EDIT DISCOVERY IN HUMAN–AI INTERACTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Human–AI interactions are becoming a rich source of data for improving AI systems: while many model responses are already satisfactory, others expose failures that could become valuable supervision if identified and corrected. We study this setting through the notion of an *editor*: a more capable evaluation process that can inspect a response and either confirm it or modify it, but whose invocation is costly. The editor may be a human expert, a stronger AI system, or the same model equipped with additional computation, context, tools, or information. The key challenge is to decide which interactions to send to the editor before knowing whether an edit will actually be made, since invoking the editor itself consumes limited resources such as expert time, inference cost, latency, or tool-use overhead. We formulate this problem as *Edit Discovery Optimization* (EDO), which seeks to minimize edit opportunities missed among skipped interactions while controlling unnecessary confirmations and respecting an editor-cost budget. We characterize the population-optimal policy and show that it takes the form of a cost-adjusted threshold on the posterior probability of an edit. Building on this structure, we develop EDIT (*Edit Discovery via Iterative Thresholding*), an online algorithm for the partial-feedback setting in which edit labels are revealed only for selected interactions. EDIT combines an adaptive edit score, a confirmation-control threshold, and a cost-aware selection rule; it satisfies the editor budget at every round and controls the average confirmation rate up to a vanishing finite-horizon term, without distributional assumptions on the interaction stream. Across four reasoning and factual question-answering benchmarks, EDIT achieves a better trade-off between editor cost, Type-I error, and Type-II error than direct LLM-based batch selection.

1 INTRODUCTION

Data has been instrumental to the progress in modern artificial intelligence. Until recently, this largely meant collecting human- or naturally-generated content, including text, code, images, and other forms of information. However, the most readily available sources of such data have become increasingly saturated: much of the public, high-quality content has already been used, while obtaining new data of comparable quality is becoming more difficult. This calls for a rethinking of what should count as useful data for the next generation of AI systems.

One emerging source is the rapidly growing stream of human–AI interaction data, which we refer to as HAI data. In many settings, users now interact with AI systems that answer questions, handle requests, write code, summarize documents, and provide customer support. This source of data is fundamentally different from traditional supervised data. The responses are generated by the deployed system itself, and therefore are not uniformly useful as new supervision: many are already satisfactory or reflect capabilities the system has already acquired, while others expose mistakes, ambiguity, incompleteness, or misalignment with user intent. If these failure cases can be identified and corrected, however, they can be transformed from raw interaction logs into valuable supervision for improving future systems.

To formalize this correction process, we assume access to an *editor*: a more capable evaluation process that, when invoked, can inspect the original interaction and either confirm the response or modify it. The editor may be a human expert, a stronger but more expensive AI system, or even the same model given additional computation, context, tools, or access to external information.

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

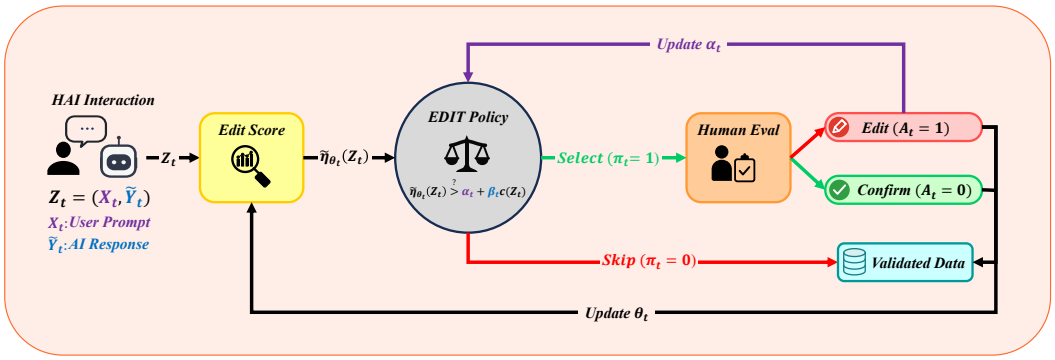

Figure 1: Overview of EDIT. Each HAI interaction $Z_t = (X_t, \widetilde{Y}_t)$ receives an estimated edit score $\widetilde{\eta}_{\theta_t}(Z_t)$. Based on this score, the policy π_t determines whether the interaction is sent to the editor or skipped. Selected responses are either confirmed or edited, yielding validated data and feedback for updating the score and policy online.

Interactions on which the editor would make a modification are precisely the cases that can yield corrective supervision.

The difficulty is deciding which interactions are worth sending to the editor. Consider a large e-commerce platform such as Amazon, where customer inquiries may increasingly be handled through AI-generated responses. Such a platform could generate tens of thousands of customer–AI interactions each day. Sending every interaction to the editor would maximize the opportunity to discover useful corrections, but could require substantial resources: human experts consume time and specialized effort, stronger AI systems incur additional inference cost, and enhanced evaluation pipelines may introduce latency or tool-use overhead. At the other extreme, invoking the editor too sparingly risks missing precisely the interactions that would provide the most valuable corrective signal. The challenge is therefore to allocate limited editorial resources toward interactions that are most likely to benefit from correction, while avoiding unnecessary editor invocations on responses that would simply be confirmed. This motivates the central question of this paper:

Can we selectively identify the interactions on which the editor would make an edit, while substantially reducing the cost of invoking the editor?

We answer this question in two steps. First, we formulate *Edit Discovery Optimization* (EDO), a constrained selection problem that formalizes the trade-off between discovering edit opportunities, avoiding unnecessary confirmations, and respecting the available editor budget. We characterize the population-optimal solution and show that it takes the form of a cost-adjusted threshold on the posterior probability of an edit. Building on this structure, we then develop EDIT (*Edit Discovery via Iterative Thresholding*), a practical online algorithm for the setting in which edit outcomes are observed only for interactions that are actually sent to the editor. This partial-feedback setting creates an additional algorithmic challenge: edit labels are revealed only for selected interactions, yet the discovery errors of interest depend on both selected and skipped interactions, so in particular the policy never observes online the skipped labels that determine missed edits. The policy must therefore learn where edits are likely to occur from selectively revealed feedback while regulating its selections without access to the full set of labels. EDIT addresses this by combining an adaptive edit score with a dynamically updated confirmation-control threshold and a cost-aware selection rule.

Contributions. Our main contributions are as follows:

- We introduce Edit Discovery Optimization (EDO), a formal framework for selecting HAI interactions under limited editor resources. The formulation distinguishes between confirmations among selected interactions and missed edits among skipped interactions, while explicitly accounting for interaction-specific editor costs.
- We characterize the optimal population policy and show that it selects an interaction when its posterior edit probability exceeds a cost-adjusted threshold. This result identifies the posterior

probability of editor modification as the appropriate edit score and clarifies how editor cost should affect the decision.

- Building on this characterization, we develop EDIT, an online selection method that combines an edit score, an adaptive confirmation-control threshold, and a batchwise cost multiplier. We prove that EDIT satisfies the editor-budget constraint at every round and controls the average per-round Type-I error up to a finite-horizon term that vanishes with the number of rounds.
- We evaluate EDIT on four reasoning and factual question-answering benchmarks using Llama-3.3-70B-Instruct, with additional experiments using Qwen3-32B, response-length editor costs, and structured distribution shifts. The results show that informative edit scores, particularly the learned representation score, reduce missed edits relative to direct LLM-based batch selection while maintaining reliable Type-I behavior and feasible editor cost.

2 Related Work

We discuss the closest work here and defer broader discussion to Appendix A.

AI-assisted data collection and selective human intervention. Classical active learning selects examples whose labels are expected to improve a predictive model (Settles, 2009; Zhang et al., 2022), while more recent AI-assisted annotation methods allocate labeling effort between a model and a human expert (Candès et al., 2026; Huang et al., 2026). Selective prediction and learning to defer similarly identify inputs on which a model should abstain or hand control to a downstream expert (Geifman and El-Yaniv, 2019; Madras et al., 2018; Mozannar and Sontag, 2020). Our setting differs in that the model response has already been produced and the goal is to discover responses that should be edited, rather than to complete a labeled dataset or improve prediction through deferral. Moreover, feedback is partial: responses that are not sent to the editor remain unlabeled, so the policy must learn only from edit decisions revealed for previously selected interactions.

Calibrated selection and online control. Conformal selection calibrates scores to return outputs that can be accepted while controlling false discoveries or related risks (Jin and Candès, 2023; Gui et al., 2024; 2025; Liu et al., 2026; Bai and Jin, 2026), while adaptive conformal inference and risk control motivate our threshold adaptation (Gibbs and Candès, 2021; Angelopoulos et al., 2024). These methods typically use labeled calibration data and exchangeability-type assumptions. In contrast, EDIT selects interactions likely to require correction and provides guarantees on arbitrary streams with labels revealed only after editor feedback, while enforcing a cost constraint.

3 Fundamentals of Edit Discovery

In this section, we formalize Edit Discovery Optimization (EDO) and characterize its optimal population policy. The formulation makes explicit the two ways a selection policy can fail: spending editor effort on responses ultimately confirmed without modification, or skipping responses that would have benefited from editing. It also incorporates the cost of invoking the editor under limited evaluation resources. The resulting analysis characterizes how selection should depend on edit likelihood and cost and reveals the structure that our finite-sample method seeks to emulate.

3.1 Problem Formulation

Let $\mathcal{X}$ and $\mathcal{Y}$ be the prompt and answer spaces. For each prompt $X \in \mathcal{X}$, the AI system produces a response $\widetilde{Y} \in \mathcal{Y}$. Before editor feedback, the observable item is the prompt–response pair

$$Z = (X, \widetilde{Y}) \in \mathcal{Z}, \qquad \mathcal{Z} := \mathcal{X} \times \mathcal{Y}.$$

If the item is selected, the editor inspects the AI response and provides a validated final answer $Y \in \mathcal{Y}$. When the response is unsatisfactory, this answer may correct an error, resolve ambiguity, complete missing information, or better align the response with the user's intent. Invoking the editor consumes limited resources such as expert time, annotation budget, or latency, so the goal is to select interactions most likely to yield useful corrections rather than review every response.

For each selected item, define the edit indicator as the function

$$A(Z, Y) := \begin{cases} 1, & \text{if the editor modifies the AI answer, } Y \neq \widetilde{Y}, \\ 0, & \text{if the editor confirms the AI answer, } Y = \widetilde{Y}. \end{cases}$$

For notational brevity, we write $A := A(Z, Y)$ when the underlying interaction and validated answer are clear from context. Thus, $A = 1$ marks an informative edit outcome in which the interaction reveals a failure mode and produces corrective data that can be used, for example, as supervision for subsequent model training. Conversely, $A = 0$ indicates that the editor accepts the AI response without modification. Such confirmations may still be useful for auditing system behavior and for reinforcing capabilities that the model already performs reliably.

Central population quantity. Efficient edit discovery requires predicting which AI responses are most likely to trigger an editor modification before the editor is invoked. The central population quantity is therefore the posterior edit probability

$$\eta(z) := \Pr[A = 1 \mid Z = z] \,.$$

For a given interaction z, this quantity measures the probability that invoking the editor will result in an edit rather than a confirmation. It therefore provides a natural measure of the expected corrective value of selecting that interaction. Responses with larger values of $\eta(z)$ are more likely to produce corrective data, whereas responses with smaller values are more likely to be confirmed without modification. As we show in Section 3.2, this quantity determines the ordering of interactions under the population-optimal policy, up to an adjustment that accounts for the cost of invoking the editor.

Editor cost. Each interaction has a finite editor cost $c(z) > 0$, known at selection time. It may encode human time, monetary expense, latency, domain expertise, or any normalized resource needed to obtain feedback in practice. For instance, asking the editor to inspect a long mathematical derivation can be substantially more costly than checking a short factual answer. Allowing c to depend on z lets the policy trade off the expected value of an edit against the total effort required to obtain it.

Selection policy and discovery errors. A population selection policy is a measurable function $\pi : \mathcal{Z} \to [0, 1]$, where $\pi(z)$ is the probability that item z is sent to the editor. Conditional on $Z = z$, let $S_\pi \in \{0, 1\}$ denote the realized selection decision, with $\Pr[S_\pi = 1 \mid Z = z] = \pi(z)$, generated independently of A given Z. Thus $\pi(z) = 1$ means the item is selected, $\pi(z) = 0$ means it is skipped, and intermediate values allow randomization. Selecting broadly consumes scarce resources on responses that the editor may simply confirm, whereas selecting narrowly can miss failures that would have produced valuable corrections. Edit discovery therefore seeks to concentrate selection on likely edits while controlling both the confirmation burden and total editor cost.

We define the Type-I error of a policy π as

$$\mathrm{Err}_{\mathrm{I}}(\pi) := \Pr[A = 0 \mid S_\pi = 1] \,, \tag{1}$$

the confirmation rate among selected examples. This is the probability that a selected item is already correct and does not require editor modification. Similarly, we define the Type-II error of π as

$$\mathrm{Err}_{\mathrm{II}}(\pi) := \Pr[A = 1 \mid S_\pi = 0] \,, \tag{2}$$

the edit rate among unselected examples. This is the probability that a skipped item would have produced an editor correction. The goal is to minimize Type-II error while controlling Type-I error and respecting the available editor budget.

Edit Discovery Optimization (EDO)

Let $0 \leq \varepsilon < 1$ denote the Type-I error tolerance, and let $B \geq 0$ be the population editor budget. The population edit-discovery problem is[1]

$$\begin{aligned} \underset{\pi:\mathcal{Z}\to[0,1]}{\text{minimize}} \quad & \Pr[A = 1 \mid S_\pi = 0] \\ \text{subject to} \quad & \Pr[A = 0 \mid S_\pi = 1] \leq \varepsilon, \\ & \mathbb{E}[\pi(Z)c(Z)] \leq B. \end{aligned} \tag{EDO}$$

The parameter ε specifies the Type-I error tolerance, equivalently the maximum confirmation rate allowed among selected interactions. Smaller values require the selected set to contain a higher fraction of items that actually lead to edits, thereby placing greater emphasis on edit precision. The budget B controls the total editor cost. Thus, EDO provides a decision-theoretic formulation of selective editor invocation, making it possible to study how corrective value, selection reliability, and resource constraints jointly determine which interactions should enter the editor feedback loop.

The remainder of the analysis proceeds in two steps. Section 3.2 studies the idealized population regime, where the distribution of (Z, A) is known, and derives the form of an optimal solution to EDO. Section 4 replaces unavailable population quantities with their observed, sequential counterparts, yielding a practical distribution-free algorithm with finite-horizon guarantees.

3.2 Optimal Policy in the Population Regime

We study EDO in the population regime, where the joint distribution of (Z, A) is assumed known. Although unavailable in practice, this regime provides a useful benchmark for understanding the structure of the optimal policy and how posterior edit probability and editor cost should jointly determine selection. This characterization guides the method in Section 4, where these quantities are unknown and the policy must learn from partial feedback revealed only for selected interactions.

Theorem 1 (Optimal population policy)**.** *Let $\pi^\star$ be an optimal policy for* (EDO)*. Then there exist scalars $\alpha^\star, \beta^\star \geq 0$ such that*

$$\pi^\star(z) = \mathbb{1}\left\{\eta(z) > \alpha^\star + \beta^\star c(z)\right\}, \tag{3}$$

with possible randomization on the boundary $\{z : \eta(z) = \alpha^\star + \beta^\star c(z)\}$, if needed.

Theorem 1 reduces a constrained set-selection problem to a one-dimensional ranking rule. An interaction is selected when its probability of producing an edit exceeds a global threshold plus a cost-dependent adjustment. The scalar $\alpha^\star$ controls how informative the selected set must be, with larger values filtering out interactions with relatively low edit probability. The term $\beta^\star c(z)$ accounts for scarce editor capacity by requiring a higher edit probability for more expensive interactions.

This characterization also explains why learning an edit score is central. The online controller need not reconstruct the full joint distribution of (Z, A), it only needs a score that orders interactions like η and adaptive analogues of the population thresholds. The cost term reflects scarce editor capacity, as it is active only when the budget is restrictive and vanishes otherwise.

This directly motivates the online algorithm. In practice, η, $\alpha^\star$, and $\beta^\star$ are unknown, so we replace η with an adaptive score $\widetilde{\eta}_t$, update the global threshold from observed feedback, and choose the cost multiplier each round to enforce the budget. The online policy thus preserves the structure of the population-optimal rule using quantities observable during deployment.

4 Online Edit Discovery

The population analysis identifies the form of the optimal EDO policy, but its ingredients are unavailable in deployment: the interaction distribution is unknown, η cannot be evaluated, and edit labels are observed only for selected responses. In this section, we address the same edit discovery

[1]Throughout the population analysis, whenever a conditional probability is represented as a ratio whose numerator and denominator are both zero, we use the convention $0/0 = 0$.

task under this information constraint, without imposing any distributional assumptions on the incoming stream. We replace the population errors in Section 3.1 by their online counterparts and replace the unknown population threshold rule by an adaptive finite-sample policy. Thus, the online setting specifies how to operationalize EDO from a stream of partially revealed feedback.

We now describe the online interaction protocol used throughout this section.

Online interaction protocol

At each round $t = 1, \dots, T$, the following sequence occurs:

1. **Batch arrival.** A batch $\mathcal{D}_t = \{Z_{t,i}\}_{i=1}^{n_t}$, with $n_t := |\mathcal{D}_t|$, arrives together with predictable editor costs $\{c(Z_{t,i})\}_{i=1}^{n_t}$. The edit labels $A_{t,i} := \mathbb{1}\{Y_{t,i} \neq \widetilde{Y}_{t,i}\}$ are not yet observed.
2. **Selection.** Based on the information available at round t, the policy chooses $\pi_t \in \{0,1\}^{n_t}$, where $\pi_{t,i} = 1$ sends interaction i to the editor and $\pi_{t,i} = 0$ skips it.
3. **Edit feedback.** The editor returns $Y_{t,i}$, and hence reveals $A_{t,i}$, only for interactions with $\pi_{t,i} = 1$. The newly revealed feedback may then be used to update the policy before the next batch arrives.

We define the following two quantities as the empirical counterparts of the population Type-I and Type-II errors in Equations (1) and (2), respectively. For a realized stream, we define the online Type-I error as the average within-round confirmation rate,

$$\widehat{\mathrm{Err}}_{\mathrm{I}}(\pi) := \frac{1}{T}\sum_{t=1}^{T} \frac{\sum_{i=1}^{n_t} \pi_{t,i}(1 - A_{t,i})}{1 \vee \sum_{i=1}^{n_t} \pi_{t,i}}. \tag{4}$$

Likewise, we define the online Type-II error as the average within-round edit rate among skipped interactions,

$$\widehat{\mathrm{Err}}_{\mathrm{II}}(\pi) := \frac{1}{T}\sum_{t=1}^{T} \frac{\sum_{i=1}^{n_t} (1 - \pi_{t,i}) A_{t,i}}{1 \vee \sum_{i=1}^{n_t} (1 - \pi_{t,i})}. \tag{5}$$

Here, we use the notation $a \vee b := \max\{a, b\}$. We treat each round as the relevant operational unit and therefore average the corresponding conditional error rates across rounds rather than pooling all interactions. This gives equal weight to different selection rounds and prevents high-volume rounds from dominating the error criterion. $\widehat{\mathrm{Err}}_{\mathrm{I}}$ records the average confirmation rate among selected interactions across rounds, whereas $\widehat{\mathrm{Err}}_{\mathrm{II}}$ records the average edit rate among skipped interactions. The latter is observable only retrospectively in our fully labeled evaluation streams, while in deployment it remains counterfactual because skipped labels are not revealed.

For the same Type-I error tolerance $0 \leq \varepsilon < 1$ as in EDO, we seek low $\widehat{\mathrm{Err}}_{\mathrm{II}}$ while controlling $\widehat{\mathrm{Err}}_{\mathrm{I}} \leq \varepsilon$. The population budget $\mathbb{E}[\pi(Z)c(Z)] \leq B$ becomes a per-round constraint,

$$\sum_{i=1}^{n_t} \pi_{t,i} c(Z_{t,i}) \leq B_t, \qquad t = 1, \dots, T. \tag{6}$$

This provides the sequential analogue of EDO under the information available in the online setting. In particular, $\widehat{\mathrm{Err}}_{\mathrm{II}}$ depends on the unobserved labels of skipped interactions and is therefore an evaluation criterion rather than a signal the policy can optimize directly during deployment. By contrast, the Type-I contribution at round t depends only on selected labels, and the budget is known before selection; both can be controlled directly by the online policy.

Theorem 1 suggests how to proceed without the population quantities. We replace η by a parametrized edit score $\widetilde{\eta}_\theta(z) \in [0,1]$, where θ denotes the parameters governing the score function, replace $\alpha^\star$ by an adaptive threshold α_t updated from observed confirmations, and choose a batchwise multiplier β_t to enforce Equation (6). We call the resulting method EDIT: *Edit Discovery via Iterative Thresholding*. When the score is learnable, EDIT also updates θ using the previously revealed labels.

4.1 EDIT POLICY

Motivated by the population solution, EDIT uses the binary threshold policy

$$\pi_{t,i}^{\mathrm{EDIT}} = \mathbb{1}\left\{\widetilde{\eta}_{\theta_t}(Z_{t,i}) > \alpha_t + \beta_t c(Z_{t,i})\right\}, \tag{7}$$

where $\alpha_t \geq 0$ is the online confirmation-control threshold and $\beta_t \geq 0$ is the batchwise cost multiplier. Initialize $\alpha_1 = 0$, and fix threshold stepsizes $\{\gamma_t\}_{t=1}^{T}$, with $\gamma_t > 0$. Given α_t, define

$$\begin{aligned} \beta_t &= \inf\left\{\beta \geq 0 : \sum_i \mathbb{1}\left\{\widetilde{\eta}_{\theta_t}(Z_{t,i}) > \alpha_t + \beta c(Z_{t,i})\right\} c(Z_{t,i}) \leq B_t\right\}, \\ \alpha_{t+1} &= \max\left\{0, \alpha_t + \gamma_t\left(\frac{\sum_i \pi_{t,i}^{\mathrm{EDIT}}(1 - A_{t,i})}{1 \vee \sum_i \pi_{t,i}^{\mathrm{EDIT}}} - \varepsilon\right)\right\}. \end{aligned} \tag{8}$$

Note that the set inside the infimum is nonempty, since the sum is zero for all sufficiently large β.

The batchwise β_t follows the same principle as the population multiplier. Given α_t, it is zero when the unadjusted rule is budget-feasible and otherwise increases only enough to make the selected set satisfy the current budget.

After the selected examples are evaluated and their labels are observed, the score parameter is updated by a gradient step on the selected-sample cross-entropy loss

$$\ell_t(\theta) = \frac{\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}} \left[-A_{t,i} \log \widetilde{\eta}_\theta(Z_{t,i}) - (1 - A_{t,i}) \log\left(1 - \widetilde{\eta}_\theta(Z_{t,i})\right)\right]}{1 \vee \sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}}. \tag{9}$$

For a score learning rate $\xi_t > 0$, the score update is

$$\theta_{t+1} = \theta_t - \xi_t \nabla_{\theta_t} \ell_t(\theta) = \theta_t - \xi_t \frac{\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}} \left[\frac{\widetilde{\eta}_{\theta_t}(Z_{t,i}) - A_{t,i}}{\widetilde{\eta}_{\theta_t}(Z_{t,i})\left(1 - \widetilde{\eta}_{\theta_t}(Z_{t,i})\right)}\right] \nabla_{\theta_t} \widetilde{\eta}_{\theta_t}(Z_{t,i})}{1 \vee \sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}}. \tag{10}$$

We now show that EDIT controls the average per-round Type-I error up to a vanishing term in T.

Theorem 2 (Online constraint guarantee)**.** *Suppose that $\gamma_t = \gamma > 0$ for all $t = 1, \ldots, T$. The* EDIT *policy satisfies*

$$\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}} c(Z_{t,i}) \leq B_t, \qquad t = 1, \ldots, T.$$

Moreover,

$$\widehat{\mathrm{Err}}_{\mathrm{I}}(\pi^{\mathrm{EDIT}}) \leq \varepsilon + \frac{1 + \gamma}{\gamma T}. \tag{11}$$

This guarantee requires no distributional assumptions on the incoming interaction stream: batches may be dependent, non-identically distributed, or nonstationary, and the bound applies to the realized interaction and observed feedback sequence. It is also agnostic to how the edit score is constructed and therefore applies across a broad range of score designs. We instantiate random, representation-based, and LLM-based scores in Section 5.1. The full EDIT algorithm appears in Appendix C.

5 EXPERIMENTS

We evaluate EDIT by Type-I error among selected interactions, Type-II error among skipped interactions, and editor cost. Section 5.1 describes the scores, models, datasets, baselines, protocol, and metrics, and Section 5.2 presents the main results. Appendix D provides experimental details, additional evaluations, and representation ablations; Appendix D.2 validates the benchmark-derived edit labels through a human audit with GPT-5.6 Sol as an independent model annotator.

5.1 Experimental Settings

Edit-score design. Theorem 2 allows any adaptive score in $[0, 1]$; score quality affects efficiency, not controller validity. We consider three edit-score functions in our experiments:

- **Random score (EDIT-RAND).** As an uninformative control, each interaction receives an independent score $\widetilde{\eta}_t(Z_{t,i}) \sim \mathrm{Uniform}(0, 1)$. It retains the same α_t update and budget-feasible rule as EDIT but removes semantic information, isolating the contribution of the controller alone.
- **LLM score (EDIT-LLM).** Motivated by evidence that capable language models can estimate answer correctness or uncertainty (Kadavath et al., 2022), this training-free variant asks an LLM, without the reference answer, to estimate from the prompt–response pair the probability of a human edit and return a score in $[0, 1]$ for each candidate before editor invocation.
- **Representation score (EDIT-REP).** Our primary score design maps the prompt–response pair Z to a frozen representation $\phi(Z) \in \mathbb{R}^d$ and learns a logistic linear probe defined as:

$$\widetilde{\eta}_\theta(Z) = \sigma(\langle \phi(Z), \theta \rangle), \qquad \sigma(u) = \frac{1}{1 + \exp(-u)}. \tag{12}$$

  Here, only θ is learned while ϕ remains fixed, and the selected-example gradient in Eq. (10) becomes

$$\nabla_\theta \ell_t(\theta) = \frac{\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}} \big(\widetilde{\eta}_\theta(Z_{t,i}) - A_{t,i}\big) \phi(Z_{t,i})}{1 \vee \sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}}. \tag{13}$$

  Appendix D.8 studies larger representation models. We use all-MiniLM-L6-v2 (Wang et al., 2020; Reimers and Gurevych, 2019) and update only the linear probe online. At 22.7M parameters, it is over three orders of magnitude smaller than the 70B response model, testing whether EDIT-REP can rank edits with a lightweight encoder in the online deployment setting.

Models and datasets. The primary response model is Llama-3.3-70B-Instruct (Grattafiori et al., 2024), used with deterministic decoding to generate responses and, when applicable, provide the EDIT-LLM score or execute LLM-Select. Appendix D.5 repeats the experiment with Qwen3-32B (Yang et al., 2025). We use four complementary benchmarks: MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024), a 500-problem MATH subset; MMLU-Pro (Wang et al., 2024), a ten-option multi-domain reasoning benchmark; PopQA (Mallen et al., 2023), an entity-centric long-tail factual QA benchmark; and GPQA (Rein et al., 2024), a graduate-level multiple-choice science benchmark.

Baselines. Our main baseline, LLM-Select, receives the full indexed batch, item costs, and budget and returns a budget-feasible subset requiring correction. It gives a capable LLM joint access to all candidates for direct ranking and allocation, testing this prompted decision against EDIT's explicit edit score, Type-I feedback through α_t, and budget control through β_t. We also compare against EDIT-RAND and EDIT-LLM as training-free score variants.

Online protocol and metrics. We compare Type-II error at comparable editor cost and Type-I error; lower Type-II error under matched conditions indicates better edit discovery. Unless stated otherwise, runs use $T = 250$ rounds with $n_t = 10$, sampling batches with replacement from a fixed response pool, with every method receiving the same ordered stream. The main setting uses $c(z) = 1$ and $B_t = 5$; Appendix D.7 considers nonstationary streams. For each model–dataset pair, we choose ε so the bound in Equation (11) approximately matches LLM-Select's online Type-I error. We report running-average Type-I and Type-II errors and editor cost; Appendix D.3 gives complementary metrics and operating points. We audit 200 held-out interactions (50 per dataset) with a human and GPT-5.6 Sol under the same confirm-versus-edit criterion; Appendix D.2 reports their agreement.

5.2 Results

Figure 2 shows the online trajectories for Llama-3.3-70B-Instruct. Across all four datasets, EDIT-REP generally attains lower Type-II error than LLM-Select with comparable or more stable Type-I behavior. This supports separating an edit score, adaptive Type-I control, and budget mechanism rather than direct LLM batch selection. Unlike EDIT, LLM-Select cannot explicitly adapt to the observed confirmation rate, making its Type-I behavior less consistent across datasets.

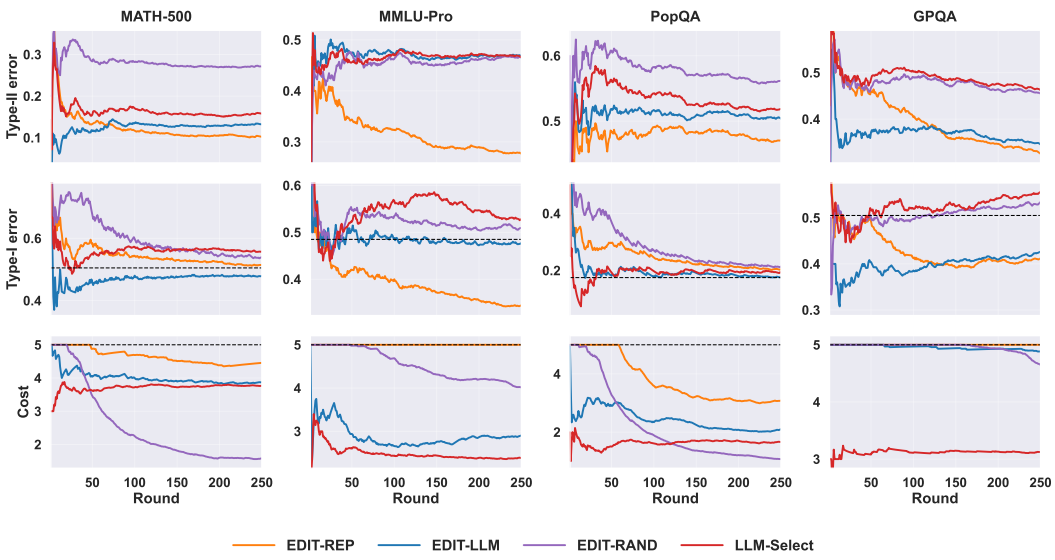

Figure 2: Main results with Llama-3.3-70B-Instruct. Columns correspond to MATH-500, MMLU-Pro, PopQA, and GPQA. Rows report running-average Type-II error, Type-I error, and editor cost; dashed horizontal lines mark the configured Type-I error tolerance and per-round editor budget.

Score quality also strongly affects discovery. EDIT-REP attains the lowest final Type-II error on all four datasets, with especially clear gains on MMLU-Pro and GPQA; EDIT-LLM also improves over EDIT-RAND on several datasets. Since all variants share the selection rule, the differences reflect score information: the controller sets selectivity and the score prioritizes interactions within budget.

The main EDIT-REP score uses a 22.7M-parameter MiniLM encoder, tiny relative to the 70B response model. Appendix D.8 shows Qwen3-0.6B, 1.7B, and 4B representations progressively improving edit discovery, with 4B attaining the lowest final Type-II error on all four datasets, suggesting stronger encoders can further improve performance in this setting.

The Type-I trajectories show α_t adapting to the observed confirmation rate toward the configured tolerance. Cost must be interpreted jointly with both errors: lower cost is not preferable if a conservative policy misses more editable responses. We therefore compare Type-II error under the prescribed Type-I tolerance and budget, where EDIT-REP is consistently strongest.

6 Conclusion and Limitations

We introduced edit-discovery optimization and EDIT, a cost-aware online method that combines edit scoring with adaptive Type-I control and per-round budget enforcement. The population analysis shows that optimal selection follows a cost-adjusted threshold on edit probability, motivating the decomposition used by EDIT. In the online setting, the method controls the confirmation rate among selected interactions without distributional assumptions while satisfying the editor budget by construction. Across four reasoning and factual question-answering benchmarks, EDIT-REP reduces missed edits relative to direct LLM-based batch selection while maintaining reliable Type-I behavior, showing the value of separating edit scoring, online control, and budget allocation. The results further indicate that score quality matters: even a lightweight representation model provides useful rankings, while stronger encoders can further improve edit discovery. This modular design allows stronger edit predictors to be incorporated without changing the online control rule.

Our experiments use benchmark-derived edit labels, supported by a limited 200-example human audit, rather than large-scale human editing. We also evaluate edit discovery itself rather than the downstream value of incorporating the resulting corrections into model training or adaptation. Future work can study richer human editing workflows, heterogeneous editing costs, and whether discovered edits translate into measurable downstream model improvements in realistic human–AI deployment settings with diverse users, domains, editor expertise, and resource constraints.

AI USE STATEMENT

During the preparation of this manuscript, the authors used AI tools only for limited writing assistance, including proofreading and improving readability. In addition, an LLM (GPT-5.6 Sol) was used as an independent annotator in the human audit experiment described in Appendix D.2. This usage is part of the experimental evaluation and is fully disclosed in the paper. All technical contributions were developed and verified by the authors, who take full responsibility for the content of this submission.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the proposed algorithm, theoretical analysis, experimental protocol, datasets, evaluation metrics, model configurations, and hyperparameter settings in the paper. The implementation code will be publicly released upon acceptance.

REFERENCES

Anastasios N. Angelopoulos, Stephen Bates, Clara Fannjiang, Michael I. Jordan, and Tijana Zrnic. Prediction-powered inference. *Science*, 382(6671):669–674, 2023. doi: 10.1126/science.adi6000.

Anastasios N. Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. Conformal risk control. *Journal of Machine Learning Research*, 25(64):1–45, 2024.

Anastasios N. Angelopoulos, Jacob Eisenstein, Jonathan Berant, Alekh Agarwal, and Adam Fisch. Cost-optimal active AI model evaluation. *arXiv preprint arXiv:2506.07949*, 2025.

Tian Bai and Ying Jin. Conformal selective prediction with general risk control. *arXiv preprint arXiv:2603.24704*, 2026.

Emmanuel J. Candès, Andrew Ilyas, and Tijana Zrnic. Probably approximately correct labels. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Adam Fisch, Joshua Maynez, R. Alex Hofer, Bhuwan Dhingra, Amir Globerson, and William W. Cohen. Stratified prediction-powered inference for hybrid language model evaluation. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Yonatan Geifman and Ran El-Yaniv. SelectiveNet: A deep neural network with an integrated reject option. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2151–2159, 2019.

Isaac Gibbs and Emmanuel Candès. Adaptive conformal inference under distribution shift. In *Advances in Neural Information Processing Systems*, volume 34, pages 1660–1672, 2021.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Yu Gui, Ying Jin, and Zhimei Ren. Conformal alignment: Knowing when to trust foundation models with guarantees. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Yu Gui, Ying Jin, Yash Nair, and Zhimei Ren. ACS: An interactive framework for conformal selection. *arXiv preprint arXiv:2507.15825*, 2025.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems: Datasets and Benchmarks Track*, 2021.

Huipeng Huang, Wenbo Liao, Huajun Xi, Hao Zeng, Mengchen Zhao, and Hongxin Wei. Model-agnostic selective labeling with provable statistical guarantees, 2026. URL https://arxiv.org/abs/2510.14581.

Ying Jin and Emmanuel J. Candès. Selection by prediction with conformal p-values. *Journal of Machine Learning Research*, 24(244):1–41, 2023.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. Holistic evaluation of language models. *Transactions on Machine Learning Research*, 2023.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Kangdao Liu, Huajun Xi, Chi Man Vong, and Hongxin Wei. Online conformal selection with accept-to-reject changes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 23765–23773, 2026. doi: 10.1609/aaai.v40i28.39551.

David Madras, Toniann Pitassi, and Richard Zemel. Predict responsibly: Improving fairness and accuracy by learning to defer. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023.

Hussein Mozannar and David Sontag. Consistent estimators for learning to defer to an expert. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7076–7087, 2020.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 2019.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Burr Settles. Active learning literature survey. Technical Report 1648, University of Wisconsin–Madison, 2009.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, 2020.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3018.

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

Zhisong Zhang, Emma Strubell, and Eduard Hovy. A survey of active learning for natural language processing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022.

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

Tijana Zrnic and Emmanuel Candès. Active statistical inference. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 62993–63010, 2024.

APPENDIX CONTENTS

A Related Work **14**

B Proofs **15**

B.1 Proof of Theorem 1 15

B.2 Proof of Theorem 2 17

C EDIT Pseudocode **17**

D Additional Experiments **18**

D.1 Additional Tables and Figures 18

D.2 Human Audit of Benchmark-Derived Edit Labels 18

D.3 Operating-Point Trade-offs and Complementary Metrics. 18

D.4 List of LLM Prompts 20

D.5 Additional Experiments with Qwen3-32B 26

D.6 Response-Length-Aware Editor Costs 26

D.7 Additional Experiments under Distribution Shift 27

D.8 Effect of Embedding-Model Size 28

A Related Work

Efficient AI evaluation. A growing line of work studies how model predictions can reduce the cost of evaluation. Prediction-powered inference combines a small labeled sample with predictions on a much larger unlabeled sample to estimate population quantities with valid uncertainty (Angelopoulos et al., 2023). This idea has been specialized to hybrid language-model evaluation through stratified sampling (Fisch et al., 2024), while cost-optimal active evaluation allocates queries between weak and strong raters to obtain an efficient estimate of mean strong-rater performance (Angelopoulos et al., 2025). Active statistical inference similarly chooses which labels to acquire in order to improve the precision of confidence intervals or tests (Zrnic and Candès, 2024). These works are closely related to our motivation because they explicitly account for expensive human evaluation. Their objective, however, is accurate statistical inference about an aggregate quantity. In contrast, our goal is to identify the individual interactions that should be edited, while controlling confirmations among selected interactions and respecting the available editor budget. Standard LLM evaluation frameworks and LLM-based judges also focus on measuring model quality (Liang et al., 2023; Zheng et al., 2023), rather than deciding which interactions should enter an online human feedback loop.

AI-assisted data collection. Classical active learning selects examples whose labels are expected to improve a predictive model (Settles, 2009; Zhang et al., 2022). More recent work considers hybrid annotation pipelines in which an AI system supplies some labels and an expert supplies the remainder. PAC labeling uses model uncertainty to construct a completed dataset with bounded aggregate labeling error (Candès et al., 2026), whereas Conformal Labeling controls the false discovery rate among labels assigned by the AI system (Huang et al., 2026). Although these methods also reduce human annotation effort, their output is a fully labeled dataset in which skipped expert labels are replaced by model labels. In our setting, skipped responses remain unlabeled, and the purpose of selection is to discover corrections rather than to approximate all missing labels. This distinction is important in practice because our policy must learn from only the edit decisions revealed for the interactions it previously selected.

Conformal selection and online calibration. Conformal selection wraps a prediction model with calibrated p-values and selects units whose unobserved outcomes exceed a target while controlling false discoveries (Jin and Candès, 2023). Conformal Alignment applies this perspective to foundation-model outputs and certifies a selected set as trustworthy (Gui et al., 2024). Related extensions support interactive data analysis, irreversible online decisions, and general bounded risks (Gui et al., 2025; Liu et al., 2026; Bai and Jin, 2026). These methods select outputs to accept or trust, whereas our positive decision sends an output to a human because it is likely to require correction. Reversing an acceptance rule does not preserve its guarantee or yield control of confirmations among selected items. Moreover, these methods generally require labeled calibration data or exchangeability-type conditions. Our threshold update is technically related to adaptive conformal inference and conformal risk control (Gibbs and Candès, 2021; Angelopoulos et al., 2024), but our guarantee applies to an arbitrary stream with labels revealed only after editor feedback and is coupled to a per-round cost constraint.

Selective prediction and learning to defer. Selective prediction allows a model to abstain on inputs for which its prediction is unreliable, including end-to-end methods that optimize the risk–coverage trade-off (Geifman and El-Yaniv, 2019). Learning to defer extends this formulation by explicitly accounting for the accuracy, bias, or cost of a downstream human expert (Madras et al., 2018; Mozannar and Sontag, 2020). In these formulations, deferral is an alternative action used to improve the accuracy or utility of the deployed prediction system. Our setting differs in two ways. First, the model response has already been produced, and the human is asked to edit or confirm it in order to create corrective data. Second, a selection policy can fail both by selecting responses that are ultimately confirmed and by skipping responses that would have been edited. Standard deferral methods do not address this edit-discovery objective or the partial feedback induced by selective editor invocation.

B PROOFS

B.1 PROOF OF THEOREM 1

Proof. First, we rewrite the two conditional probabilities in terms of η. For any policy π, by the law of total expectation,

$$\Pr[A = 1, S_\pi = 0] = \mathbb{E}[\Pr[A = 1, S_\pi = 0 \mid Z]] = \mathbb{E}[(1 - \pi(Z))\eta(Z)].$$

Also, $\Pr[S_\pi = 0] = \mathbb{E}[1 - \pi(Z)]$. Therefore,

$$\Pr[A = 1 \mid S_\pi = 0] = \frac{\Pr[A = 1, S_\pi = 0]}{\Pr[S_\pi = 0]} = \frac{\mathbb{E}[(1 - \pi(Z))\eta(Z)]}{\mathbb{E}[1 - \pi(Z)]}.$$

Similarly,

$$\Pr[A = 0, S_\pi = 1] = \mathbb{E}[\Pr[A = 0, S_\pi = 1 \mid Z]] = \mathbb{E}[\pi(Z)(1 - \eta(Z))],$$

and $\Pr[S_\pi = 1] = \mathbb{E}[\pi(Z)]$. Hence,

$$\Pr[A = 0 \mid S_\pi = 1] = \frac{\Pr[A = 0, S_\pi = 1]}{\Pr[S_\pi = 1]} = \frac{\mathbb{E}[\pi(Z)(1 - \eta(Z))]}{\mathbb{E}[\pi(Z)]}.$$

Thus the constraint $\Pr[A = 0 \mid S_\pi = 1] \le \varepsilon$ is equivalent to

$$\mathbb{E}[\pi(Z)(1 - \eta(Z) - \varepsilon)] \le 0.$$

Indeed, if $\mathbb{E}[\pi(Z)] > 0$, this follows by multiplying both sides by $\mathbb{E}[\pi(Z)]$. If $\mathbb{E}[\pi(Z)] = 0$, both sides reduce to $0 \le 0$ under the population convention $0/0 = 0$.

If $\Pr[S_{\pi^\star} = 0] = 0$, then $\pi^\star(z) = 1$ almost surely. In this case the claimed threshold form follows by taking $\alpha^\star = 0$ and $\beta^\star = 0$, with the boundary chosen so that the policy selects almost surely.

It remains to consider the case $\Pr[S_{\pi^\star} = 0] > 0$. Let $\pi^\star$ be an optimal policy. By optimality, for every feasible policy π with $\mathbb{E}[1 - \pi(Z)] > 0$,

$$\Pr[A = 1 \mid S_\pi = 0] \ge \Pr[A = 1 \mid S_{\pi^\star} = 0]\,.$$

Equivalently,

$$\frac{\mathbb{E}[(1 - \pi(Z))\eta(Z)]}{\mathbb{E}[1 - \pi(Z)]} \ge \Pr[A = 1 \mid S_{\pi^\star} = 0]\,.$$

Since $\mathbb{E}[1 - \pi(Z)] > 0$, this is equivalent to

$$\mathbb{E}\left[(1 - \pi(Z))\left(\eta(Z) - \Pr[A = 1 \mid S_{\pi^\star} = 0]\right)\right] \ge 0.$$

For $\pi = \pi^\star$, equality holds:

$$\mathbb{E}\left[(1 - \pi^\star(Z))\left(\eta(Z) - \Pr[A = 1 \mid S_{\pi^\star} = 0]\right)\right] = 0.$$

Therefore, $\pi^\star$ solves the auxiliary linear problem

$$\begin{aligned} \min_{\pi:\mathcal{Z}\to[0,1]} \quad & \mathbb{E}\left[(1 - \pi(Z))\left(\eta(Z) - \Pr[A = 1 \mid S_{\pi^\star} = 0]\right)\right] \\ \text{s.t.} \quad & \mathbb{E}[\pi(Z)(1 - \eta(Z) - \varepsilon)] \le 0, \\ & \mathbb{E}[\pi(Z)c(Z)] \le B. \end{aligned} \tag{14}$$

The policy space in (EDO) already allows $0 \le \pi \le 1$, so no relaxation is introduced here. Since the objective and constraints in (14) are linear in π, the pointwise optimizer is binary except possibly on equality boundaries.

We next verify Slater's condition, after handling two degenerate cases directly. First, if $B = 0$, then $c(Z) > 0$ implies that every feasible policy satisfies $\pi(Z) = 0$ almost surely. The claimed threshold form therefore follows by taking $\alpha^\star = 1$ and $\beta^\star = 0$, with the boundary chosen to remain unselected.

Next, suppose $\Pr[\eta(Z) > 1 - \varepsilon] = 0$. Then $1 - \eta(Z) - \varepsilon \ge 0$ almost surely, and feasibility implies

$$\pi^\star(Z)(1 - \eta(Z) - \varepsilon) = 0 \qquad \text{almost surely.}$$

Hence $\pi^\star$ can select only on the set $\{\eta(Z) = 1 - \varepsilon\}$. The claimed threshold form follows by taking $\alpha^\star = 1 - \varepsilon$ and $\beta^\star = 0$, with the boundary chosen to reproduce $\pi^\star$.

We may therefore assume that $B > 0$ and $\Pr[\eta(Z) > 1 - \varepsilon] > 0$. Since $c(Z) < \infty$, there exists some $M < \infty$ such that the measurable set

$$H_M := \{z : \eta(z) > 1 - \varepsilon,\ c(z) \le M\}$$

has positive probability. Let

$$\bar{\pi}(z) = \delta\, \mathbb{1}\{z \in H_M\},$$

where $\delta \in (0, 1]$ is chosen sufficiently small that $\delta M \Pr[Z \in H_M] < B$. Then

$$\mathbb{E}[\bar{\pi}(Z)(1 - \eta(Z) - \varepsilon)] < 0 \qquad \text{and} \qquad \mathbb{E}[\bar{\pi}(Z)c(Z)] \le \delta M \Pr[Z \in H_M] < B.$$

Thus the problem in (14) satisfies Slater's condition. Viewing π as an element of $L^\infty(P_Z)$, its objective and constraints are continuous linear functionals, and standard convex duality therefore gives strong duality; the KKT conditions are necessary and sufficient for optimality.

For multipliers $\lambda_1, \lambda_2 \ge 0$, define the Lagrangian

$$\begin{aligned}\mathcal{L}(\pi, \lambda_1, \lambda_2) &= \mathbb{E}\left[(1 - \pi(Z))\left(\eta(Z) - \Pr[A = 1 \mid S_{\pi^\star} = 0]\right)\right] \\ &\quad + \lambda_1 \mathbb{E}[\pi(Z)(1 - \eta(Z) - \varepsilon)] + \lambda_2\big(\mathbb{E}[\pi(Z)c(Z)] - B\big) \\ &= \mathbb{E}\left[\eta(Z) - \Pr[A = 1 \mid S_{\pi^\star} = 0]\right] - \lambda_2 B \\ &\quad + \mathbb{E}\Big[\pi(Z)\big(\Pr[A = 1 \mid S_{\pi^\star} = 0] - \eta(Z) + \lambda_1(1 - \eta(Z) - \varepsilon) + \lambda_2 c(Z)\big)\Big].\end{aligned}$$

By the KKT conditions, there exist optimal dual variables $\lambda_1^\star, \lambda_2^\star \ge 0$ such that:

- $\mathbb{E}[\pi^\star(Z)(1 - \eta(Z) - \varepsilon)] \le 0, \qquad \mathbb{E}[\pi^\star(Z)c(Z)] - B \le 0.$
- $\lambda_1^\star, \lambda_2^\star \ge 0.$
- $\lambda_1^\star \mathbb{E}[\pi^\star(Z)(1 - \eta(Z) - \varepsilon)] = 0, \qquad \lambda_2^\star\big(\mathbb{E}[\pi^\star(Z)c(Z)] - B\big) = 0.$
- $\pi^\star \in \arg\min_{0 \le \pi \le 1} \mathcal{L}(\pi, \lambda_1^\star, \lambda_2^\star).$

Now fix $\lambda_1 = \lambda_1^\star$ and $\lambda_2 = \lambda_2^\star$. Since the Lagrangian separates pointwise, minimizing $\mathcal{L}(\pi, \lambda_1^\star, \lambda_2^\star)$ over $0 \le \pi \le 1$ is the same as solving, for almost every z,

$$\min_{0 \le t \le 1} t\big(\Pr[A = 1 \mid S_{\pi^\star} = 0] - \eta(z) + \lambda_1^\star(1 - \eta(z) - \varepsilon) + \lambda_2^\star c(z)\big).$$

Thus the minimizer is

$$t^\star(z) = \begin{cases} 1, & \text{if } \Pr[A = 1 \mid S_{\pi^\star} = 0] - \eta(z) + \lambda_1^\star(1 - \eta(z) - \varepsilon) + \lambda_2^\star c(z) < 0, \\ 0, & \text{if } \Pr[A = 1 \mid S_{\pi^\star} = 0] - \eta(z) + \lambda_1^\star(1 - \eta(z) - \varepsilon) + \lambda_2^\star c(z) > 0, \\ \text{any value in } [0, 1], & \text{if } \Pr[A = 1 \mid S_{\pi^\star} = 0] - \eta(z) + \lambda_1^\star(1 - \eta(z) - \varepsilon) + \lambda_2^\star c(z) = 0. \end{cases}$$

The last case is precisely the threshold boundary and can be resolved by randomization if needed.

Finally, since $\lambda_1^\star, \lambda_2^\star \ge 0$,

$$\begin{aligned}&\Pr[A = 1 \mid S_{\pi^\star} = 0] - \eta(z) + \lambda_1^\star(1 - \eta(z) - \varepsilon) + \lambda_2^\star c(z) < 0 \\ &\iff (1 + \lambda_1^\star)\eta(z) > \Pr[A = 1 \mid S_{\pi^\star} = 0] + \lambda_1^\star(1 - \varepsilon) + \lambda_2^\star c(z) \\ &\iff \eta(z) > \frac{\Pr[A = 1 \mid S_{\pi^\star} = 0] + \lambda_1^\star(1 - \varepsilon)}{1 + \lambda_1^\star} + \frac{\lambda_2^\star}{1 + \lambda_1^\star} c(z).\end{aligned}$$

Define

$$\alpha^\star := \frac{\Pr[A = 1 \mid S_{\pi^\star} = 0] + \lambda_1^\star(1 - \varepsilon)}{1 + \lambda_1^\star}, \qquad \beta^\star := \frac{\lambda_2^\star}{1 + \lambda_1^\star}.$$

Since $0 \le \varepsilon < 1$, $\Pr[A = 1 \mid S_{\pi^\star} = 0] \ge 0$, and $\lambda_1^\star, \lambda_2^\star \ge 0$, we have $\alpha^\star, \beta^\star \ge 0$. Therefore,

$$\pi^\star(z) = \mathbb{1}\left\{\eta(z) > \alpha^\star + \beta^\star c(z)\right\},$$

with possible randomization on the boundary $\eta(z) = \alpha^\star + \beta^\star c(z)$. This proves the claimed form of the optimal policy. □

B.2 Proof of Theorem 2

Proof. The budget constraint holds by the definition of β_t. Indeed, from (8) and the EDIT threshold rule (7),

$$\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}} c(Z_{t,i}) = \sum_{i=1}^{n_t} \mathbb{1}\left\{\widetilde{\eta}_{\theta_t}(Z_{t,i}) > \alpha_t + \beta_t c(Z_{t,i})\right\} c(Z_{t,i}) \le B_t.$$

It remains to prove the average per-round Type-I constraint guarantee. Since $\gamma_t = \gamma$, the update rule gives

$$\alpha_{t+1} = \max\left\{0, \alpha_t + \gamma\left(\frac{\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}(1 - A_{t,i})}{1 \vee \sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}} - \varepsilon\right)\right\}.$$

Therefore,

$$\alpha_{t+1} \ge \alpha_t + \gamma\left(\frac{\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}(1 - A_{t,i})}{1 \vee \sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}} - \varepsilon\right) \implies \frac{\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}(1 - A_{t,i})}{1 \vee \sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}} - \varepsilon \le \frac{\alpha_{t+1} - \alpha_t}{\gamma}.$$

Summing over $t = 1, \dots, T$, and using $\alpha_1 = 0$, gives

$$\sum_{t=1}^{T}\left(\frac{\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}(1 - A_{t,i})}{1 \vee \sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}} - \varepsilon\right) \le \sum_{t=1}^{T} \frac{\alpha_{t+1} - \alpha_t}{\gamma} = \frac{\alpha_{T+1} - \alpha_1}{\gamma} = \frac{\alpha_{T+1}}{\gamma}.$$

We now show that $\alpha_{T+1} \le 1 + \gamma$. To do so, we prove the stronger claim that $\alpha_t \le 1 + \gamma$ for all $t = 1, \dots, T+1$. The proof is by induction on t. Since $\alpha_1 = 0$, the claim holds initially. If $\alpha_t \le 1$, then the selected confirmation ratio is at most one, and therefore

$$\alpha_{t+1} \le \alpha_t + \gamma \le 1 + \gamma.$$

If $\alpha_t > 1$, then $\widetilde{\eta}_{\theta_t}(Z_{t,i}) \in [0, 1]$, $\beta_t \ge 0$, and $c(Z_{t,i}) > 0$ imply that no item can satisfy $\widetilde{\eta}_{\theta_t}(Z_{t,i}) > \alpha_t + \beta_t c(Z_{t,i})$. Hence $\pi_{t,i}^{\mathrm{EDIT}} = 0$ for all i. Therefore,

$$\frac{\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}(1 - A_{t,i})}{1 \vee \sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}} = 0 \implies \alpha_{t+1} = \max\{0, \alpha_t - \gamma\varepsilon\} \le \alpha_t.$$

Thus $\alpha_t \le 1 + \gamma$ for all t, and in particular $\alpha_{T+1} \le 1 + \gamma$. Consequently,

$$\widehat{\mathrm{Err}}_{\mathrm{I}}(\pi^{\mathrm{EDIT}}) - \varepsilon \le \frac{1 + \gamma}{\gamma T}.$$

This proves the theorem. □

C EDIT Pseudocode

Algorithm 1: EDIT: Edit Discovery via Iterative Thresholding

Require: Batches $\{\mathcal{D}_t\}_{t=1}^T$, budgets $\{B_t\}_{t=1}^T$, Type-I error tolerance ε, threshold stepsizes $\{\gamma_t\}_{t=1}^T$, score $\widetilde{\eta}_\theta$, score stepsizes $\{\xi_t\}_{t=1}^T$, initial parameter θ_1

$\alpha_1 \leftarrow 0$;

for $t = 1, \dots, T$ **do**

compute $\widetilde{\eta}_{\theta_t}(Z_{t,i}), \quad i = 1, \dots, n_t$;

$\beta_t \leftarrow \inf\left\{\beta \ge 0 : \sum_{i=1}^{n_t} c(Z_{t,i}) \mathbb{1}\{\widetilde{\eta}_{\theta_t}(Z_{t,i}) > \alpha_t + \beta c(Z_{t,i})\} \le B_t\right\}$;

$\pi_{t,i}^{\mathrm{EDIT}} \leftarrow \mathbb{1}\{\widetilde{\eta}_{\theta_t}(Z_{t,i}) > \alpha_t + \beta_t c(Z_{t,i})\}, \quad i = 1, \dots, n_t$;

observe $A_{t,i}$ for all i with $\pi_{t,i}^{\mathrm{EDIT}} = 1$;

$\alpha_{t+1} \leftarrow \max\left\{0, \alpha_t + \gamma_t\left(\frac{\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}(1 - A_{t,i})}{1 \vee \sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}} - \varepsilon\right)\right\}$;

$\theta_{t+1} \leftarrow \theta_t - \xi_t \frac{\sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}} \left[\frac{\widetilde{\eta}_{\theta_t}(Z_{t,i}) - A_{t,i}}{\widetilde{\eta}_{\theta_t}(Z_{t,i})(1 - \widetilde{\eta}_{\theta_t}(Z_{t,i}))}\right] \nabla_{\theta_t} \widetilde{\eta}_{\theta_t}(Z_{t,i})}{1 \vee \sum_{i=1}^{n_t} \pi_{t,i}^{\mathrm{EDIT}}}$;

Table 1: Type-I error tolerance ε for each model–dataset pair. The same Type-I error tolerance is used across all EDIT score variants.

Dataset	Llama-3.3-70B	Qwen3-32B
MATH-500	0.505	0.615
MMLU-Pro	0.485	0.425
PopQA	0.175	0.000
GPQA	0.505	0.375

Table 2: Accuracy of the response models on the benchmark interactions.

Dataset	Llama-3.3-70B	Qwen3-32B
MATH-500	0.730	0.794
MMLU-Pro	0.526	0.558
PopQA	0.444	0.220
GPQA	0.540	0.458

Table 3: Hyperparameters shared across the main online experiments.

T	n_t	B_t	$c(z)$	γ	ξ	α_1	θ_0	d
250	10	5	1	0.1	1.0	0	$0.1 \times \mathbf{1}$	384

D Additional Experiments

D.1 Additional Tables and Figures

Table 1 reports the Type-I error tolerances used for each model–dataset pair, and Table 2 reports the corresponding base-model accuracies on the benchmark interactions. Table 3 summarizes the hyperparameters shared across the main online experiments.

D.2 Human Audit of Benchmark-Derived Edit Labels

The benchmark experiments use task correctness as a scalable proxy for whether a response requires modification, so we directly validate this assumption on 200 held-out prompt–response pairs: 50 examples each from MATH-500, MMLU-Pro, PopQA, and GPQA. A human annotator evaluates every example with the interface in Figure 3 and assigns either *confirm as-is* or *needs edit*; the interface exposes the user prompt, candidate response, and reference answer and optionally records a reason category. GPT-5.6 Sol independently annotates the same 200 examples using the same binary criterion and the common evaluation prompt reported in Appendix D.4.4. We treat the human labels as the primary validation target and use GPT-5.6 Sol only as an additional consistency check.

The benchmark-derived labels closely track the human judgments at both the aggregate and example levels. Across all 200 examples, benchmark and human labels agree on 96.5% of interactions, with Cohen's $\kappa = 0.930$ and edit-class $F_1 = 0.963$; their overall edit rates are also nearly identical, 0.470 for the benchmark proxy versus 0.465 for the human annotator. Dataset-level benchmark–human agreement remains between 0.960 and 0.980, with $\kappa > 0.916$ for every dataset. GPT-5.6 Sol produces an edit rate of 0.460, agrees with the benchmark in 0.980 of cases ($\kappa = 0.960$), and agrees with the human annotator in 0.975 of cases ($\kappa = 0.950$). Table 4 gives the complete dataset-level breakdown.

Taken together, the audit provides direct support for using benchmark correctness as the edit indicator in the main experiments rather than merely matching edit prevalence by coincidence. The benchmark proxy disagrees with the human annotator on only 7 of 200 audited interactions, while its per-dataset edit rates stay close to the human rates and its example-level agreement is consistently high. The independent GPT-5.6 Sol annotations lead to the same qualitative conclusion, which further suggests that the benchmark labels capture a stable notion of substantive edit need.

D.3 Operating-Point Trade-offs and Complementary Metrics.

The main experiments use one operating point for each model–dataset pair. To examine whether the observed behavior is specific to that choice, we additionally sweep the Type-I error tolerance $\varepsilon \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$ and the per-round editor budget $B \in \{2, 3, 4, 5, 7\}$ for EDIT-REP. For each configuration, we repeat the online experiment over five realized streams and report the mean together with 95% confidence intervals. The LLM-Select marker corresponds to its main $B = 5$ run and is included as a reference operating point.

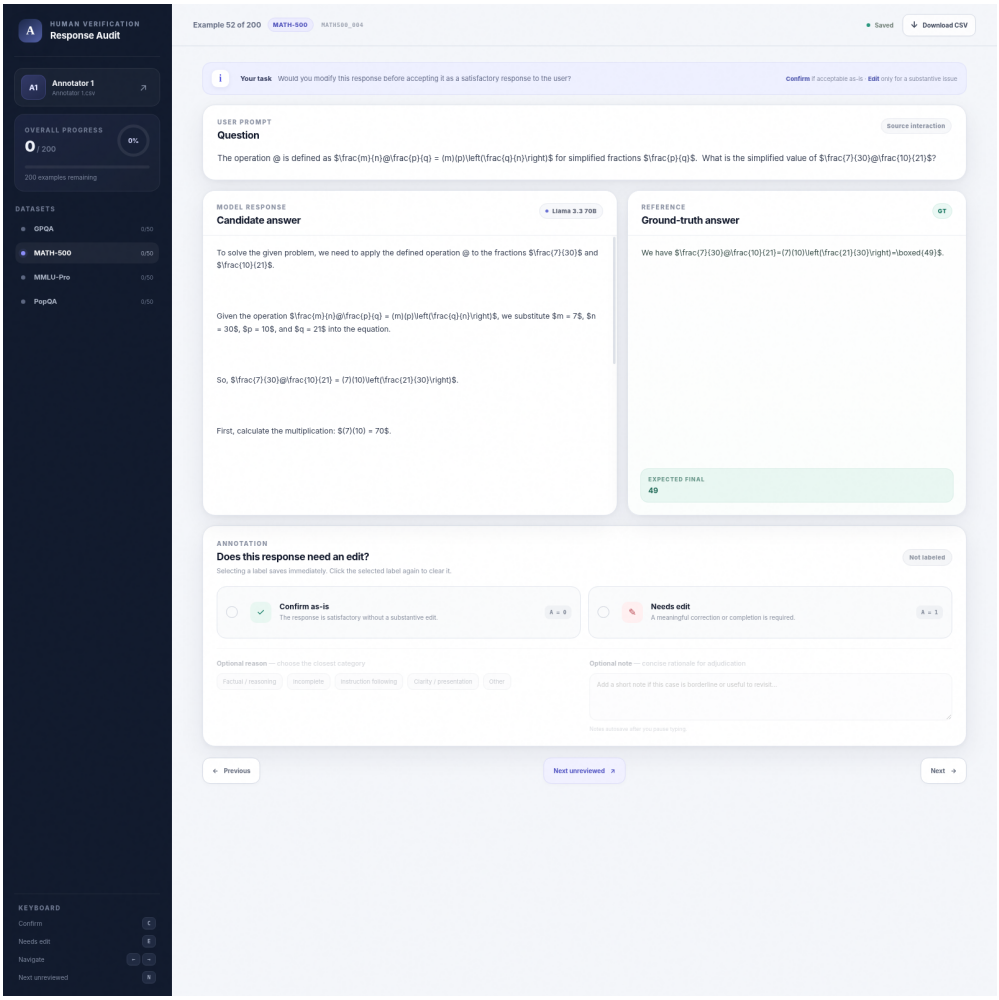

Figure 3: Human-audit interface. The human annotator inspects the user prompt, candidate response, and reference answer, then decides whether the response can be confirmed as-is or requires a substantive edit. Optional reason categories support later inspection of disagreement cases.

Table 4: Validation of the benchmark-derived edit labels. The human annotator is the primary validation target; GPT-5.6 Sol is an independent model annotator. "B–H agree", κ_{BH}, and $F_{1,BH}$ compare the benchmark with the human labels. "H–GPT agree" reports agreement between the human annotator and GPT-5.6 Sol. The positive class is "needs edit".

Dataset	N	Benchmark edit	Human edit	GPT-5.6 Sol edit	B–H agree	κ_{BH}	$F_{1,BH}$	H–GPT agree
MATH-500	50	0.280	0.300	0.260	0.980	0.951	0.966	0.960
MMLU-Pro	50	0.460	0.420	0.440	0.960	0.919	0.955	0.980
PopQA	50	0.560	0.520	0.540	0.960	0.920	0.963	0.980
GPQA	50	0.580	0.620	0.600	0.960	0.917	0.967	0.980
Overall	**200**	**0.470**	**0.465**	**0.460**	**0.965**	**0.930**	**0.963**	**0.975**

Figure 4 makes the operating-point trade-off explicit. As the policy is allowed to select more broadly, it can reduce the edit rate among skipped interactions, at the cost of a larger confirmation rate among selected interactions. Across the four datasets, the resulting curves show that the relative advantage of informative EDIT-REP ranking is not tied to a single choice of ε or B. Figure 5 presents the same sweep in terms of the amount of editor effort actually used and the fraction of editable interactions

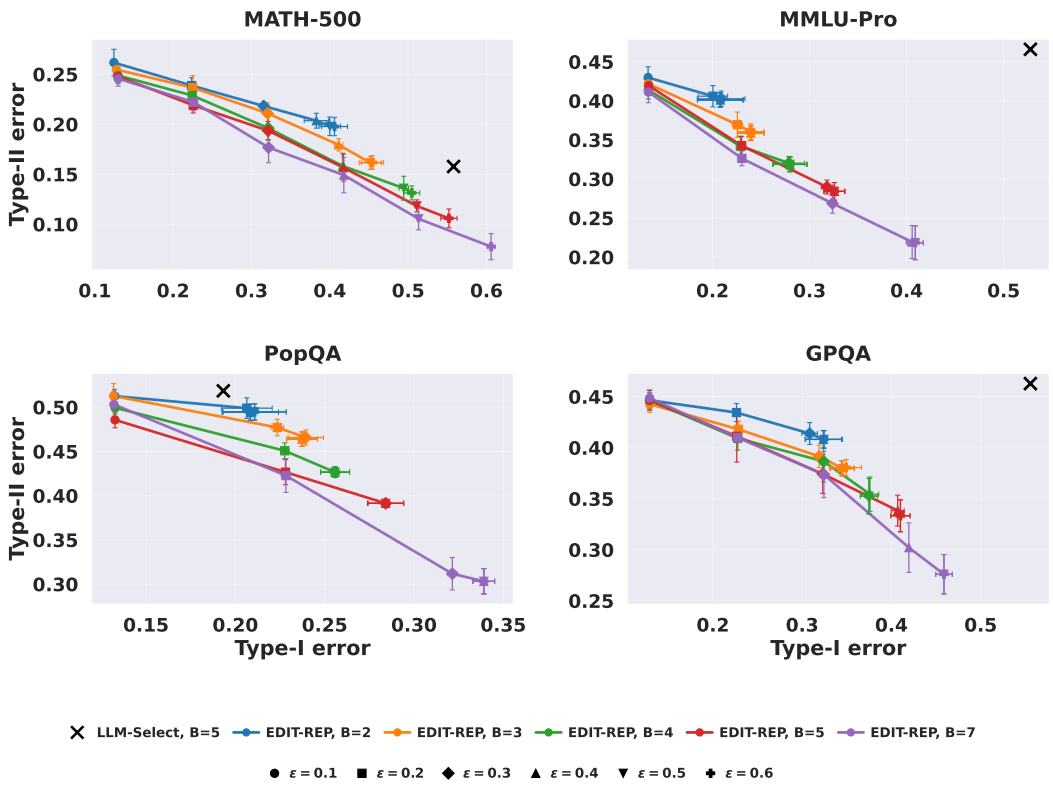

Figure 4: Type-I–Type-II trade-offs for EDIT-REP. Panels correspond to MATH-500, MMLU-Pro, PopQA, and GPQA. Colored curves vary the editor budget B, while marker shapes denote the Type-I error tolerance ε. Error bars show 95% confidence intervals across five realized streams. The black cross shows the existing LLM-SELECT run with $B = 5$. Lower values on both axes are better.

discovered. Increasing the available editor budget generally increases edit recall, while the different values of ε determine how aggressively that budget is used.

To complement the conditional Type-I and Type-II errors used in the main text, Table 5 reports three discovery-centric quantities on the existing main Llama-3.3-70B-Instruct and Qwen3-32B runs: edit recall, the absolute number of missed edits in the realized stream, and edit precision among selected interactions. These metrics provide a complementary view of how much corrective data each policy recovers at its realized editor operating point. Across both response models, EDIT-REP attains the highest edit recall and fewest missed edits on most dataset–model pairs, while EDIT-LLM achieves the strongest edit precision in several settings.

D.4 LIST OF LLM PROMPTS

The experiments and audit use four prompt families. The first generates the candidate response $\widetilde{Y}$ for each benchmark question. The second is used by LLM-SELECT to make one budgeted selection decision for an entire batch, the third is used by EDIT-LLM to produce one scalar edit score for each prompt–response pair, and the fourth is the common evaluation prompt used by GPT-5.6 Sol in the audit of benchmark-derived edit labels.

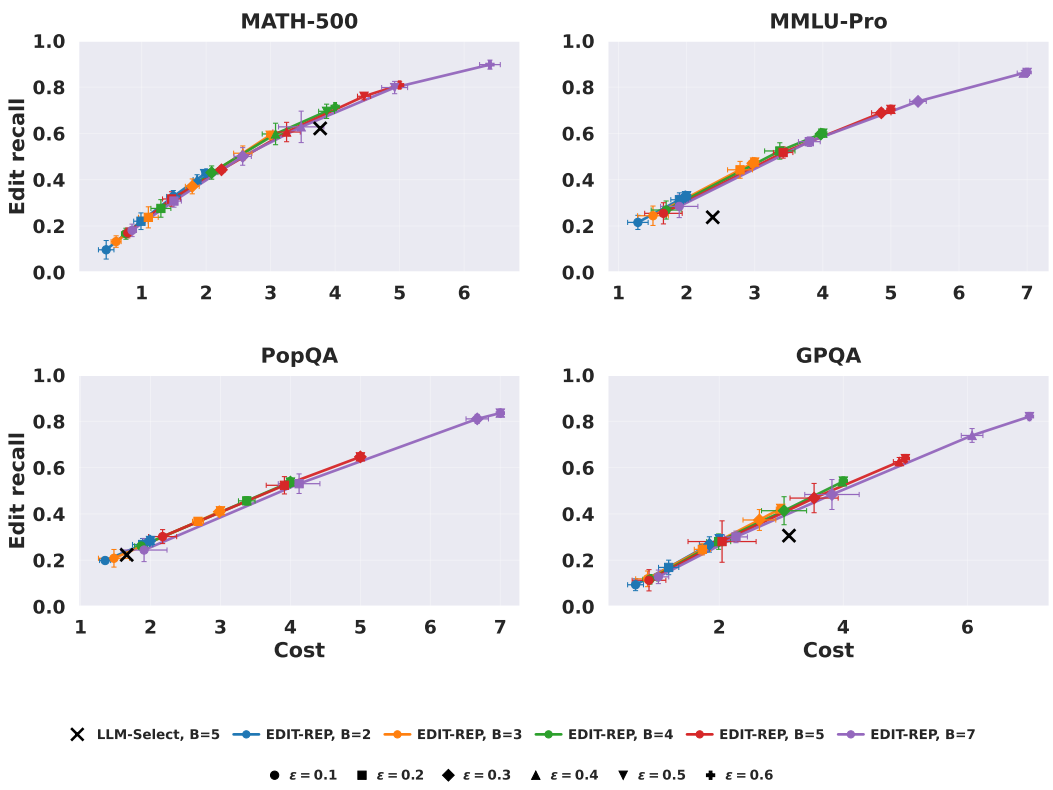

Figure 5: Editor-cost–edit-recall trade-offs for EDIT-REP. Panels correspond to MATH-500, MMLU-Pro, PopQA, and GPQA. Colored curves vary the editor budget B, while marker shapes denote the Type-I error tolerance ε. Error bars show 95% confidence intervals across five realized streams. The black cross shows the existing LLM-SELECT run with $B = 5$. Higher edit recall is better.

D.4.1 CANDIDATE-RESPONSE GENERATION PROMPTS

MATH-500

System instruction.

```
You are a careful mathematical problem solver.
Hard requirements:
- Solve the problem accurately.
- End with exactly one final line in this format:
#### <answer>
The final line must contain only the marker #### followed by the
final answer.
```

User prompt.

```
Solve the following math problem.
End with exactly one final line in this format:  #### <answer>
Problem:
{question}
```

MMLU-Pro

System instruction.

```
You are a careful multiple-choice reasoning system.
Hard requirements:
- Solve the question and choose exactly one option.
```

Table 5: Complementary metrics on the main Llama-3.3-70B-Instruct and Qwen3-32B runs. Edit recall is the fraction of editable interactions that are selected; missed edits counts editable interactions that are skipped; and edit precision is the fraction of selected interactions that require an edit. Within each model–dataset block, the best value is shown in **bold** and the second-best is underlined.

Dataset	Model	Method	Edit recall ↑	Missed edits ↓	Edit precision ↑
MATH-500	Llama-3.3-70B-Instruct	EDIT-REP	**0.784**	**143**	0.467
		EDIT-LLM	0.691	205	**0.473**
		EDIT-RAND	0.151	563	0.253
		LLM-SELECT	0.621	251	0.437
	Qwen3-32B	EDIT-REP	0.810	96	0.366
		EDIT-LLM	**0.939**	**31**	**0.407**
		EDIT-RAND	0.190	410	0.192
		LLM-SELECT	0.765	119	0.342
MMLU-Pro	Llama-3.3-70B-Instruct	EDIT-REP	**0.703**	**346**	**0.656**
		EDIT-LLM	0.292	826	0.469
		EDIT-RAND	0.401	699	0.466
		LLM-SELECT	0.238	889	0.465
	Qwen3-32B	EDIT-REP	**0.663**	**385**	**0.609**
		EDIT-LLM	0.462	614	0.552
		EDIT-RAND	0.242	866	0.455
		LLM-SELECT	0.445	634	0.530
PopQA	Llama-3.3-70B-Instruct	EDIT-REP	**0.411**	**824**	0.746
		EDIT-LLM	0.284	1002	**0.756**
		EDIT-RAND	0.109	1246	0.571
		LLM-SELECT	0.223	1087	0.752
	Qwen3-32B	EDIT-REP	**0.394**	**1177**	0.915
		EDIT-LLM	0.215	1526	**0.961**
		EDIT-RAND	0.142	1667	0.791
		LLM-SELECT	0.260	1438	0.930
GPQA	Llama-3.3-70B-Instruct	EDIT-REP	**0.644**	**407**	**0.589**
		EDIT-LLM	0.615	440	0.576
		EDIT-RAND	0.472	604	0.464
		LLM-SELECT	0.306	793	0.448
	Qwen3-32B	EDIT-REP	**0.627**	**513**	0.690
		EDIT-LLM	0.625	516	**0.696**
		EDIT-RAND	0.353	890	0.561
		LLM-SELECT	0.360	880	0.580

```
- The final answer must be only the option letter.
- End with exactly one final line in this format:
#### <letter>
The final line must contain only the marker #### followed by a single
letter such as A, B, C, D, E, F, G, H, I, or J.
```

User prompt.

```
Solve the following MMLU-Pro multiple-choice question.
Choose the single best option.  Think carefully if needed, but do not
include reasoning in the output.
End with exactly one final line in this format:  #### <letter>
{question}
```

PopQA

System instruction.

```
You are a careful open-domain factual question-answering system.
Hard requirements:
- Answer the question accurately.
- Provide only the final short answer, not a sentence or explanation.
- The answer should be an entity, title, place, date, number, or short
```

phrase.
- End with exactly one final line in this format:
<answer>
The final line must contain only the marker #### followed by the answer.

User prompt.
Answer the following open-domain factual question from PopQA.
Give only the final short answer; do not include explanation, reasoning, or extra text.
The answer should be an entity, title, place, date, number, or short phrase.
End with exactly one final line in this format: #### <answer>
Question:
{question}

GPQA

System instruction.
You are a careful graduate-level science reasoning system.
Hard requirements:
- Solve the multiple-choice question and choose exactly one option.
- The final answer must be only the option letter.
- End with exactly one final line in this format:
<letter>
The final line must contain only the marker #### followed by a single letter: A, B, C, or D.

User prompt.
Solve the following GPQA graduate-level science multiple-choice question.
Choose the single best option. Think carefully if needed.
End with exactly one final line in this format: #### <letter>
{question}

D.4.2 LLM-SELECT BATCH-SELECTION PROMPTS

Common prompt

System instruction.
You are an expert edit-selection system. Your only task is to choose which model-generated answers should be sent for human correction. Return only valid JSON. Do not solve the questions. Do not include explanations.

User-prompt scaffold.
You are selecting examples to send to a human editor. You will receive a batch of model-generated answers. Select the items most likely to contain an error and therefore most worth sending to a human corrector. Respect the total budget {budget}; each item has a listed cost. Select any subset whose total cost is at most the budget. Use only the question/problem and the model answer. Do not use or assume the gold answer. You may select zero items if none appear worth human correction. Return only valid JSON in exactly this format: {"selected_indices": [0, 3, 4]}. Do not include explanations, markdown, or text outside the JSON object.
Batch:
{items}

MATH-500

```
The batch contains model-generated mathematics answers. Prefer
examples with suspicious reasoning, arithmetic mistakes, missing
final answers, format violations, or unsupported conclusions. Each
item is formatted with an index, cost, problem, and model answer.
```

MMLU-Pro

```
Each item is a multi-option academic or professional reasoning
question. The desired response is one option letter such as A, B,
C, ..., or J. Prefer examples with questionable reasoning, no valid
option, multiple conflicting options, unsupported certainty, or a
final answer inconsistent with the question and choices. Each item
contains an index, cost, question and options, and model answer.
```

PopQA

```
The desired response is a short factual answer: an entity, title,
place, date, number, or short phrase. Prefer likely factual errors,
missing answers, multiple conflicting answers, unsupported hedging,
non-short-answer formatting, or responses too vague to match a valid
PopQA alias. Each item contains an index, cost, question, and model
answer.
```

GPQA

```
Each item is a graduate-level science multiple-choice question
with four options. The desired response is one option letter: A,
B, C, or D. Prefer questionable scientific reasoning, no valid
option, multiple conflicting options, unsupported certainty, or a
final answer inconsistent with the question and choices. Each item
contains an index, cost, question and options, and model answer.
```

D.4.3 EDIT-LLM EDIT-SCORE PROMPTS

Common prompt

System instruction.

```
You are an expert editor of model-generated answers. Estimate
whether a human editor would need to modify the model answer. Return
only valid JSON. Do not include explanations, markdown, or extra
text.
```

User-prompt scaffold.

```
Estimate the posterior edit probability for this prompt--response
pair. The score is a probability in [0, 1] that a human editor would
modify the model answer. A high score means that correction is
likely to be required; a low score means that the answer is likely
to be confirmed without modification. Use only the question/problem
and model answer below. Do not assume access to the gold answer.
Return only valid JSON in exactly this format: {"score": 0.73}.
```

MATH-500

```
Treat mathematical errors, unsupported conclusions, missing final
answers, ambiguity, and formatting problems as evidence that human
modification may be required.
Problem:
{question}

Model answer:
{model_answer}
```

MMLU-Pro

```
The expected response is one option letter such as A, B, C, ..., or
J. Treat a likely wrong option, an invalid option letter, ambiguity,
unsupported reasoning, missing final answers, multiple conflicting
choices, or answer text without a clear option letter as evidence
that correction may be required.
Question and options:
{question}

Model answer:
{model_answer}
```

PopQA

```
The expected response is a concise entity, title, place, date, number,
or short phrase.  Treat likely factual errors, missing answers,
excessive vagueness or specificity, malformed output, explanations,
multiple conflicting answers, unsupported hedging, and text unlikely
to match a valid answer alias as evidence that correction may be
required.
Question:
{question}

Model answer:
{model_answer}
```

GPQA

```
The expected response is one option letter:  A, B, C, or D. Treat
a likely wrong option, an invalid letter, ambiguity, unsupported
scientific reasoning, missing final answers, multiple conflicting
choices, or answer text without a clear option letter as evidence
that correction may be required.
Question and options:
{question}

Model answer:
{model_answer}
```

D.4.4 GPT-5.6 SOL HUMAN-AUDIT PROMPT

Common prompt

```
You are an independent editor of model responses.  For each example,
decide whether the response would need to be modified before being
accepted as a satisfactory answer to the user.
Assign:
- A = 0 if the response is acceptable as-is.
- A = 1 if the response genuinely requires editing before acceptance.
Judge the response based on correctness, reasoning quality,
completeness, instruction following, relevance, and clarity.  Be
conservative with A = 1:  minor stylistic issues, verbosity, or
wording that could be improved should not count as requiring an edit
if the response is still satisfactory.
Use the reference or ground-truth answer as supporting evidence, not
as a strict string-matching criterion.  Accept equivalent answers,
aliases, and reasonable interpretations of ambiguous questions.
For reasoning and mathematical tasks, evaluate the full response
rather than only the final answer.  A correct final answer may still
require editing if the reasoning contains substantive errors or
contradictions.
The key question is:  ''Would you modify this response before
accepting it as a satisfactory response to the user?''
```

D.5 ADDITIONAL EXPERIMENTS WITH QWEN3-32B

To test whether the main conclusions depend on the primary response-model family, we repeat the main experiment with Qwen3-32B (Yang et al., 2025). The response model, EDIT-LLM scorer, and LLM-SELECT selector are changed consistently to Qwen3-32B; the online protocol and metric computation remain otherwise unchanged.

Figure 6 shows that the main qualitative trends persist with Qwen3-32B. As in the main experiments, EDIT-REP generally achieves lower Type-II error than LLM-SELECT across the four datasets, with particularly clear improvements on MMLU-Pro and GPQA. This suggests that the explicit decomposition into an edit score, online Type-I feedback, and budget control remains effective when the underlying response model is changed. The comparison among the EDIT variants again highlights the importance of score quality. EDIT-REP and EDIT-LLM substantially outperform EDIT-RAND on MATH-500, MMLU-Pro, and PopQA, while their relative ordering varies across datasets. Overall, these results show that the benefits of informative edit scores, together with the EDIT controller, are not specific to the underlying response model used in our main experiments.

D.6 RESPONSE-LENGTH-AWARE EDITOR COSTS

Unit editor cost abstracts away an important practical asymmetry: checking a long derivation usually requires more editor effort than validating a short factual response. We therefore define the response-length cost

$$c_{\text{len}}(z) = \text{clip}\left(\frac{L(\widetilde{Y})}{\text{median}_{z'}\, L(\widetilde{Y}')}, 0.25, 4.0\right), \tag{15}$$

where $L(\widetilde{Y})$ is the response word count and the median is computed within the corresponding generation. We retain $B_t = 5$, so the number of examples that fit within the budget depends on their individual editor costs. The Type-I error tolerances used in these experiments are reported in Table 6.

Figure 7 shows the resulting cost distributions. MATH-500 has a pronounced right tail because some responses contain long derivations, whereas PopQA is more concentrated around unit cost, with a smaller number of long-form outliers. This variation illustrates why assigning the same editor cost to every interaction may not accurately reflect the editor effort required to evaluate different responses.

Figure 8 shows that the cost-aware structure of EDIT remains effective when editor effort varies across interactions. On MATH-500, both semantically informed EDIT variants substantially reduce

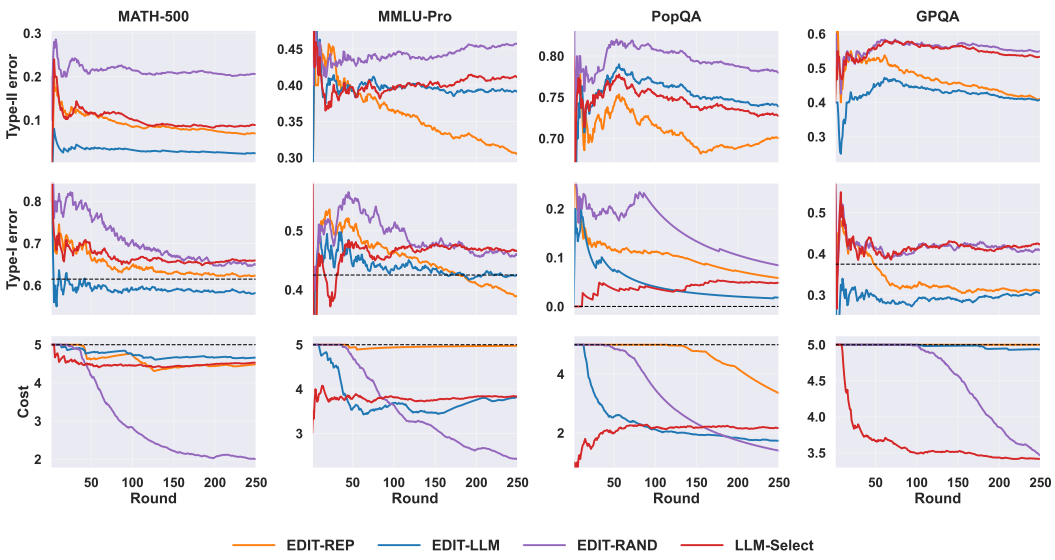

Figure 6: Main results with Qwen3-32B. Columns correspond to MATH-500, MMLU-Pro, PopQA, and GPQA; rows show running-average Type-II error, Type-I error, and average editor cost. Dashed lines indicate the configured Type-I error tolerance and per-round editor budget.

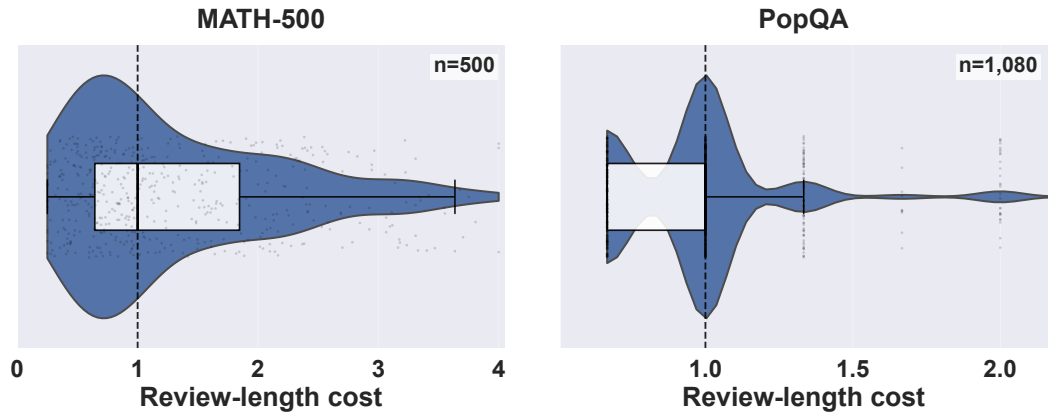

Figure 7: Distribution of normalized response-length editor costs. The violin, box, and point layers show the empirical cost distributions for MATH-500 (left) and PopQA (right). The dashed vertical line marks unit cost, corresponding to the median response length within the relevant generation; costs are clipped to the interval $[0.25, 4.0]$.

Type-II error relative to the random-score control. On GPQA, EDIT-REP achieves the strongest late-stage discovery performance while remaining below the displayed Type-I error tolerance. Although LLM-SELECT often spends less, that apparent saving comes with more missed edits. The cost term is therefore consequential rather than cosmetic: EDIT directs a fixed amount of editor resources toward interactions with the strongest estimated correction value after accounting for editor burden.

D.7 ADDITIONAL EXPERIMENTS UNDER DISTRIBUTION SHIFT

The main experiments randomly mix examples from a fixed benchmark pool. We additionally consider a nonstationary setting in which the distribution of the stream changes over time while the policy is not reset. For MMLU-Pro, the stream moves through `STEM`, `life/health`, `humanities/social science`, and `law/other subject` groups. For PopQA, it moves from `head` entities to `middle-popularity` and then `long-tail` entities. All methods receive the same sequence of

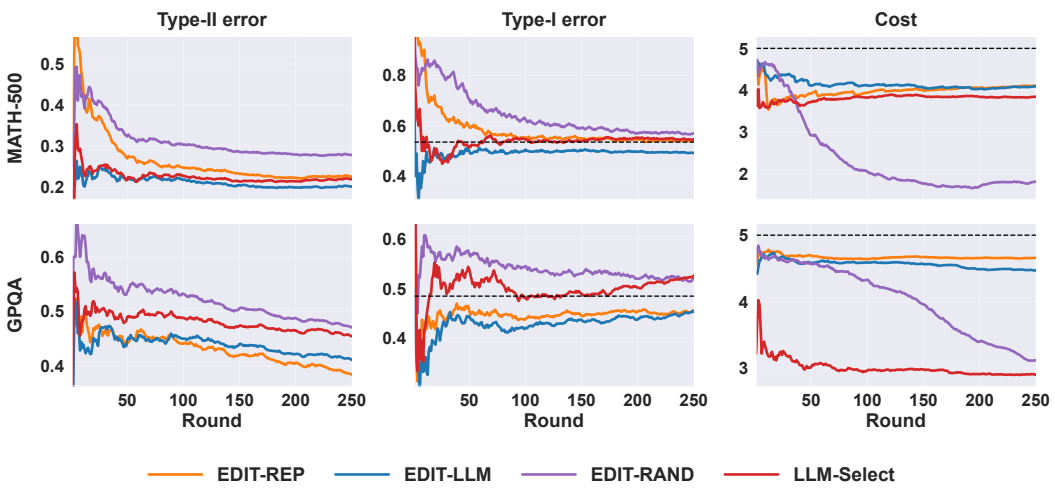

Figure 8: **Edit discovery with response-length-dependent editor cost.** Results use Llama-3.3-70B-Instruct on MATH-500 and GPQA with the cost in Equation (15) and budget $B_t = 5$. Rows report running-average Type-II error, Type-I error, and average editor cost. Dashed lines denote the configured Type-I error tolerance and per-round editor budget.

Table 6: Type-I error tolerance used in the response-length-aware experiments.

Dataset	ε
MATH-500	0.535
GPQA	0.485

Table 7: Response-model accuracy on the distribution-shift streams.

Dataset	Accuracy
MMLU-Pro	0.540
PopQA	0.393

Table 8: Type-I error tolerance used in the distribution-shift experiments.

Dataset	ε
MMLU-Pro	0.445
PopQA	0.100

phases and the same shift boundaries. The response-model accuracies and Type-I error tolerances for these experiments are reported in Tables 7 and 8, respectively.

As Figure 9 shows, EDIT-REP generally attains lower Type-II error than LLM-SELECT while regulating Type-I error more reliably across phase changes. The separation between EDIT-RAND and the two informed scores persists after the shifts, confirming that adaptation of the threshold cannot compensate for an uninformative ranking. Conversely, the stable Type-I trajectories across score variants illustrate the complementary roles predicted by the theory: the score determines which interactions receive review, while the controller determines how selective the policy should be.

D.8 EFFECT OF EMBEDDING-MODEL SIZE

The main experiments use a compact MiniLM encoder to represent each prompt–response pair in EDIT-REP. We additionally study how the capacity of the frozen representation model affects online edit discovery. We replace the default encoder with Qwen3 models of three sizes: Qwen3-0.6B, Qwen3-1.7B, and Qwen3-4B. In all cases, the response model, online linear probe, selection rule, Type-I error tolerance, budget, stream order, and remaining hyperparameters are held fixed. Thus, the only experimental difference is the model used to construct the representation $\phi(Z)$.

Figure 10 shows that increasing the embedding-model size generally improves edit discovery. Qwen3-4B achieves the lowest final Type-II error on all four datasets, with the clearest improvements on PopQA and GPQA. Qwen3-1.7B also consistently improves over Qwen3-0.6B and approaches the performance of Qwen3-4B on MATH-500 and MMLU-Pro. These results indicate that larger representation models provide features that better separate interactions likely to require correction.

The Type-I trajectories remain similar across encoder sizes, showing that the adaptive threshold continues to regulate the confirmation rate independently of the representation model. On MMLU-Pro and GPQA, all three variants use essentially the same editor budget, so the reduction in Type-II error

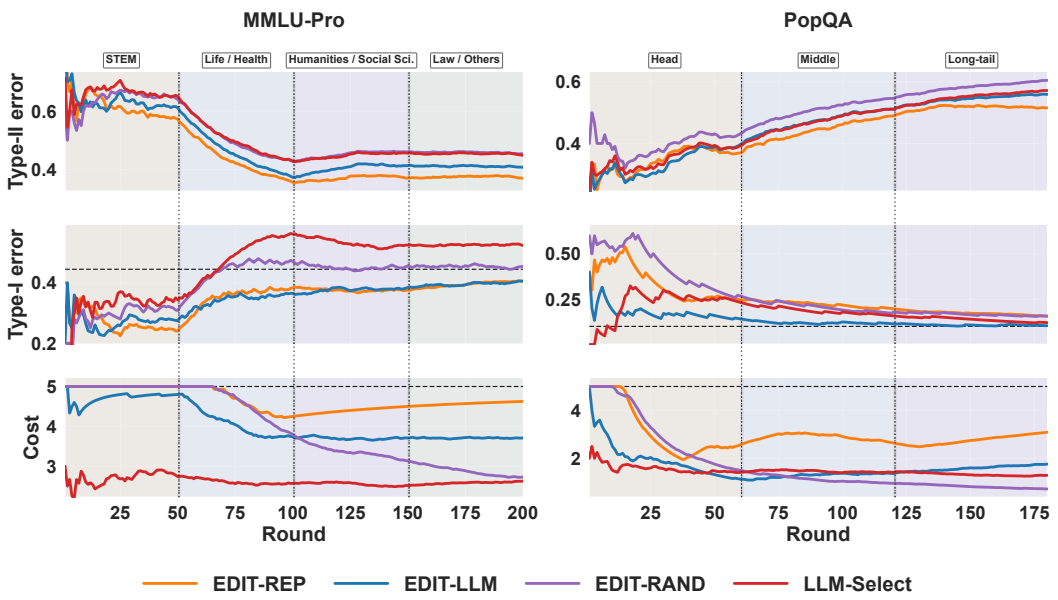

Figure 9: Robustness under structured distribution shift. Vertical dotted lines mark phase boundaries in the MMLU-Pro subject stream and the PopQA popularity stream. Rows report running-average Type-II error, Type-I error, and average editor cost. Dashed horizontal lines indicate the configured Type-I error tolerance and per-round editor budget.

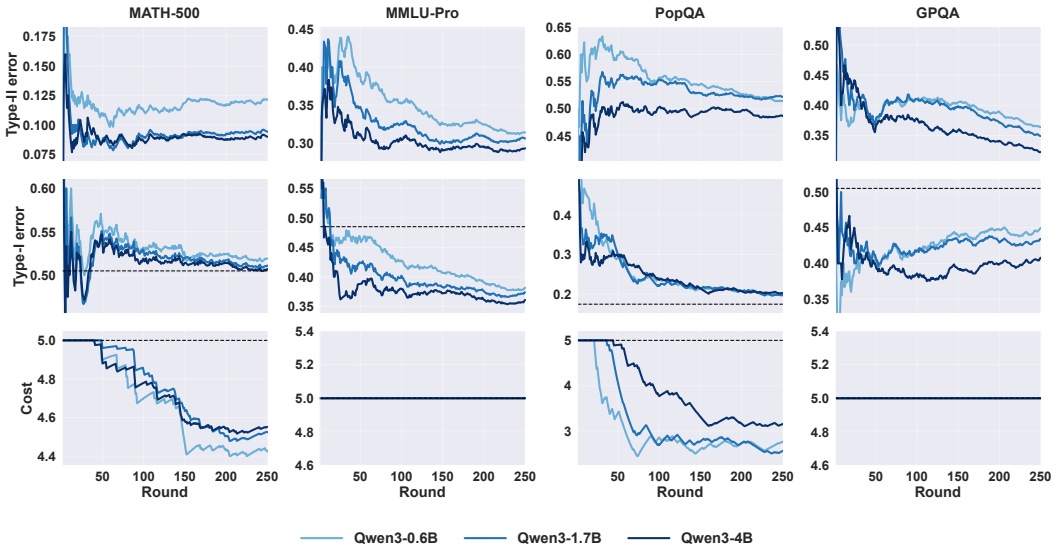

Figure 10: Effect of the embedding-model size on EDIT-REP. Results use Llama-3.3-70B-Instruct as the response model and Qwen3-0.6B, Qwen3-1.7B, or Qwen3-4B as the frozen representation model. Columns correspond to MATH-500, MMLU-Pro, PopQA, and GPQA. Rows report running-average Type-II error, Type-I error, and average editor cost per round. Dashed horizontal lines indicate the configured Type-I error tolerance and per-round editor budget.

can be attributed directly to improved ranking rather than increased expenditure. On MATH-500 and PopQA, the expenditure trajectories differ modestly, but the larger encoders still provide a stronger Type-I–Type-II trade-off. Overall, the experiment shows that embedding capacity affects the efficiency of EDIT-REP, while the online controller still regulates the edit quality and budget use.