores, not confidence bounds with a coverage guarantee (Guo et al., 2017). We select the candidate with the largest penalized target correction among those within the predicted degradation limits:

$$\begin{aligned}p_t^* &\in \arg \max_{p_t^j \in \mathcal{P}_t} \tilde{C}_{t,j} \\ \text{subject to } \tilde{D}_{t,j,b} &\leq \epsilon_b, \quad b \in \mathcal{B}.\end{aligned}\quad (7)$$

If none qualifies, the round leaves the rubric and model unchanged. Otherwise, paired grading on $\mathcal{D}_{\text{dyn},\text{train}}$ measures the selected edit’s effect. We add this observation to the training records $\mathcal{T}_t$ and the state history $\mathcal{H}_t$, then update all five models. The training records support fitting across states; the history tells the predictor what has already been tried in the current run. Rejected edits remain in both: their measured effects help the model recognize similar unsuccessful candidates later.

3.5 Checking Whether to Accept the Edit

A favorable prediction is not enough to update the rubric. We evaluate the selected edit on the unused answers in $\mathcal{D}_{\text{cert}}^{(t)}$ and require evidence that it corrects enough target errors while introducing few new ones. We call this check *certification*.

The certification scope $\mathcal{B}_{\text{cert}} \subseteq \mathcal{B}$ lists the boundaries for which the experiment requires this check. It is fixed before candidate outcomes are inspected, so a boundary cannot be dropped because an edit performed poorly there. The target b_t^* must belong to this scope. All boundaries still enter predictive screening, but statistical claims cover only the prespecified scope. The required minimum correction is $\gamma = 0.80$ and the maximum degradation is $\epsilon_b = 0.08$.

Each initially wrong decision either becomes correct or remains wrong; each initially correct decision either becomes wrong or remains correct. Under independent binomial sampling, these counts support exact one-sided confidence bounds (Clopper and Pearson, 1934). We use a lower bound for target correction, because it must be large enough, and an upper bound for degradation, because it must stay small. For per-edit error probability δ , we divide the error budget equally among the one correction bound and the $|\mathcal{B}_{\text{cert}}|$ degradation bounds.

This accounts for checking several requirements on the same edit.

Let $L_{t,b_t^*}^C$ be the target-correction lower bound and $U_{t,b}^D$ the degradation upper bound at boundary b , computed from the corresponding initially wrong or correct decisions. The edit is accepted only if

$$\begin{aligned}L_{t,b_t^*}^C &\geq \gamma, \\ U_{t,b}^D &\leq \epsilon_b, \quad b \in \mathcal{B}_{\text{cert}}.\end{aligned}\quad (8)$$

If a required sample is too small to establish its bound even in the best case, the edit is marked unsupported. Otherwise, failing a bound gives a rejection. Both outcomes retain the rubric. The binomial formulas, sample requirements, and assumptions are in Appendix K. The coverage statement applies to one edit; a joint guarantee across rounds requires additional confidence allocation.

An accepted edit gives $R_{t+1} = R_t \oplus p_t^*$; otherwise $R_{t+1} = R_t$. We then refresh the training-answer summary under the resulting rubric to form the next state. After T rounds, R_T is used for test scoring.

4 Experiments

4.1 Experimental Design

We evaluate final grading quality, optimization cost, the role of the current rubric state, and the contributions of prediction, selection, and certification. We also track grading errors over 100 editing rounds.

Benchmarks. EIR (National) contains five private short-answer items, with item-specific scores from 0 to 3 or 0 to 4 (Chu et al., 2025). ASAP2.0 contains student essays scored from 1 to 6 (The Learning Agency Lab, 2024; Crossley et al., 2025). ASAP-SAS evaluates short answers; the reported Set 1 uses scores from 0 to 3 (Barbara et al., 2012; Sakaguchi et al., 2015). The ordered score scales allow us to examine both exact scores and decisions at each adjacent score boundary. Counts and task-level splits are in Appendix B.

Frozen LLM graders. We use Llama-3.1-8B-Instruct, an 8B-parameter instruction-tuned language model (Grattafiori et al., 2024); QwQ-32B-Preview, a 32B-parameter reasoning language model from Qwen (Qwen Team, 2024); and OpenAI’s API-based language model gpt-5-mini (OpenAI, 2026). Llama and QwQ use the 4-bit activation-aware weight-quantized (AWQ) checkpoints in Table 1 (Lin et al., 2024). Only the scoring rubric is updated.

Method	EIR (National)			ASAP2.0			ASAP-SAS		
	ACC	QWK	F1	ACC	QWK	F1	ACC	QWK	F1
Meta-Llama-3.1-8B-Instruct-AWQ-INT4									
Zero-shot (Brown et al., 2020)	55.42 $\pm$ 0.54	54.10 $\pm$ 0.46	54.85 $\pm$ 1.92	12.50 $\pm$ 2.15	5.75 $\pm$ 2.13	4.55 $\pm$ 0.70	1.99 $\pm$ 0.64	1.44 $\pm$ 0.87	1.20 $\pm$ 2.08
Few-shot (Brown et al., 2020)	57.12 $\pm$ 2.00	52.80 $\pm$ 0.56	55.15 $\pm$ 0.20	9.40 $\pm$ 0.54	4.78 $\pm$ 0.79	4.74 $\pm$ 1.45	19.92 $\pm$ 1.39	0.01 $\pm$ 2.25	7.75 $\pm$ 2.01
APO (Pryzant et al., 2023)	61.35 $\pm$ 1.17	60.15 $\pm$ 1.97	59.90 $\pm$ 1.17	15.45 $\pm$ 1.68	8.92 $\pm$ 1.27	7.98 $\pm$ 0.79	21.51 $\pm$ 0.75	8.90 $\pm$ 1.56	14.50 $\pm$ 0.61
GradeOpt (Chu et al., 2025)	54.20 $\pm$ 0.83	56.45 $\pm$ 0.18	53.10 $\pm$ 1.99	10.48 $\pm$ 0.68	7.37 $\pm$ 2.11	5.44 $\pm$ 1.53	33.07 $\pm$ 1.76	9.15 $\pm$ 1.53	21.09 $\pm$ 0.43
GEPA (Agrawal et al., 2026)	58.90 $\pm$ 2.09	57.20 $\pm$ 1.96	57.45 $\pm$ 0.31	13.10 $\pm$ 1.31	6.80 $\pm$ 1.62	5.90 $\pm$ 0.67	16.40 $\pm$ 0.96	4.50 $\pm$ 1.37	8.20 $\pm$ 1.62
RUBRICWORLD	64.88 $\pm$ 1.10	63.12 $\pm$ 0.45	62.45 $\pm$ 1.89	34.94 $\pm$ 0.52	32.94 $\pm$ 1.90	30.77 $\pm$ 0.30	29.50 $\pm$ 1.65	9.40 $\pm$ 1.22	20.15 $\pm$ 0.36
Qwen/QwQ-32B-Preview-AWQ-INT4									
Zero-shot (Brown et al., 2020)	68.15 $\pm$ 1.46	67.40 $\pm$ 2.20	67.90 $\pm$ 0.91	22.15 $\pm$ 0.72	18.40 $\pm$ 0.77	15.60 $\pm$ 1.29	14.20 $\pm$ 1.28	11.05 $\pm$ 0.43	10.80 $\pm$ 1.03
Few-shot (Brown et al., 2020)	66.40 $\pm$ 2.06	68.95 $\pm$ 2.10	65.20 $\pm$ 0.37	18.30 $\pm$ 2.14	12.60 $\pm$ 0.62	11.90 $\pm$ 1.64	25.40 $\pm$ 1.97	4.25 $\pm$ 0.24	15.60 $\pm$ 1.47
APO (Pryzant et al., 2023)	75.30 $\pm$ 1.10	71.85 $\pm$ 0.93	74.10 $\pm$ 1.26	31.40 $\pm$ 2.07	28.15 $\pm$ 1.63	25.90 $\pm$ 2.13	26.80 $\pm$ 1.65	14.30 $\pm$ 1.31	21.15 $\pm$ 2.02
GradeOpt (Chu et al., 2025)	70.12 $\pm$ 1.13	69.50 $\pm$ 2.05	69.95 $\pm$ 1.00	28.55 $\pm$ 2.11	22.10 $\pm$ 0.34	21.45 $\pm$ 1.87	33.45 $\pm$ 1.06	16.80 $\pm$ 0.35	28.40 $\pm$ 0.61
GEPA (Agrawal et al., 2026)	72.05 $\pm$ 1.54	69.80 $\pm$ 2.15	71.15 $\pm$ 2.20	26.80 $\pm$ 1.24	23.40 $\pm$ 0.20	20.25 $\pm$ 2.31	22.10 $\pm$ 1.92	12.50 $\pm$ 1.38	17.30 $\pm$ 0.68
RUBRICWORLD	73.85 $\pm$ 0.18	74.12 $\pm$ 1.96	72.60 $\pm$ 1.97	38.90 $\pm$ 1.35	37.25 $\pm$ 1.39	35.10 $\pm$ 0.76	34.12 $\pm$ 0.24	22.45 $\pm$ 0.62	31.15 $\pm$ 0.37
OpenAI gpt-5-mini									
Zero-shot (Brown et al., 2020)	73.42 $\pm$ 1.87	72.85 $\pm$ 0.50	73.11 $\pm$ 0.78	18.20 $\pm$ 1.18	13.55 $\pm$ 2.31	12.10 $\pm$ 0.30	7.80 $\pm$ 1.97	5.15 $\pm$ 0.97	5.45 $\pm$ 1.72
Few-shot (Brown et al., 2020)	74.88 $\pm$ 0.60	74.12 $\pm$ 1.88	74.35 $\pm$ 0.97	14.35 $\pm$ 0.39	10.20 $\pm$ 0.34	10.80 $\pm$ 2.07	23.10 $\pm$ 2.12	2.45 $\pm$ 1.06	13.90 $\pm$ 0.47
APO (Pryzant et al., 2023)	77.25 $\pm$ 1.66	76.90 $\pm$ 2.05	76.58 $\pm$ 0.30	24.10 $\pm$ 2.32	20.55 $\pm$ 2.23	18.40 $\pm$ 1.23	27.55 $\pm$ 1.53	14.20 $\pm$ 2.31	19.80 $\pm$ 1.76
GradeOpt (Chu et al., 2025)	82.94 $\pm$ 0.49	83.12 $\pm$ 2.15	82.85 $\pm$ 0.27	21.80 $\pm$ 1.22	15.40 $\pm$ 1.85	14.95 $\pm$ 1.15	35.20 $\pm$ 1.80	14.85 $\pm$ 1.63	25.60 $\pm$ 1.84
GEPA (Agrawal et al., 2026)	75.60 $\pm$ 1.89	74.90 $\pm$ 0.85	75.10 $\pm$ 1.48	21.30 $\pm$ 0.82	17.65 $\pm$ 1.28	15.20 $\pm$ 1.79	21.40 $\pm$ 0.25	10.30 $\pm$ 0.35	14.15 $\pm$ 0.63
RUBRICWORLD	81.15 $\pm$ 0.97	81.38 $\pm$ 0.36	81.62 $\pm$ 2.33	46.70 $\pm$ 1.57	45.10 $\pm$ 1.61	43.25 $\pm$ 0.88	39.15 $\pm$ 0.85	19.85 $\pm$ 1.63	29.70 $\pm$ 2.27

Table 1: **Final grading scores across three benchmarks and frozen LLM graders.** Three-seed mean $\pm$ SD; ACC/F1 in %, QWK on a 0–100 scale. Blue/red denotes changes from zero-shot; bold marks the best displayed score.



Figure 2: **Prediction, selection, and certification on ASAP2.0 (gpt-5-mini).** (a) Held-out-state MAE (pp); (b) selected-edit correction and degradation (%); (c) test QWK (0–100) and degradation (%). Three-seed mean $\pm$ SD; variants in Appendix N.

Variant	QWK@5 $\uparrow$	ACC@5 $\uparrow$	Tokens (M) $\downarrow$	
			5 rounds	20 rounds
Full RubricWorld	38.43	40.18	6.84	8.32
w/o State	33.16	35.42	6.41	8.68
w/o Learned Model	24.85	27.31	6.72	8.15

Table 2: **Edit-effect model ablation on ASAP2.0.** Average across three frozen graders with shared generator and certification.

4.2 Methods, Protocol, and Measures

Comparison methods. Zero-shot uses the original rubric, and few-shot adds labeled examples without

changing it (Brown et al., 2020). APO edits instructions using language-based feedback on mistakes (Pryzant et al., 2023); GradeOpt refines grading guidelines from misgraded answers (Chu et al., 2025); and GEPA uses evaluated traces, reflection, and a Pareto-based prompt search (Agrawal et al., 2026). All methods begin with the same rubric and frozen grader.

Editing and evaluation. The main comparison uses five refinement rounds and three random seeds. RUBRICWORLD considers five candidate edits per round. Before editing, we reserve test an-

swers and then the certification pool, remove cross-pool duplicates, and sample dynamics-validation then dynamics-training answers, stratified by human score with fixed seeds. Certification uses disjoint round-specific subsets (Appendix B; Appendix G.1). Predictor variants use the same edit embeddings. Baselines use their native search procedures within the declared refinement protocol; implementation details are in Appendix D.

What we measure. We report exact-score accuracy (ACC), quadratic weighted kappa (QWK), and macro-F1 for final grading quality (Cohen, 1968; Attali and Burstein, 2006). For edit-effect prediction, we report errors in the predicted correction and degradation rates (MAE_C and MAE_D) in percentage points (pp); comparisons use unseen rubric states rather than random transition splits. For selection, we measure the chosen edit’s actual target correction, mean degradation, and empirical feasibility rate (EFR). For acceptance rules, we compare final QWK and observed degradation under different acceptance rules. Metric definitions appear in Appendix M.

Cost and long-horizon protocol. Optimization cost counts LLM tokens spent on proposing edits, collecting warm-start and online observations, paired scoring, and independent commit checks, excluding the shared final evaluation. The cost sweep reports three-seed mean QWK versus optimization tokens for gpt-5-mini.

The 20-round continuation retains the five-round settings. The 100-round plots use matched ASAP2.0 / gpt-5-mini runs, tracking validation ACC and residual degradation separately.

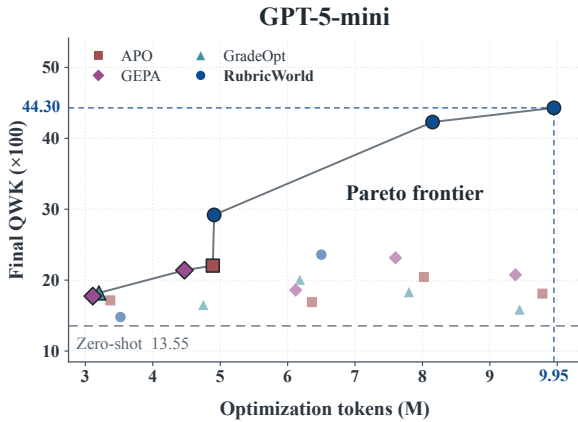


Figure 3: **Quality-cost trade-off on ASAP2.0 (gpt-5-mini).** Three-seed means including warm-start tokens; outlined points are Pareto-optimal.

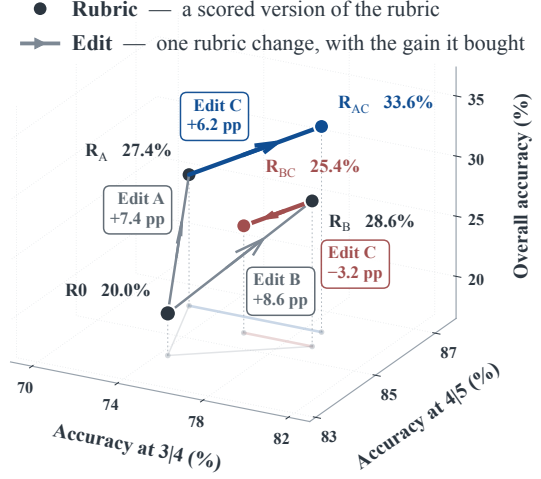


Figure 4: **Illustrative rubric-editing trajectories (ASAP2.0 scale; gpt-5-mini setting).** The same edit C is applied after A or B.

5 Results and Analysis

5.1 Overall Grading Quality

RUBRICWORLD achieves the highest score in 20 of the 27 dataset-grader-metric comparisons in Table 1. Its strongest gains occur on ASAP2.0, where all nine comparisons favor RUBRICWORLD. QWK reaches 32.94, 37.25, and 45.10 with Llama-3.1-8B, QwQ-32B, and gpt-5-mini, respectively; the strongest baselines reach 8.92, 28.15, and 20.55.

The short-answer results are less uniform. On EIR with QwQ-32B, APO has higher ACC (75.30% versus 73.85%) but lower QWK (71.85 versus 74.12), which penalizes larger scoring errors. GradeOpt exceeds RUBRICWORLD in ACC and QWK on EIR with gpt-5-mini, and in ACC on ASAP-SAS with Llama-3.1-8B. The gains therefore vary by dataset and metric.

5.2 Grading Quality Versus Token Cost

Figure 3 compares final QWK against optimization tokens on ASAP2.0 with gpt-5-mini. RUBRICWORLD reaches 44.30 QWK near 10M tokens, with smaller differences among methods at lower budgets. These token-constrained checkpoints differ from the fixed five-round endpoint in Table 1 (45.10 QWK).

Fewer candidate evaluations need not mean lower total token cost (Table 2). After twenty rounds, the full model uses 8.32M tokens, compared with 8.68M without state and 8.15M with direct evaluation. These totals include warm-start collection and do not identify which calls account

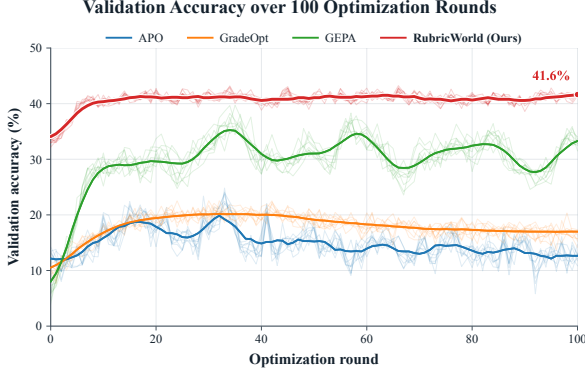


Figure 5: 100-round validation ACC.

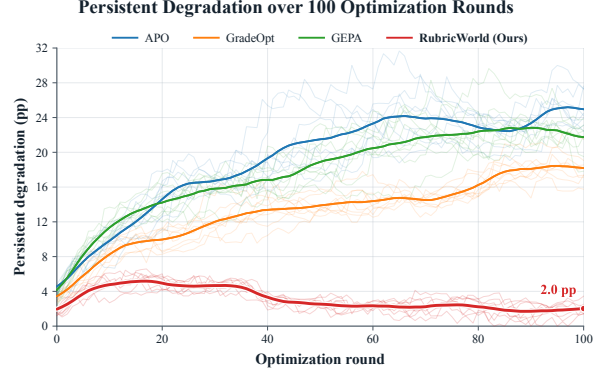


Figure 6: 100-round persistent degradation (pp).

for the differences. The table’s quality measurements cover five rounds only.

5.3 Why the Current Rubric Matters

Figure 4 illustrates how the same edit C can have opposite effects after edits A and B. Although the starting accuracies are close (27.4% and 28.6%), C raises the first to 33.6% and lowers the second to 25.4%. It corrects 41 answers and harms 10 on the A branch, compared with 7 corrections and 23 harms on the B branch. At boundary 3|4, correction/degradation rates are 29.9%/2.8% and 6.7%/5.8%, respectively. This example motivates conditioning on the current rubric. It does not establish how common the effect is or how well a model predicts it; the axes show grading accuracy, not learned representations. We assess predictive value separately on unseen rubric states.

5.4 Prediction, Selection, and Acceptance

State conditioning improves prediction and candidate selection (Figure 2a,b). Compared with edit-only prediction, the full model lowers MAE_C from 39.1 to 24.6 pp and MAE_D from 34.5 to 19.8 pp; additive history ties on MAE_D . The selected edits increase target correction from 8.5% to 16.2% and reduce degradation from 8.8% to 2.5%.

The end-to-end ablation averages over the three graders (Table 2). Five-round ACC/QWK is 40.18%/38.43 for the full model, 35.42%/33.16 without state, and 27.31%/24.85 with direct candidate evaluation and no learned predictor (Appendix N). These matched-seed means do not establish statistical significance.

The acceptance rule trades some QWK for lower degradation (Figure 2c). QWK/degradation is 46.0/7.8% for an empirical gate, 45.1/3.2% for certification, and 41.2/14.2% for prediction-only ac-

ceptance. Thus certification reduces degradation by 4.6 pp at a cost of 0.9 QWK relative to the empirical gate. These are test measurements, not certification bounds.

5.5 Grading After 100 Refinement Rounds

Figures 5 and 6 follow the same three ASAP2.0 runs with gpt-5-mini. RUBRICWORLD stays near 41% validation ACC while persistent degradation falls toward two percentage points. An earlier regression leaves this measure only when a later edit corrects that boundary decision. Accuracy can therefore recover while some earlier damage remains. The two plots distinguish final grading quality from unrepaired errors accumulated during editing; they do not imply a joint finite-sample guarantee across rounds.

6 Conclusion

RUBRICWORLD learns corrections and degradations from paired grading results to predict an edit’s effect under the current rubric. Predictions guide candidate selection; held-out evidence determines acceptance. Rejected edits also contribute to learning. Across three benchmarks and three frozen graders, the largest gains occur on ASAP2.0. Ablations support the use of state information and learned effects, while longer runs track errors left by earlier edits.

Limitations(does not count toward the main-text page limit)

RUBRICWORLD relies on initial rubrics and human reference scores; high reference agreement does not inherently guarantee educational validity when human targets contain noise (Messick, 1995). Empirical certification requires enough examples at checked boundaries; rare score categories and

small round-specific samples may prevent otherwise useful edits from being certified. The edit-effect model may not generalize to unfamiliar criteria or tasks. Our evaluation covers educational grading, not open-domain judging, safety evaluation, or temporal streaming responses (Hashemi et al., 2024).

Ethical Considerations(does not count toward the main-text page limit)

RUBRICWORLD supports research on educational scoring; passing its audit does not establish suitability for high-stakes grading. Incorrect scores can affect students’ opportunities, so consequential or disputed decisions should receive human review, with a way to contest errors. Agreement with reference scores does not establish fairness: references may contain bias, and average metrics can hide unequal errors across groups. Use in practice would require checking these differences in the intended population. Student and teacher responses may also contain identifying information. Sharing them, including with external model services, should respect access conditions and minimize disclosure of personal information. These obligations apply to both the private EIR (National) dataset and the public ASAP2.0 and ASAP-SAS benchmarks (Appendix B.4).

References

- Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziemis, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alex Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. 2026. [GEPA: Reflective prompt evolution can outperform reinforcement learning](#). In *International Conference on Learning Representations*.
- Yigal Attali and Jill Burstein. 2006. [Automated essay scoring with e-rater V.2](#). *The Journal of Technology, Learning and Assessment*, 4(3).
- Barbara, Ben Hamner, Jaison Morgan, lynnvande, and Mark Shermis. 2012. [The Hewlett Foundation: Short answer scoring](#). Kaggle Competition.
- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. [Language models are few-shot learners](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.
- Steven Burrows, Iryna Gurevych, and Benno Stein. 2015. [The eras and trends of automatic short answer grading](#). *International Journal of Artificial Intelligence in Education*, 25(1):60–117.
- Wenzhi Cao, Vahid Mirjalili, and Sebastian Raschka. 2020. [Rank consistent ordinal regression for neural networks with application to age estimation](#). *Pattern Recognition Letters*, 140:325–331.
- Yucheng Chu, Hang Li, Kaiqi Yang, Yasemin Copur-Gencurk, Joseph Krajcik, Namsoo Shin, and Jiliang Tang. 2026. [Confusion-aware rubric optimization for LLM-based automated grading](#). In *Proceedings of the 19th International Conference on Educational Data Mining*, pages 67–78, Seoul, Republic of Korea. International Educational Data Mining Society.
- Yucheng Chu, Hang Li, Kaiqi Yang, Harry Shomer, Yasemin Copur-Gencurk, Leonora Kaldaras, Kevin Haudek, Joseph Krajcik, Namsoo Shin, Hui Liu, and Jiliang Tang. 2025. [A LLM-powered automatic grading framework with human-level guidelines optimization](#). In *Proceedings of the 18th International Conference on Educational Data Mining*, pages 31–41, Palermo, Italy. International Educational Data Mining Society.
- C. J. Clopper and E. S. Pearson. 1934. [The use of confidence or fiducial limits illustrated in the case of the binomial](#). *Biometrika*, 26(4):404–413.
- Jacob Cohen. 1968. [Weighted kappa: Nominal scale agreement with provision for scaled disagreement or partial credit](#). *Psychological Bulletin*, 70(4):213–220.
- Scott A. Crossley, Perpetual Baffour, L. Burleigh, and Jules King. 2025. [A large-scale corpus for assessing source-based writing quality: ASAP 2.0](#). *Assessing Writing*, 65:100954.
- Scott A. Crossley, Yu Tian, Perpetual Baffour, Alex Franklin, Meg Benner, and Ulrich Boser. 2024. [A large-scale corpus for assessing written argumentation: PERSUADE 2.0](#). *Assessing Writing*, 61:100865.
- Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. 2015. [The reusable holdout: Preserving validity in adaptive data analysis](#). *Science*, 349(6248):636–638.
- Diana Galvan-Sosa, Gabrielle Gaudeau, Pride Kavumba, Yunmeng Li, Hongyi Gu, Zheng Yuan, Keisuke Sakaguchi, and Paula Buttery. 2025. [Rubrik’s cube: Testing a new rubric for evaluating explanations on the CUBE dataset](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23800–23839, Vienna, Austria. Association for Computational Linguistics.

- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 540 others. 2024. [The Llama 3 herd of models](#). *arXiv preprint arXiv:2407.21783*.
- Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. [On calibration of modern neural networks](#). In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.
- Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. 2025. [Mastering diverse control tasks through world models](#). *Nature*, 640(8059):647–653.
- Nicklas Hansen, Hao Su, and Xiaolong Wang. 2024. [TD-MPC2: Scalable, robust world models for continuous control](#). In *International Conference on Learning Representations*.
- Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. 2024. [LLM-Rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13806–13834, Bangkok, Thailand. Association for Computational Linguistics.
- Owen Henkel, Libby Hills, Adam Boxer, Bill Roberts, and Zach Levonian. 2024. [Can large language models make the grade? an empirical study evaluating LLMs ability to mark short answer questions in K-12 education](#). In *Proceedings of the Eleventh ACM Conference on Learning @ Scale*, pages 300–304. Association for Computing Machinery.
- Owen Henkel, Kirk Vanacore, and Bill Roberts. 2025. [When humans can’t agree, neither can machines: The promise and pitfalls of LLMs for formative literacy assessment](#). In *Proceedings of the Artificial Intelligence in Measurement and Education Conference (AIME-Con): Coordinated Session Papers*, pages 69–78, Wyndham Grand Pittsburgh, Downtown, Pittsburgh, Pennsylvania, United States. National Council on Measurement in Education (NCME).
- Sture Holm. 1979. [A simple sequentially rejective multiple test procedure](#). *Scandinavian Journal of Statistics*, 6(2):65–70.
- Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. 2024a. [Prometheus: Inducing fine-grained evaluation capability in language models](#). In *International Conference on Learning Representations*.
- Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024b. [Prometheus 2: An open source language model specialized in evaluating other language models](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4334–4353, Miami, Florida, USA. Association for Computational Linguistics.
- Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. [Simple and scalable predictive uncertainty estimation using deep ensembles](#). In *Advances in Neural Information Processing Systems*, volume 30, pages 6403–6414. Curran Associates, Inc.
- Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. [AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration](#). In *Proceedings of Machine Learning and Systems*, volume 6, pages 87–100.
- Yang Liu, Dan Iter, Yichong Xu, Shuhang Wang, Ruochen Xu, and Chenguang Zhu. 2023. [G-Eval: NLG evaluation using GPT-4 with better human alignment](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.
- Watheq Ahmad Mansour, Salam Albatarni, Sohaila Eltanbouly, and Tamer Elsayed. 2024. [Can large language models automatically score proficiency of written essays?](#) In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2777–2786, Torino, Italia. ELRA and ICCL.
- Peter McCullagh. 1980. [Regression models for ordinal data](#). *Journal of the Royal Statistical Society: Series B (Methodological)*, 42(2):109–127.
- Samuel Messick. 1995. [Validity of psychological assessment: Validation of inferences from persons’ responses and performances as scientific inquiry into score meaning](#). *American Psychologist*, 50(9):741–749.
- OpenAI. 2026. [GPT-5 mini model](#). OpenAI API Documentation. Undated official OpenAI API model documentation; accessed 2026-10-08. Year denotes access year, not publication date.
- Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. 2024. [Optimizing instructions and demonstrations for multi-stage language model programs](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9340–9366, Miami, Florida, USA. Association for Computational Linguistics.
- Ellis B. Page. 1966. [The imminence of... grading essays by computer](#). *Phi Delta Kappan*, 47(5):238–243.

- Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. 2023. [Automatic prompt optimization with “gradient descent” and beam search](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7957–7968, Singapore. Association for Computational Linguistics.
- Qwen Team. 2024. [QwQ-32B-Preview](#). Hugging Face Model Card.
- Steffen Rendle. 2010. [Factorization machines](#). In *2010 IEEE International Conference on Data Mining*, pages 995–1000.
- Keisuke Sakaguchi, Michael Heilman, and Nitin Madnani. 2015. [Effective feature integration for automated short answer scoring](#). In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1049–1054, Denver, Colorado. Association for Computational Linguistics.
- Subhash Bangalore Satheesha, Nirvik Pande, Deepthi Duddempudi, and Bharath Dandala. 2026. [RLMOpt: Adaptive prompt optimization via recursive language models](#). *Preprint*, arXiv:2608.10471.
- Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. [Quantifying language models’ sensitivity to spurious features in prompt design or: How I learned to start worrying about prompt formatting](#). In *International Conference on Learning Representations*.
- Mark D. Shermis and Ben Hamner. 2013. [Contrasting state-of-the-art automated scoring of essays](#). In Mark D. Shermis and Jill Burstein, editors, *Handbook of Automated Essay Evaluation: Current Applications and New Directions*, pages 313–346. Routledge.
- Kaveh Taghipour and Hwee Tou Ng. 2016. [A neural approach to automated essay scoring](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1882–1891, Austin, Texas. Association for Computational Linguistics.
- The Learning Agency Lab. 2024. [Learning Agency Lab - automated essay scoring 2.0](#). Kaggle Competition.
- Xinyuan Wang, Chenxi Li, Zhen Wang, Fan Bai, Hao-tian Luo, Jiayou Zhang, Nebojsa Jojic, Eric P. Xing, and Zhiting Hu. 2024. [PromptAgent: Strategic planning with language models enables expert-level prompt optimization](#). In *International Conference on Learning Representations*.
- Yifan Xu, Yixuan Li, Xinzhuo Li, Yixin Gu, Yifan Shen, Lijun Yu, and Haohan Wang. 2026. [STEVE: Stabilizing textual gradient-based prompt optimization via error-driven refinement and regularized verification](#). *Preprint*, arXiv:2609.23716.
- Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2024. [Large language models as optimizers](#). In *International Conference on Learning Representations*.
- Jin Peng Zhou, Séb Arnold, Nan Ding, Kilian Q Weinberger, Nan Hua, and Fei Sha. 2025. [Graders should cheat: Privileged information enables expert-level automated evaluations](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 16572–16590, Suzhou, China. Association for Computational Linguistics.

A AI Assistance Disclosure

ChatGPT assisted with language polishing (including the reduction of redundant and colloquial terms) and the layout design of figures and tables. The authors bear full responsibility for the scientific content, experimental results, references, and conclusions of the manuscript.

B Dataset Counts and Data Use

Our essay tasks assign each response one ordered score, as in earlier automated essay scoring comparisons (Shermis and Hamner, 2013). Short-answer tasks use question-specific scoring criteria (Burrows et al., 2015). The configuration specifies stratification by reference score when validation responses are selected from the training data. Unless otherwise stated, optimization uses only the training split and validation subsets; the final test set is held out until the final evaluation.

B.1 EIR (National)

EIR (National) contains five private short-answer items, each with an item-specific score scale. To document the experimental split without exposing restricted response content, Table 3 reports aggregate counts only.

Item	n	Scale	Train	Val/Test
U41Q1	199	0–3	159	20 / 20
U41Q2	230	0–4	184	23 / 23
U42Q1	248	0–4	198	25 / 25
U42Q2	262	0–3	209	26 / 27
U43Q1	434	0–4	347	43 / 44
Total	1,373	–	1,097	137 / 139

Table 3: EIR (National) aggregate split statistics. Items use fixed 80/10/10 partitions, stratified by reference score where class counts permit.

No response appears in more than one partition, and the test partition is reserved for final evaluation, consistent with the protocol in Section 4.

B.2 ASAP2.0

ASAP2.0 uses a 1–6 holistic scoring scale (The Learning Agency Lab, 2024). Its essays come from the PERSUADE 2.0 corpus (Crossley et al., 2024). The task requires evaluating student essays according to argument quality, evidence use, organization, language control, and mechanics. Table 4 reports the split statistics used in our experiments.

Split	#Samples	Score distribution
Train	14,711	1058 / 4023 / 5326 / 3350 / 827 / 127
Test	2,596	194 / 700 / 954 / 576 / 143 / 29

Table 4: ASAP2.0 split statistics. Score distributions are ordered from score 1 to score 6.

ASAP2.0 collection allocation. Recorded edit-effect collection uses split seed 0. Of the 14,711 training answers, 1,500 are reserved for $\mathcal{D}_{\text{cert}}$ (10.2%). We then sample 150 answers for $\mathcal{D}_{\text{dyn, val}}$ from the remaining 13,211, followed by the dynamics-training sample from the remaining 13,061. The 2,596 test answers remain unchanged and are used only for final evaluation. Table 5 lists these allocations.

Data role	Answers
$\mathcal{D}_{\text{dyn, train}}$	60 / 200 / 400
$\mathcal{D}_{\text{dyn, val}}$	150
$\mathcal{D}_{\text{cert}}$	1,500
$\mathcal{D}_{\text{test}}$	2,596

Table 5: ASAP2.0 data allocation for edit-effect collection. The three training counts are alternative collection budgets; the certification count is the total pool size, not a per-round eligible sample count.

The 60-answer sample is used for the initial diagnostic, the 200-answer sample for Pilot A and prompt-order checks, and the 400-answer sample for the audit and main transition collection. They are separate stratified samples from the same pool, not three additional data partitions. Their identifiers are recorded separately. The 150-answer validation sample is 1.1% of its source pool; the three training budgets are 0.5%, 1.5%, and 3.0% of theirs.

B.3 ASAP-SAS

ASAP-SAS consists of item-specific short-answer scoring tasks (Barbara et al., 2012). We use the public train/test split and carve out a validation set from the training data. The score scale is item-specific; for the reported Set 1 experiments, the score range is 0–3. We use Set 1 because its four score levels are sufficiently populated to support the three boundary-level diagnostics; we do not claim that this result transfers to the remaining items. Table 6 lists the training, validation, and test counts.

Short-answer collection budgets. The dynamics-training and dynamics-validation

Split	#Samples	Score distribution
Train	1,170	266 / 301 / 366 / 237
Valid	252	Stratified from training data
Test	251	57 / 64 / 79 / 51

Table 6: ASAP-SAS Set 1 split statistics. Score distributions are ordered from score 0 to score 3.

budgets are 400 and 150 answers for ASAP-SAS, and 120 and 30 answers per EIR item. These are internal collection subsets, distinct from the benchmark train/validation/test counts above.

B.4 Data Access: Private and Public Datasets

EIR (National) is a private dataset and is not publicly available. It contains written responses used to assess teachers’ subject knowledge and knowledge of teaching (Chu et al., 2025). The other two datasets, ASAP2.0 and ASAP-SAS, are public benchmarks. Their availability does not imply unrestricted reuse or redistribution; users should follow the original providers’ access and usage terms.

C Prompt and Patch Format

The scoring prompt is the text sent to the frozen grader. A patch proposes an edit to that text and records why the change is needed. The formats below show the initial essay rubric, the information supplied with a candidate edit, and diagnostic paired-effect records; independent certification makes the final commit decision (Appendix K).

C.1 Initial Scoring Prompt

The initial prompt tells the grader what each essay score means. It is the starting specification; optimization adds or revises scoring instructions without changing the grader’s parameters. This differs from approaches that train rubric-following evaluators or construct explanatory evaluation rubrics (Kim et al., 2024b; Galvan-Sosa et al., 2025).

Initial Holistic Scoring Prompt

You are grading a student essay response on a 1-6 holistic scale.

Rubric dimensions:

- Point of view and critical thinking
- Use of examples, reasons, and evidence
- Organization, focus, coherence, and progression
- Language quality, including vocabulary and sentence structure

- Grammar, usage, and mechanics

Score 6:

Clear and consistent mastery. The essay presents a strong position, well-developed reasoning, relevant evidence, coherent organization, and skillful language control.

Score 5:

Reasonably consistent mastery with occasional lapses.

Score 4:

Adequate mastery with lapses in quality.

Score 3:

Developing mastery. The essay shows partial control of the task but has limited support, weak explanation, inconsistent organization, or accumulating language problems.

Score 2:

Little mastery. The essay may contain a vague position or minimal support, with weak organization and frequent language problems.

Score 1:

Very little or no mastery. The essay is highly limited, disorganized, off-topic, or difficult to understand.

C.2 Boundary-Targeted Patch Schema

Each candidate contains the proposed scoring text and the evidence for it. In the schema, patch_text is the edit applied to the specification; the other fields identify its target and explain the intended effect.

Boundary-Targeted Patch Schema

```
{
  "target_boundary": "<score
    transition b|b+1>",
  "error_direction": "<over-scoring |
    under-scoring>",
  "criterion": "<affected rubric
    criterion>",
  "failure_cases": "<misgraded
    examples requiring correction>",
  "anchor_cases": "<correctly scored
    training examples>",
  "patch_text": "<proposed edit to the
    scoring instructions>",
  "expected_effect": "<intended
    boundary-level correction>",
  "rationale": "<why this update
    addresses the observed failure>"
}
```

A patch for 3 | 4 should clarify when a re-

sponse deserves at least score 4. Failure cases illustrate mistakes to repair, while training anchors are correctly scored examples supplied to the generator. The intended scope guides candidate filtering, while paired-effect auditing measures the resulting grading changes.

Illustrative edit. Suppose the selected group contains answers over-scored at 3 | 4: the grader gives at least 4, while their human scores are at most 3. If these answers list evidence without explaining how it supports the position, a candidate could add:

To award a score of 4 or higher for evidence use, require an explanation of how the examples or reasons support the stated position. Listing examples alone does not meet this requirement.

This example illustrates the format; it is not an edit recovered from an experimental run. Its target is 3 | 4, its direction is over-scoring, and its criterion is evidence use. Correctly graded training examples help the generator preserve answers that deserve 4. The edit may still lower some of those answers, so paired grading measures both correction and degradation at every boundary before acceptance is considered.

C.3 Patch Generation Constraints

The generation instructions ask for an edit that addresses the observed failure, retains the original criteria, and targets the selected score distinction. These requests do not by themselves guarantee compliance. For example, an edit about explaining evidence should not add a new requirement unrelated to the rubric.

We represent the candidate as $p = (b, d_{\text{dir}}, \mathcal{F}^{\text{train}}, \mathcal{A}^{\text{train}}, \Delta R)$: b is its intended boundary, d_{dir} is over- or under-scoring, $\mathcal{F}^{\text{train}}$ contains training failures, $\mathcal{A}^{\text{train}}$ contains training anchors, and ΔR is the proposed text edit. The target metadata identifies intended corrections; paired grading measures the effects of the proposed edit.

C.4 Audit Metadata

A selected candidate is associated with diagnostic paired-effect measurements and a separate held-out certification outcome:

Patch Audit Record

```
{
  "target_correction": "<paired target
    -boundary correction>",
```

Method	Text supplied or changed	Selection basis
Zero-shot	Original rubric	No update
Few-shot	Scored examples appended to prompt	Validation agreement
APO	Rewritten scoring instructions	Aggregate validation score
GradeOpt	Guidelines and examples derived from errors	Validation improvement
GEPA	Prompt variants generated from reflection	Candidate-pool selection
RUBRICWORLD	Edit to a chosen score distinction	Repair and per-boundary breakage

Table 7: What each comparison method changes and evaluates.

```
"boundary_degradation": "<paired
  degradation by boundary>",
"repair_support": "<eligible paired
  counts by boundary>",
"degradation_support": "<eligible
  paired counts by boundary>",
"certification_repair_support": "<
  held-out target-error count>",
"certification_degradation_support":
  "<held-out initially-correct
  counts by boundary>",
"certification_status": "<certified
  | rejected | unsupported>",
"commit_status": "<accepted |
  retained>"
}
```

The paired edit-effect record contains correction and degradation rates at the observed score boundaries. These measurements update the learned predictor irrespective of the subsequent commit decision. The final acceptance rule instead applies exact one-sided confidence bounds on the unused certification split: a candidate is committed only when the target correction lower bound and every degradation upper bound in the prespecified certification scope meet their respective thresholds (Section 3.5; Appendix K.3). Residual boundary regressions are tracked separately for long-horizon analysis (Appendix H.6).

D Baseline Implementation Details

The methods differ in what they add to or revise in the scoring specification, and how they choose the resulting text. The grading LLM is frozen for every method.

APO uses feedback on grading errors to propose revised instructions, then selects by aggregate val-

Parameter	Value
Optimization rounds	5
Main-table repetitions	3 seeds; mean $\pm$ std
Long-horizon analysis rounds	100
Candidate patches per round	5
Validation sampling	By reference score
Data used to find errors	Training partition
Minimum target repair γ	0.80
Protected regression tolerance ϵ_b	0.08
Intended patch scope	One score boundary

Table 8: Reported optimization settings for RUBRIC-WORLD.

idation performance. GradeOpt turns misgraded responses into revised guidelines and examples. GEPA uses reflection on evaluation traces to generate prompt variants and maintains a pool of candidates. RUBRICWORLD evaluates an edit by the intended failures it repairs and the previously correct decisions it breaks at each score boundary. Their search strategies follow the original method descriptions, while the frozen grader is shared across comparisons.

E Hyperparameter Settings

Table 8 lists the reported settings. A round is one cycle of target selection, candidate generation, and acceptance checking. A patch count refers to proposed text edits, including rejected candidates.

F Audit Threshold Selection

The two thresholds constrain different one-sided confidence bounds. $\gamma = 0.80$ is the minimum required lower bound for target-boundary correction; $\epsilon_b = 0.08$ is the maximum allowed upper bound for degradation on each boundary in the prespecified certification scope. For illustration, a sample with 20 target failures and 100 initially correct decisions would have empirical point estimates of 80% correction and 8% degradation after 16 repairs and eight regressions. Certification, however, evaluates the corresponding confidence limits rather than these point estimates (Appendix K.2).

The thresholds are fixed before final test evaluation and shared across datasets and graders; improvements at one boundary do not offset a bound violation at another certified boundary.

G Data Isolation and Candidate Construction

This section provides the data-isolation protocol and candidate construction procedure used by the optimization loop in Section 3.

G.1 Data Isolation

Each data point is a student or teacher response with its human score for a fixed scoring task. We reserve the final test partition and assign the remaining labeled answers to three separate optimization roles, yielding four disjoint answer sets:

$$\mathcal{D}_{\text{dyn,train}}, \quad \mathcal{D}_{\text{dyn,val}}, \quad \mathcal{D}_{\text{cert}}, \quad \mathcal{D}_{\text{test}}.$$

Training responses supply candidate construction, paired-effect measurements, and model fitting. The dynamics-validation set is reserved for predictor tuning; the certification set governs commit decisions and the final test set is accessed only for final evaluation.

Identifiers and sampling order. An answer is identified by a SHA-1 hash of its item and response text, $\text{sha1}(\text{item}||\text{text})$, rather than its row number. We reserve the supplied test split, then the certification pool. Cross-pool duplicates are removed with priority given to test, then certification, then dynamics data. From the remaining training pool, we draw dynamics-validation answers first and dynamics-training answers second, both stratified by human score, using seeds *seed* and *seed*+1, respectively.

The split loader raises an error if an identifier remains in more than one pool. Certification and test records are also rejected as world-model training inputs. These checks detected cross-pool duplicates in EIR; duplicate removal therefore precedes sampling.

The answer partitions are distinct from the world model’s training records: a model record contains a state, an edit, and aggregate rates measured on a batch of training answers. Reference-score stratification is used when validation answers are drawn from training data, as described in Appendix B.

Grouped prediction evaluation is a separate procedure. Within the dynamics-training data, we separately form held-out folds by *edit_id* (edit-instance groups), *edit_text* (unseen clauses), and *rubric_state_id* (unseen states). Every example from a held-out group stays outside that fold’s model training data; group-level results are

not used for certification. This grouping tests prediction generalization and does not replace the four dataset partitions or the final test holdout.

Partition membership is fixed before optimization. Responses used for response-model fitting or candidate construction are never reused as certification examples. Likewise, no information from $\mathcal{D}_{\text{test}}$ is used for candidate generation, response prediction, candidate selection, or certification.

When duplicate response texts occur in the original data, duplicates are assigned to the same partition before optimization. This prevents identical responses from entering both the model-training and certification pathways.

G.2 Round-Wise Certification Splits

For a run containing T optimization rounds, the certification data are partitioned in advance into disjoint subsets:

$$\mathcal{D}_{\text{cert}} = \bigcup_{t=0}^{T-1} \mathcal{D}_{\text{cert}}^{(t)},$$

with

$$\mathcal{D}_{\text{cert}}^{(t)} \cap \mathcal{D}_{\text{cert}}^{(t')} = \emptyset \quad \text{for } t \neq t'.$$

Round t accesses only $\mathcal{D}_{\text{cert}}^{(t)}$. Therefore, certification outcomes from earlier rounds cannot reveal examples that will later be used to certify another update.

This distinction is important because candidate generation and response prediction adapt to earlier optimization outcomes. Fresh round-specific certification subsets keep the final commit decision separate from this adaptive optimization process. The outer benchmark split counts in [Appendix B](#) are distinct from the eligible before-edit correction and degradation counts used in each certificate. Those round-level denominators are n_t^C and $n_{t,b}^D$ in [Appendix K.1](#).

G.3 Certification Scope

For contiguous score labels from $y_{\min}$ to $y_{\max}$, let

$$\mathcal{B} = \{y_{\min}, \dots, y_{\max} - 1\}$$

denote the actual score boundaries, including $0 \mid 1$ on a 0–3 item. Finite-sample guarantees are reported only over a pre-specified subset

$$\mathcal{B}_{\text{cert}} \subseteq \mathcal{B}.$$

The certification scope $\mathcal{B}_{\text{cert}}$ specifies which boundaries the experiment seeks to protect with statistical evidence. It is fixed before candidate outcomes are inspected; boundaries cannot be removed after observing poor results or insufficient support. Each required confidence bound is then checked using the corresponding eligible certification decisions.

Boundaries outside $\mathcal{B}_{\text{cert}}$ remain part of the edit response and are still considered by the response model and candidate selection rule. They are reported empirically, but no finite-sample certification claim is made for them.

G.4 Error Groups

At optimization round t , the frozen grader evaluates $\mathcal{D}_{\text{dyn},\text{train}}$ using the current scoring specification R_t . Each incorrect prediction is associated with the score boundaries that separate the predicted score from its reference score.

Candidate construction operates on localized error groups. We write a group as

$$g = (b, d_{\text{dir}}, o),$$

where b denotes the affected score boundary, d_{dir} is the over- or under-scoring direction, and o records whether the error was already present or appeared after an earlier accepted revision.

Over-scoring at b has $\hat{z}_{i,b}^{(t)} = 1$ and $z_{i,b} = 0$; under-scoring has $\hat{z}_{i,b}^{(t)} = 0$ and $z_{i,b} = 1$. There are at most $2(K - 1)$ boundary–direction groups, or $4(K - 1)$ after separating errors present under the initial rubric from errors introduced by accepted edits. Empty groups are omitted. An answer with an error spanning multiple boundaries contributes to each affected group.

The origin field is used to describe the observed optimization history; it is not treated as a causal explanation of the error.

Let $\mathcal{E}_t$ be the set of training-answer grading errors at round t , and let $\mathcal{G}_t$ be the groups formed from those errors. The routing rule selects

$$g_t = \rho_t(\mathcal{E}_t) = (b_t^*, d_{t,\text{dir}}, o_t) \in \mathcal{G}_t$$

for candidate construction. The corresponding boundary b_t^* becomes the target boundary for that optimization round.

Selection order. We first exclude groups whose associated rubric-clause family has already failed

twice. Groups smaller than the configured minimum size are set aside unless no larger eligible group remains. We then select the largest remaining group. Ties are resolved by increasing boundary, followed by direction name and origin in lexical order. This rule uses no random sampling, so identical error records give the same selection. The origin field follows the two categories defined above: errors present initially and errors introduced later.

G.5 Scoped Candidate Generation

The candidate generator receives the current scoring specification R_t , examples from the selected error group, and relevant correctly graded responses from $\mathcal{D}_{\text{dyn}, \text{train}}$.

The signed error direction first determines the intended function of the revision. Depending on the observed failure, the candidate may tighten a scoring distinction, loosen it, or clarify an ambiguous decision rule. The target and direction are determined before the final text of the candidate clause is generated.

The generator then produces a small collection

$$\mathcal{P}_t = \{p_t^1, \dots, p_t^M\}$$

of localized revisions. Each p_t^j describes a textual modification to R_t rather than an entire replacement rubric.

Candidates are filtered before response prediction. A candidate is removed if it merely duplicates an existing rule, depends on the topic or wording of a specific response, introduces an unrelated scoring requirement, directly contradicts the current specification, or duplicates another candidate in the same pool.

Candidate generation itself never modifies the deployed scoring specification. For every candidate,

$$R_t \oplus p_t^j$$

is a temporary specification until the candidate passes both model-guided selection and empirical certification.

H Edit-Response Measurement Details

This section expands the paired edit-response definition introduced in Section 3.1.

H.1 Ordinal Boundary Representation

For contiguous labels $y_{\min}, \dots, y_{\max}$, $\mathcal{B} = \{y_{\min}, \dots, y_{\max} - 1\}$ contains all $K - 1$ thresholds.

EIR items scored 0–3 use $0 \mid 1$, $1 \mid 2$, and $2 \mid 3$; items scored 0–4 have four boundaries. Each score is represented by $K - 1$ binary decisions:

$$z_{i,b} = \mathbb{I}(y_i > b), \quad b \in \mathcal{B}.$$

Likewise, the prediction under the current rubric is

$$\hat{z}_{i,b}^{(t)} = \mathbb{I}(\hat{y}_i^{(t)} > b).$$

After applying candidate edit p ,

$$\hat{z}_{i,b}^p = \mathbb{I}(\hat{y}_i^p > b)$$

denotes the corresponding decision under $R_t \oplus p$.

A single score change can modify several boundary decisions. For example, moving from score 2 to score 4 changes both the 2|3 and 3|4 decisions. The boundary representation therefore preserves information that would be lost if an edit were represented only by the scalar difference between predicted scores.

H.2 Repair and Regression Sets

Before applying candidate p , define

$$\mathcal{F}_{t,b} = \left\{ i : z_{i,b} \neq \hat{z}_{i,b}^{(t)} \right\},$$

and

$$\mathcal{A}_{t,b} = \left\{ i : z_{i,b} = \hat{z}_{i,b}^{(t)} \right\}.$$

$\mathcal{F}_{t,b}$ contains decisions that are currently wrong at boundary b . $\mathcal{A}_{t,b}$ contains decisions that are currently correct at the same boundary.

For nonempty $\mathcal{F}_{t,b}$, target repair is measured as

$$C_{t,b}(p) = \frac{1}{|\mathcal{F}_{t,b}|} \sum_{i \in \mathcal{F}_{t,b}} \mathbb{I}(z_{i,b} = \hat{z}_{i,b}^p).$$

For nonempty $\mathcal{A}_{t,b}$, regression is measured as

$$D_{t,b}(p) = \frac{1}{|\mathcal{A}_{t,b}|} \sum_{i \in \mathcal{A}_{t,b}} \mathbb{I}(z_{i,b} \neq \hat{z}_{i,b}^p).$$

Thus, $C_{t,b}$ measures errors repaired by the edit, while $D_{t,b}$ measures correct decisions damaged by the same edit.

H.3 Validity Indicators

Some response components can be undefined because their denominator is empty. We therefore define

$$m_{t,b}^C = \mathbb{I}(|\mathcal{F}_{t,b}| > 0),$$

and

$$m_{t,b}^D = \mathbb{I}(|\mathcal{A}_{t,b}| > 0).$$

The complete validity mask is

$$\mathbf{m}_t = [\mathbf{m}_t^C; \mathbf{m}_t^D].$$

Incomplete-record drop rule. We retain an answer–edit evaluation only if the reference, before-edit, and after-edit decisions are available at all $K - 1$ boundaries. If any boundary entry is missing, the entire record is dropped; no missing entry is padded or filled with zero. This differs from an empty correction/degradation denominator: such a well-formed transition is retained, but the undefined aggregate rate coordinate receives a zero mask and is excluded from the relevant prediction loss.

H.4 Response Vector

The response associated with candidate edit p is

$$\mathbf{e}_t(p) = [\mathbf{C}_t(p); \mathbf{D}_t(p)],$$

where

$$\mathbf{C}_t(p) = [C_{t,b}(p)]_{b \in \mathcal{B}},$$

and

$$\mathbf{D}_t(p) = [D_{t,b}(p)]_{b \in \mathcal{B}}.$$

The response dimension is therefore

$$d_e = 2(K - 1).$$

The coordinates follow the score boundaries in $\mathcal{B}$ for the scoring task under consideration. Thus $d_e = 10$ for a 1–6 scale, $d_e = 8$ for a 0–4 scale, and $d_e = 6$ for a 0–3 scale.

For a candidate targeting b_t^* ,

$$C_{t,b_t^*}(p)$$

is the intended repair coordinate. The remaining repair and regression coordinates describe effects away from the target boundary.

The boundary vector is therefore used as a coordinate system for the effect of an edit rather than as a separate optimization objective.

H.5 Transition Records

Whenever a selected candidate is evaluated on $\mathcal{D}_{\text{dyn,train}}$, we record

$$(s_t, p_t^*, \mathbf{e}_t^{\text{obs}}, \mathbf{m}_t^{\text{obs}}).$$

This record is retained even when the candidate is later rejected by the certification stage. Certification controls whether the rubric changes; it does not alter the fact that the paired edit response has been observed.

The response history therefore evolves as

$$\mathcal{H}_{t+1} = \mathcal{H}_t \cup \left\{ (p_t^*, \mathbf{e}_t^{\text{obs}}, \mathbf{m}_t^{\text{obs}}) \right\}.$$

The supervised transition set becomes

$$\mathcal{T}_{t+1} = \mathcal{T}_t \cup \left\{ (s_t, p_t^*, \mathbf{e}_t^{\text{obs}}, \mathbf{m}_t^{\text{obs}}) \right\}.$$

Candidates that are never applied to $\mathcal{D}_{\text{dyn,train}}$ have no observed transition and therefore do not enter $\mathcal{H}_t$ or $\mathcal{T}_t$.

H.6 Persistent Regressions

For trajectory-level analysis, we additionally track regressions that remain active after subsequent optimization rounds.

A decision becomes an active regression when an accepted edit changes a previously correct boundary decision into an incorrect one. It remains active until a later accepted edit restores that decision.

This diagnostic distinguishes transient damage from errors that persist through the optimization trajectory. It is used only for analysis and is not an additional term in the round-level optimization objective.

I State-Conditioned Edit-Effect Model Details

This section specifies the edit-effect model, its response targets, and the transition data used for supervised updating.

I.1 State Definition

The optimization state at round t is

$$s_t = (R_t, \mathcal{H}_t, \mathbf{v}_t),$$

where R_t is the active scoring rubric, $\mathcal{H}_t$ stores previously measured edit effects, and $\mathbf{v}_t$ summarizes the frozen grader’s behavior on labeled training answers. Only R_t conditions the frozen grader; $\mathcal{H}_t$ and $\mathbf{v}_t$ provide predictive information to the edit-effect model. The supervised transition set $\mathcal{T}_t$ is distinct from the state history $\mathcal{H}_t$.

For each training answer x_i , let $\pi_{i,k}^{(t)}$ denote its available score probability for label k . Define its

cumulative probability above boundary b , threshold margin, and correctness as

$$q_{i,b}^{(t)} = \sum_{k>b} \pi_{i,k}^{(t)},$$

$$r_{i,b}^{(t)} = \left| q_{i,b}^{(t)} - \frac{1}{2} \right|, \quad c_{i,b}^{(t)} = \mathbb{I}(\hat{z}_{i,b}^{(t)} = z_{i,b}).$$

For a fixed scoring task and grader, we pool answer features and $\mathbf{c}_i$ into the fixed-dimensional summary $\mathbf{v}_t$, adding $\mathbf{q}_i$ and $\mathbf{r}_i$ only when the grader provides usable score probabilities. In particular, API-only graders need not expose per-label probabilities: missing probability and margin features are omitted, not inferred from token logits. All reference-based features use training labels only; neither certification nor test labels enter the state encoder.

I.2 Edit and State Encoders

The candidate-edit encoder is a deterministic mapping

$$\phi : p \mapsto a_p \in \mathbb{R}^{d_a}.$$

We use a general-purpose text encoder to represent the proposed clause together with its target boundary and correction direction:

$$a_p = \phi(p) = \text{Enc}(p, b^*, d_{\text{dir}}).$$

The embedding describes the edit’s content and intended target. Its effect on grading is learned from the paired measurements, rather than inferred from the embedding alone. The text encoder is separate from the frozen grader and does not require its parameters or internal activations. The text encoder is frozen; only the world-model predictor is trained. Feature extraction is deterministic. Representations are L2-normalized, computed in 32-bit floating point with batches of 32, and cached on disk. No encoder is fine-tuned on certification or test data.

The state encoder $\psi(s_t)$ combines a text embedding of the current rubric with historical edit measurements and the training-answer summary $\mathbf{v}_t$, giving $h_t \in \mathbb{R}^{d_h}$. History provides evidence about earlier attempts; the answer summary describes current grading behavior. Score-probability and margin features are included only when the grader provides them, as described in [Appendix I.1](#). Thus API-only graders can use text embeddings, returned scores, and training labels without exposing model weights or score probabilities. The grader is

never retrained. The predictor and end-to-end ablations use the same edit embedding so that changing state inputs or removing prediction does not also change the edit representation.

I.3 Factorized Edit–State Interaction

For a general state–edit pair (s, p) , let

$$a_p = \phi(p), \quad h_s = \psi(s).$$

The response logits are

$$\boldsymbol{\eta}_\theta(s, p) = W_a a_p + U [(A a_p) \odot (B h_s)] + c.$$

The predicted response is

$$\boldsymbol{\mu}_\theta(s, p) = \sigma(\boldsymbol{\eta}_\theta(s, p)),$$

where σ is applied element-wise.

The parameters have dimensions

$$W_a \in \mathbb{R}^{d_e \times d_a},$$

$$A \in \mathbb{R}^{r_{\text{int}} \times d_a}, \quad B \in \mathbb{R}^{r_{\text{int}} \times d_h},$$

$$U \in \mathbb{R}^{d_e \times r_{\text{int}}}, \quad c \in \mathbb{R}^{d_e}.$$

The term

$$W_a a_p$$

captures the component of the response associated with the candidate edit independently of the current state.

The interaction term

$$U [(A a_p) \odot (B h_s)]$$

captures how the same edit representation changes under different states.

The two projected vectors

$$A a_p, \quad B h_s \in \mathbb{R}^{r_{\text{int}}}$$

are combined through element-wise multiplication. The resulting interaction features are mapped by U into the repair–regression response space.

A full three-way interaction over edit, state, and response dimensions would require parameters proportional to $d_a d_h d_e$. The factorized form instead uses a shared interaction dimension r_{int} and avoids explicitly constructing that tensor.

I.4 Edit-Only Model

To test whether the current rubric state contributes predictive information beyond the candidate edit itself, we remove the edit-state interaction and use

$$\mu_{\text{edit}}(p) = \sigma(W_a a_p + c).$$

The response targets, edit representation, transition data, and training procedure remain unchanged. The difference between this model and the full model therefore isolates the value of conditioning on the current optimization state.

I.5 Transition Training Set

Before round t , the response model has access to

$$\mathcal{T}_t = \{(s_n, p_n, \mathbf{e}_n, \mathbf{m}_n)\}_{n=1}^{N_t}.$$

Each transition records the state from which an edit was evaluated, the candidate edit, its measured paired response, and the validity mask for that response.

An evaluated candidate remains in $\mathcal{T}_t$ regardless of whether it was subsequently committed. This allows the model to learn from unsuccessful or unsafe edits without allowing those edits to modify the deployed scoring specification.

I.6 Masked Response Loss

Equation (5) gives the training objective. The predicted and observed rate at coordinate k are $\mu_{\theta,k}(s, p)$ and e_k , respectively. The inner sum includes only coordinates with $m_k = 1$; the outer sum covers the measured state-edit records in $\mathcal{T}_t$.

An empty correction or degradation denominator leaves its rate undefined. The mask removes that coordinate instead of treating it as zero. The sigmoid constrains all predicted rates to $[0, 1]$, the same range as the observed rates.

I.7 Warm-Start Transitions

The model is initialized from paired edit-effect measurements on $\mathcal{D}_{\text{dyn,train}}$, collected before model-guided selection. The number of answer-boundary observations is

$$N_{\text{boundary}} = \sum_{(s,p) \in \mathcal{T}_0} n_{s,p}(K_{s,p} - 1),$$

where $n_{s,p}$ counts complete answer-edit records for a measured state-edit intervention. Under a balanced design, this becomes $N_{\text{state}}N_{\text{edits/state}}N_{\text{answers/edit}}(K - 1)$. Our

six-state pilot collected 36,000 boundary records, corresponding to 7,200 complete answer-edit records for the five-boundary scale. Boundary records are not treated as independent rubric trajectories. The additional observations from selected edits, including rejected edits, use the same training schema; $\mathcal{D}_{\text{cert}}$ and $\mathcal{D}_{\text{test}}$ are never used to fit the model.

State-held-out evaluation.

Using rubric_state_id-grouped folds, the full state-conditioned model reaches $R^2 = 0.283$ on unseen states, compared with 0.241 for the edit-only model (a difference of 0.042). The separate edit_id and edit_text groupings check edit-instance and unseen-clause generalization; folds do not replace the four dataset partitions defined above.

I.8 Predictive Uncertainty

We train $E = 5$ independently seeded edit-effect models, following deep-ensemble uncertainty estimation (Lakshminarayanan et al., 2017). For candidate p_t^j , let $\mu_e(s_t, p_t^j)$ denote the prediction of ensemble member e . The ensemble mean and per-coordinate sample standard deviation are

$$\bar{\mu} = \frac{1}{E} \sum_{e=1}^E \mu_e,$$

$$\sigma = \sqrt{\frac{1}{E-1} \sum_{e=1}^E (\mu_e - \bar{\mu})^2}.$$

The selection rule uses $\bar{\mu}$ as its predicted mean; the standard deviation measures disagreement among models, not a calibrated confidence interval.

The corresponding target-repair quantities are

$$\mu_{t,j,b_t^*}^C, \quad \sigma_{t,j,b_t^*}^C,$$

while the regression quantities are

$$\mu_{t,j,b}^D, \quad \sigma_{t,j,b}^D.$$

These quantities are converted into

$$\tilde{C}_{t,j} = \mu_{t,j,b_t^*}^C - \beta \sigma_{t,j,b_t^*}^C,$$

and

$$\tilde{D}_{t,j,b} = \mu_{t,j,b}^D + \beta \sigma_{t,j,b}^D.$$

The predictive mean and standard deviation affect candidate ranking and filtering only. Even if the ensemble is miscalibrated, it cannot bypass the separate certification gate: commits rely

on fresh $\mathcal{D}_{\text{cert}}$ observations and one-sided finite-sample bounds, subject to their sampling assumptions. Miscalibration can impair which candidates are explored; it does not itself provide authority to commit an edit.

I.9 Online Model Updating

After selecting p_t^* , the candidate is evaluated on $\mathcal{D}_{\text{dyn,train}}$ to obtain

$$\mathbf{e}_t^{\text{obs}}, \quad \mathbf{m}_t^{\text{obs}}.$$

The response history becomes

$$\mathcal{H}_{t+1} = \mathcal{H}_t \cup \left\{ (p_t^*, \mathbf{e}_t^{\text{obs}}, \mathbf{m}_t^{\text{obs}}) \right\},$$

and the supervised transition set becomes

$$\mathcal{T}_{t+1} = \mathcal{T}_t \cup \left\{ (s_t, p_t^*, \mathbf{e}_t^{\text{obs}}, \mathbf{m}_t^{\text{obs}}) \right\}.$$

The response model is updated from $\mathcal{T}_{t+1}$ before candidate selection at round $t + 1$.

If no candidate satisfies the predicted constraints at round t , no new paired transition is produced and

$$\mathcal{H}_{t+1} = \mathcal{H}_t, \quad \mathcal{T}_{t+1} = \mathcal{T}_t.$$

J Candidate Selection Details

This section expands the model-guided selection rule in Section 3.4.

J.1 Conservative Predicted Response

For candidate p_t^j targeting boundary b_t^* , define

$$\tilde{C}_{t,j} = \mu_{t,j,b_t^*}^C - \beta \sigma_{t,j,b_t^*}^C,$$

and

$$\tilde{D}_{t,j,b} = \mu_{t,j,b}^D + \beta \sigma_{t,j,b}^D.$$

The candidate chosen by the model is

$$p_t^* = \arg \max_{p_t^j \in \mathcal{P}_t} \tilde{C}_{t,j}$$

subject to

$$\tilde{D}_{t,j,b} \leq \epsilon_b, \quad \forall b \in \mathcal{B}.$$

The subtraction on repair penalizes uncertain predicted benefit. The addition on regression penalizes uncertain predicted harm.

These quantities are used only to determine which candidate should be measured with the frozen grader. They do not determine whether that candidate is committed.

J.2 No-Feasible-Candidate Case

If

$$\left\{ p_t^j \in \mathcal{P}_t : \tilde{D}_{t,j,b} \leq \epsilon_b \forall b \in \mathcal{B} \right\} = \emptyset,$$

the round produces no selected candidate.

In this case,

$$R_{t+1} = R_t,$$

and no new transition is added to the response history.

The optimizer therefore does not relax its predicted regression constraints merely to force an update at every round.

K Finite-Sample Certification

This section describes the statistical procedure used after a candidate has been selected and measured.

K.1 Certification Counts

At round t , candidate p_t^* is evaluated on the previously unused set

$$\mathcal{D}_{\text{cert}}^{(t)}.$$

For the target boundary b_t^* , let

$$n_t^C$$

be the number of certification examples that are incorrect before the candidate edit and therefore eligible for repair.

Let

$$x_t^C$$

be the number of those examples that become correct after the edit.

For each protected boundary b , let

$$n_{t,b}^D$$

be the number of examples that are correct before the edit, and let

$$x_{t,b}^D$$

be the number that become incorrect after the edit.

These denominators count the current rubric's decisions on the round-specific certification subset. They are neither the total certification-pool size nor the smaller human-label count on either side of a boundary. The numerator requires the paired after-edit score for the same answer. Thus pool sizes and label frequencies alone do not establish a certificate.

For a fixed candidate and boundary, each eligible answer produces a binary outcome: corrected or not, or degraded or not. The exact binomial bounds assume independent answers with a common event probability within each eligible population. These assumptions are conditional on the current rubric, selected candidate, and distribution used to sample certification answers. The fresh round-specific set prevents this candidate from being chosen using its own certification outcomes. It does not by itself establish the sampling assumptions. Dependence across boundaries is allowed by the subsequent union bound; multiple boundary decisions from one answer are not pooled as independent trials for a single rate.

K.2 Exact One-Sided Binomial Bounds

For an observed count x among n Bernoulli trials, we use exact one-sided binomial confidence bounds. They invert a binomial tail test; the beta quantile below is a way to compute that bound, not a prior distribution over edit effects. Here $\text{Beta}^{-1}(q; a, b)$ denotes the q -quantile of a beta distribution with shape parameters a, b .

For target repair, the lower Clopper–Pearson bound is

$$L^C(x, n; \alpha_C) = \text{Beta}^{-1}(\alpha_C; x, n - x + 1) \quad (9)$$

for $x > 0$, with

$$L^C(0, n; \alpha_C) = 0.$$

At round t ,

$$L_{t, b_t^*}^C = L^C(x_t^C, n_t^C; \alpha_C).$$

For regression, the upper Clopper–Pearson bound is

$$U^D(x, n; \alpha_D) = \text{Beta}^{-1}(1 - \alpha_D; x + 1, n - x)$$

for $x < n$, with

$$U^D(n, n; \alpha_D) = 1.$$

Hence,

$$U_{t, b}^D = U^D(x_{t, b}^D, n_{t, b}^D; \alpha_{D, b}).$$

K.3 Certification Rule

A candidate can be certified only when

$$b_t^* \in \mathcal{B}_{\text{cert}}.$$

The repair requirement is

$$L_{t, b_t^*}^C \geq \gamma.$$

For every certified boundary,

$$U_{t, b}^D \leq \epsilon_b, \quad \forall b \in \mathcal{B}_{\text{cert}}.$$

A candidate is committed only when all required inequalities hold.

If the rule succeeds,

$$R_{t+1} = R_t \oplus p_t^*.$$

Otherwise,

$$R_{t+1} = R_t.$$

K.4 Confidence Allocation Across Multiple Bounds

At each round, the certificate contains $Q_{\text{cert}} = 1 + |\mathcal{B}_{\text{cert}}|$ one-sided statements: one for target correction and one for degradation at each certified boundary. Let δ be the family-wise error probability.

We allocate individual error probabilities

$$\alpha_1, \dots, \alpha_{Q_{\text{cert}}}$$

such that

$$\sum_{q=1}^{Q_{\text{cert}}} \alpha_q \leq \delta.$$

We use a Bonferroni allocation to control simultaneous coverage, as in classical family-wise error control (Holm, 1979). A simple Bonferroni allocation sets

$$\alpha_q = \frac{\delta}{Q_{\text{cert}}}.$$

Under this construction, the probability that at least one reported bound fails to cover its corresponding population quantity is at most δ under the assumed sampling model.

A per-edit certificate allocates confidence only over the statements included in that edit’s certificate. If a joint claim is made across multiple optimization rounds, the confidence allocation must additionally account for the statements made across those rounds.

K.5 Minimum Sample Support

A required quantity is classified as *unsupported* when its sample size is insufficient to establish the prescribed bound, even under the most favorable possible observation.

For regression, consider the favorable case in which zero harmful transitions are observed:

$$x = 0.$$

The exact upper bound reduces to

$$U^D(0, n; \alpha) = 1 - \alpha^{1/n}.$$

To certify

$$D \leq \epsilon$$

with zero observed regressions, n must satisfy

$$1 - \alpha^{1/n} \leq \epsilon.$$

Equivalently,

$$n \geq \frac{\log \alpha}{\log(1 - \epsilon)}.$$

We therefore define

$$n_{\min}^D = \left\lceil \frac{\log \alpha}{\log(1 - \epsilon)} \right\rceil.$$

A regression coordinate with

$$n_{t,b}^D < n_{\min}^D$$

cannot receive the requested certificate and is marked unsupported.

A corresponding minimum-support condition is applied to target repair: the available number of target errors must be large enough that even a perfect observed repair rate can produce

$$L^C \geq \gamma.$$

Illustrative sample-size calculation. Suppose an edit is certified at three boundaries with $\delta = 0.05$. There are four statements: one target-correction bound and three degradation bounds. Equal allocation gives $\alpha = 0.05/4 = 0.0125$. With $\epsilon = 0.08$, zero observed degradations require

$$n_{\min}^D = \left\lceil \frac{\log(0.0125)}{\log(0.92)} \right\rceil = 53$$

initially correct decisions per checked boundary. For correction, the best case is $x = n$, giving $L^C(n, n; \alpha) = \alpha^{1/n}$. With $\gamma = 0.80$, this requires

$$n_{\min}^C = \left\lceil \frac{\log(0.0125)}{\log(0.80)} \right\rceil = 20$$

initially wrong decisions at the target boundary, all corrected. The two eligible sets at the target

are disjoint, so this illustration requires at least 73 certification answers in that round, even in the best case. The actual requirement can be larger when corrections are incomplete or degradations occur.

A joint claim over the same four statements in each of five rounds would instead allocate $\alpha = 0.05/(4 \times 5) = 0.0025$, giving $n_{\min}^D = 72$ and $n_{\min}^C = 27$ per round in the best case. These are illustrations, not reported per-round sample sizes. They show why an observed rate alone does not establish a certificate and why the eligible counts must be reported for each experiment.

K.6 Unsupported Outcomes

Certification produces three conceptually distinct outcomes.

Certified. All required repair and regression bounds are supported and satisfy the specified thresholds.

Rejected. The required quantities have sufficient support, but at least one confidence bound violates its threshold.

Unsupported. The target boundary lies outside the prespecified certification scope, or at least one required quantity lacks sufficient sample support for the prescribed bound.

Unsupported outcomes are never counted as certified. They are reported separately from statistical rejection because the two outcomes represent different reasons for withholding an update.

K.7 Per-Edit and Trajectory-Level Guarantees

A per-edit certificate applies only to the selected candidate and the fresh certification sample used at that round. It does not automatically provide a simultaneous guarantee for the entire optimization trajectory.

A joint multi-round guarantee requires confidence allocation over all statements included in that claim. Our round-wise certification splits permit such a construction because the certification observations for different rounds are separated in advance.

When an experiment does not use the required fresh certification data or joint confidence allocation, long-horizon results are described as empirical trajectory analyses rather than finite-sample certified guarantees.

K.8 Scope and Interpretation

The certification procedure addresses one narrow question: whether the observed repair and regression rates provide sufficient finite-sample evidence for the prescribed thresholds.

The certificate is conditional on the distribution from which the certification examples are sampled and on the reference labels used to define correctness.

It does not by itself establish educational validity, fairness, distribution-shift robustness, or the absence of all possible future grading errors.

The response model and certification procedure therefore play different roles. The former predicts which candidate is promising enough to evaluate; the latter determines whether the measured candidate has sufficient held-out evidence to be committed.

L Complete Optimization-State Update

For completeness, this section states the state update performed after each optimization round.

If no candidate satisfies the predicted selection constraints,

$$R_{t+1} = R_t,$$

$$\mathcal{H}_{t+1} = \mathcal{H}_t,$$

and

$$\mathcal{T}_{t+1} = \mathcal{T}_t.$$

If a candidate p_t^* is selected, it is first evaluated on $\mathcal{D}_{\text{dyn,train}}$. Its observed transition is added to the history and supervised transition data regardless of the later certification outcome:

$$\mathcal{H}_{t+1} = \mathcal{H}_t \cup \left\{ (p_t^*, \mathbf{e}_t^{\text{obs}}, \mathbf{m}_t^{\text{obs}}) \right\},$$

$$\mathcal{T}_{t+1} = \mathcal{T}_t \cup \left\{ (s_t, p_t^*, \mathbf{e}_t^{\text{obs}}, \mathbf{m}_t^{\text{obs}}) \right\}.$$

The certification outcome affects only the deployed scoring specification:

$$R_{t+1} = \begin{cases} R_t \oplus p_t^*, & \text{if } p_t^* \text{ is certified,} \\ R_t, & \text{otherwise.} \end{cases}$$

Thus, an edit may contribute information to the learned response model without being incorporated into the grader itself. This separation allows rejected transitions to improve future prediction while preventing uncertified edits from modifying the active rubric.

M Evaluation Metric Details

M.1 Edit-Response Prediction

Let $\mathcal{V}$ denote the held-out collection of measured state–edit transitions used for response-model evaluation. For transition (t, p) and boundary b , let $m_{t,b}^C$ and $m_{t,b}^D$ denote the validity indicators defined in Appendix H.3.

Let $q = (t, p, b)$ index the held-out state–edit–boundary set $\mathcal{I} = \{(t, p, b) : (t, p) \in \mathcal{V}, b \in \mathcal{B}\}$. Write $C_q = C_{t,b}(p)$, $D_q = D_{t,b}(p)$, and use the same shorthand for predictions and validity masks. Then

$$\text{MAE}_C [\text{pp}] = 100 \frac{\sum_{q \in \mathcal{I}} m_q^C |\hat{C}_q - C_q|}{\sum_{q \in \mathcal{I}} m_q^C},$$

$$\text{MAE}_D [\text{pp}] = 100 \frac{\sum_{q \in \mathcal{I}} m_q^D |\hat{D}_q - D_q|}{\sum_{q \in \mathcal{I}} m_q^D}.$$

The factor of 100 reports absolute errors in percentage points, as in Section 4.2; the underlying correction and degradation rates themselves range from 0 to 1. Undefined response coordinates are removed through the corresponding validity indicators rather than assigned an artificial value.

M.2 Candidate-Selection Metrics

Let $\mathcal{S}$ denote the optimization rounds in which a candidate edit is selected and evaluated.

The realized target repair is

$$\text{Repair}_{\text{sel}} = \frac{1}{|\mathcal{S}|} \sum_{t \in \mathcal{S}} C_{t,b_t^*}(p_t^*).$$

Mean realized regression is

$$\text{Regression}_{\text{sel}} = \frac{\sum_{t \in \mathcal{S}} \sum_{b \in \mathcal{B}} m_{t,b}^D D_{t,b}(p_t^*)}{\sum_{t \in \mathcal{S}} \sum_{b \in \mathcal{B}} m_{t,b}^D}.$$

An edit is empirically feasible when every observed degradation coordinate satisfies its tolerance. Let $\mathcal{B}_t^D = \{b \in \mathcal{B} : m_{t,b}^D = 1\}$, and define

$$F_t(p) = \mathbb{I}[\mathcal{B}_t^D \neq \emptyset] \prod_{b \in \mathcal{B}_t^D} \mathbb{I}[D_{t,b}(p) \leq \epsilon_b].$$

The empirical feasibility rate is

$$\text{EFR} = \frac{1}{|\mathcal{S}|} \sum_{t \in \mathcal{S}} F_t(p_t^*).$$

EFR is an empirical selection metric. It should not be interpreted as a finite-sample certification guarantee; formal certification is defined separately in Appendix K.

Algorithm 1: RUBRICWORLD: State-Conditioned Rubric Optimization

Input: $R_0; \mathcal{D}_{\text{dyn},\text{train}}; \mathcal{D}_{\text{dyn},\text{val}}; \mathcal{D}_{\text{cert}}; \text{rounds } T$
Output: final specification R_T

- 1 Initialize the response model from warm-start transitions;
- 2 **for** $t = 0, \dots, T - 1$ **do**
- 3 Generate candidate edits $\mathcal{P}_t$;
- 4 Form the current state s_t ;
- 5 Predict edit responses and uncertainty;
- 6 Select p_t^* using Eq. (7);
- 7 **if a candidate is selected then**
- 8 Measure its paired response on $\mathcal{D}_{\text{dyn},\text{train}}$;
- 9 Update the transition history and response model;
- 10 Compute certification bounds on held-out data;
- 11 **if** Eq. (8) **holds then**
- 12 $R_{t+1} \leftarrow R_t \oplus p_t^*$;
- 13 **else**
- 14 $R_{t+1} \leftarrow R_t$;
- 15 **else**
- 16 $R_{t+1} \leftarrow R_t$;
- 17 **return** R_T ;

N Prediction Variants and Ablation Controls

N.1 Four Predictor Variants

Figure 2(a) compares four ways of using the same candidate edit p . We write $a_p = \phi(p)$ for the text embedding of the edit and its target meta-data, $h_R = \psi_R(R_t)$ for the current rubric alone, and $h_s = \psi(R_t, \mathcal{H}_t, \mathbf{v}_t)$ for the full state (rubric text, measured edit history, and training-answer behavior). Every model applies the same element-wise sigmoid to its response logits and predicts the correction/degradation coordinates with the same masked training targets:

$$\begin{aligned}\eta_{\text{edit}} &= W_a a_p + c, \\ \eta_{\text{rubric}} &= W_a a_p + W_R h_R + c, \\ \eta_{\text{history}} &= W_a a_p + W_s h_s + c, \\ \eta_{\text{full}} &= W_a a_p + U[(A a_p) \odot (B h_s)] + c.\end{aligned}$$

The corresponding labels in Figure 2(a) are *Edit only*, *Edit + current rubric*, *Edit + rubric + history*, and *RubricWorld (Ours)*. The third model also reads $\mathbf{v}_t$ as part of its full-state summary; “history” in its short plot label therefore includes the observed grading profile. The first three variants use additive input combinations; the full variant learns a factorized edit–state interaction. This comparison varies architecture while holding data splits, response masks, edit representations, and evaluation metrics fixed.

N.2 End-to-End Controls and Cost Accounting

In Table 2, *Full RubricWorld* uses the state-conditioned predictor for candidate selection. The *w/o State* variant retains learned prediction and the same text-embedding representation but does not condition on the rubric state. The *w/o Learned Model* variant replaces predicted effects with directly measured paired candidate effects when selecting edits; all variants retain the same candidate generator and independent certification rule.

Five-round quality is measured after five refinements. The 20-round column reports only the cumulative cost of continuing the same per-round procedure for fifteen more rounds: grader, candidate budget, data roles, and certification rule are unchanged. Token counts include warm-start collection when applicable, proposal generation, paired grading, and commit checks; final test evaluation is excluded. The quality–cost sweep in Figure 3 plots means over three random seeds, with each point including the warm-start tokens. Local predictor training and embedding computation are separate from LLM-token usage and are not represented as free wall-clock compute.