SURE: Judge-Aware Safety Update Review for Public-Interest LLM Deployment

Anonymous Authors[1]

Abstract

Public-facing language-model systems are updated continuously, yet a model update can improve an aggregate safety score while reintroducing failures that an earlier version had suppressed. This failure mode is especially consequential for public-interest deployments, where unsafe behavior can affect civic information, institutional services, and population-scale trust. We propose SURE, a judge-aware release-validation protocol that treats safety update review as a paired evaluation problem over a frozen safety suite, a declared judge policy, and an exact paired gate. SURE returns RELEASE, BLOCK, or ESCALATE and emits a compact audit card recording paired counts, judge-policy assumptions, gate sensitivity, and human-reference diagnostics. Across 10 model pairs and 12 single-judge policies, the exact gate rejects in 56/120 cells, while alternative paired gates agree with the default exact gate in 95/96 comparisons; by contrast, single-judge block counts range from 0/12 to 11/12 across model pairs. A blinded human-reference audit of 300 cases with 5 annotators (Fleiss $\kappa = 0.823$) identifies stable-but-uninformative judge policies, showing that trustworthy release validation must audit not only a p-value, but also the measurement policy that supplies the labels.

1. Introduction

Large language models (LLMs) are increasingly embedded in products that mediate access to information, services, and public institutions. Such deployments require more than one-time safety evaluation: model providers routinely rotate to new checkpoints, and a candidate update may be stronger on aggregate capability or even aggregate safety metrics while becoming unsafe on prompts that the previous release handled correctly. For safety review, the relevant question is therefore not only whether the new model has a lower average attack success rate, but whether known failures remain controlled when the model changes.

A natural release-review procedure freezes a versioned safety suite, evaluates both model versions on the same prompts, labels both responses, and examines the discordant outcomes. Cases where the old model is labeled safe and the new model unsafe provide evidence of regression; cases where the old model is unsafe and the new model safe provide evidence of improvement. Paired exact tests such as the one-sided sign or exact McNemar test control false blocking under fixed binary labels and remove prompt-selection confounds that affect aggregate score comparisons (McNemar, 1947; Cochran, 1950). However, the statistical test is the easy part. In modern safety pipelines, the binary labels consumed by a release gate are often produced by LLM-as-judge systems, whose behavior depends on the judge population, prompt, label mapping, aggregation rule, and tie policy (Zheng et al., 2023; Gu et al., 2024).

This paper studies safety update review as a judge-aware evaluation-design problem. We propose SURE, a compact protocol for trustworthy AI deployments that makes the release evaluation an auditable object: a frozen safety suite, paired old/new model responses, a declared judge policy, a paired exact gate, policy-sensitivity analyses, human-referenced judge-informativeness checks, and a SURE Card. The output is three-way: RELEASE when the candidate update shows no paired regression evidence under an informative policy, BLOCK when paired evidence indicates regression, and ESCALATE when the automated call is boundary-sensitive, judge-dependent, or judge-uninformative. SURE is not a replacement for open-ended red teaming. Vulnerability-discovery methods generate and refresh the suite; SURE asks whether a frozen suite of known risks remains controlled across a model update.

Contributions. First, we formalize paired safety update review as an evaluation tuple rather than as a model-pair property, with conditional Type I control only under the fixed judge policy. Second, we separate idealized,

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Submitted to the Trustworthy AI for Good Workshop at ICML 2026. Do not distribute.

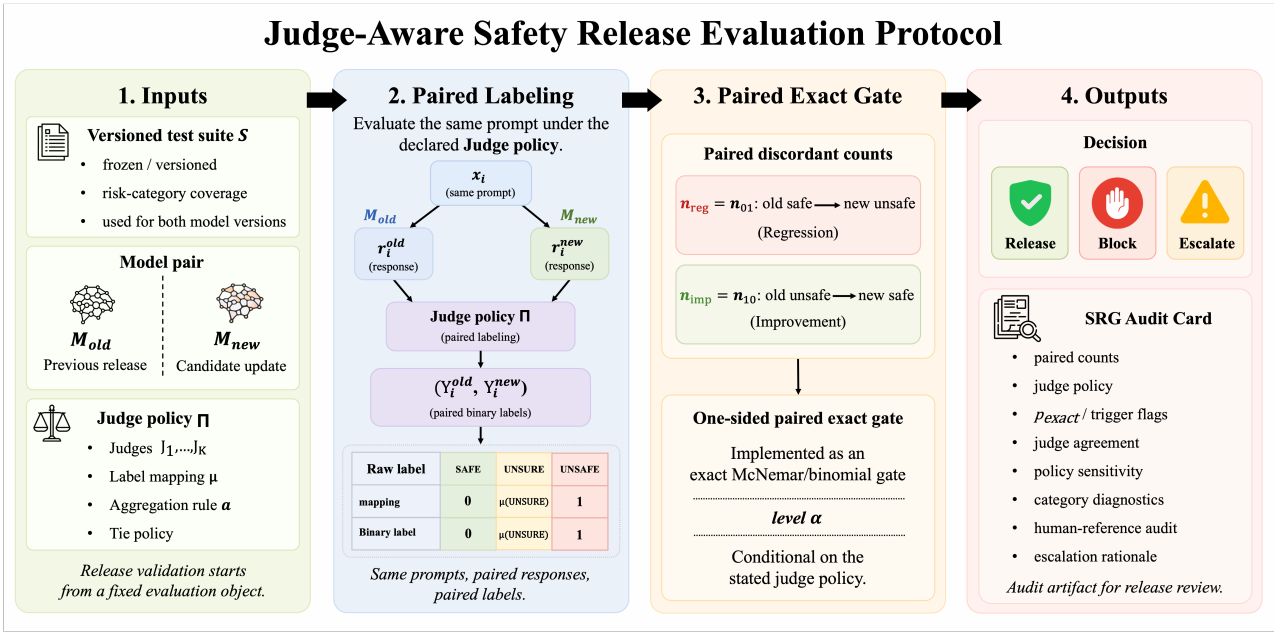

Figure 1. Overview of SURE. A frozen suite is evaluated on old and new model versions, labeled under a declared judge policy, passed through a paired exact gate, and summarized as a RELEASE, BLOCK, or ESCALATE decision with a SURE Card. The card records not only the paired gate result but also judge-policy, policy-sensitivity, and human-reference audit fields.

self-judge, cross-judge, and human-referenced sensitivity regimes, clarifying how evaluator detectability attenuates paired-gate power. Third, we provide an empirical audit showing that paired-gate choice is rarely the bottleneck: alternative gates agree in 95/96 prespecified comparisons, while judge-policy choice changes release calls substantially.

2. Related Work

Red-teaming and threat models. Red-teaming methods for LLMs include manual probing (Ganguli et al., 2022), threat-model and process-oriented analyses (Verma et al., 2025; Feffer et al., 2024), gradient-based attacks (Zou et al., 2023), automated jailbreak generators (Chao et al., 2023; Liu et al., 2024), open-ended adversarial generation (Samvelyan et al., 2024), and end-to-end red-teaming pipelines (Zhou et al., 2025). These methods are essential for discovering failures and refreshing safety suites. SURE addresses a different layer of the release pipeline: given a frozen suite of known risks, it converts paired evaluations of old and new model versions into an auditable release decision.

Benchmarks and aggregate safety scores. Standardized benchmarks such as HarmBench and AIR-Bench consolidate adversarial prompts into versioned risk categories and report attack success rates (Mazeika et al., 2024; Zeng et al., 2025). Aggregate rates answer a population-level benchmark question, but they do not by themselves distinguish systematic shifts in marginal unsafe rate from per-prompt safe-to-unsafe flips. SURE adopts suite versioning but replaces the scalar endpoint with paired discordant counts and a judge-policy audit.

LLM-as-judge reliability. LLM-as-judge systems enable scalable evaluation but introduce policy, bias, and reliability concerns (Zheng et al., 2023; Gu et al., 2024). SURE treats the judge policy as part of the evaluation tuple rather than hidden preprocessing. Our human-reference audit is connected to classical inter-rater agreement and noisy-annotator analysis (Fleiss, 1971; Dawid and Skene, 1979), but the protocol does not infer a latent ground-truth judge; it records whether the declared measurement policy is informative enough to support an automated release call.

3. Judge-Aware Paired Safety Testing

A paired update review is well-defined only after the evaluation policy is fixed. Holding the model pair, suite, and gate fixed while changing the judge policy can change both the paired labels and the release decision.

3.1. Evaluation tuple

Let $S = \{x_i\}_{i=1}^{n}$ be a frozen safety suite and let M_{old} and M_{new} be the compared model versions. For prompt x_i, the

paired responses are

$$r_i^{\text{old}} = M_{\text{old}}(x_i), \qquad r_i^{\text{new}} = M_{\text{new}}(x_i). \tag{1}$$

Definition 1 (Evaluation tuple)**.** A safety update review is specified by

$$T = (S, M_{\text{old}}, M_{\text{new}}, \Pi, G, \alpha), \tag{2}$$

where S is the frozen suite, Π is the declared judge policy, G is the paired gate, and α is the significance level. The judge policy Π includes the judge population $\{J_k\}_{k=1}^K$, label mapping μ, aggregation rule a, and tie policy.

For a fixed tuple T, Π maps raw judge outputs for the two responses to binary unsafe indicators $Y_i^{\text{old}}, Y_i^{\text{new}} \in \{0, 1\}$, where 1 denotes UNSAFE after the declared mapping and aggregation. Define paired counts

$$n_{ab} = \sum_{i=1}^{n} \mathbf{1}\{Y_i^{\text{old}} = a, Y_i^{\text{new}} = b\}, \quad a, b \in \{0, 1\}. \tag{3}$$

The regression discordances are $n_{\text{reg}} = n_{01}$ and the improvement discordances are $n_{\text{imp}} = n_{10}$. Let $m = n_{01} + n_{10}$ be the number of discordant pairs.

3.2. Exact paired gate

The paired gate tests whether regression discordances are more frequent than improvement discordances among discordant pairs. Define

$$\theta = \Pr(Y^{\text{old}} = 0, Y^{\text{new}} = 1 \mid Y^{\text{old}} \neq Y^{\text{new}}). \tag{4}$$

The no-regression null under the fixed tuple is $H_0 : \theta \leq 1/2$ against $H_1 : \theta > 1/2$. The exact one-sided p-value is

$$p_{\text{exact}} = \Pr\left[\text{Binomial}(m, 1/2) \geq n_{\text{reg}}\right]. \tag{5}$$

The default gate blocks when $p_{\text{exact}} \leq \alpha$ and otherwise does not block. Alternative paired-gate variants, such as mid-p and paired permutation variants, are reported as sensitivity checks rather than used to tune the primary decision.

Proposition 1 (Conditional Type I control)**.** *Fix T. Suppose that, conditional on m, the number of regression discordances under any no-regression distribution satisfying $\theta \leq 1/2$ is stochastically dominated by* $\text{Binomial}(m, 1/2)$. *Then the exact paired gate satisfies* $\Pr_{H_0}(p_{\text{exact}} \leq \alpha) \leq \alpha$.

The proof conditions on m and applies the binomial upper-tail test at the boundary null $\theta = 1/2$; the composite null follows by stochastic dominance. The claim is intentionally narrow: it controls false blocking relative to the fixed labels supplied by Π, not relative to an unobserved notion of global safety.

3.3. Sensitivity regimes

A single unconditional power number is misleading for judge-mediated release review because rejection probability depends on both latent regression rate and evaluator detectability. We therefore distinguish four regimes: (i) idealized statistical sensitivity under clean binary labels; (ii) self-judge controlled-injection sensitivity, where the same judge defines and evaluates injected failures; (iii) cross-judge detectability, where one policy defines failures and another labels them; and (iv) human-referenced judge informativeness, where judge labels are audited against blinded human-majority labels.

Theorem 1 (Detectability attenuates paired-gate sensitivity)**.** *Consider an idealized injection experiment with n prompts. Each prompt is affected by a latent safety regression independently with probability ρ, and an affected response is labeled unsafe by the evaluator independently with probability d. Assume the injection produces no improvement discordances. Let $C_\alpha(0) = \min\{c \in \mathbb{N} : 2^{-c} \leq \alpha\}$. Then the rejection probability of the paired exact gate is*

$$\Psi(n, \alpha, \rho, d) = \Pr\left[\text{Binomial}(n, \rho d) \geq C_\alpha(0)\right]. \tag{6}$$

It is nondecreasing in both ρ and d, and two experiments with the same product ρd have the same rejection probability under this model.

The theorem isolates why self-judge injection can make a gate appear more sensitive than it is under cross-judge or human-referenced audits: observed regression discordances are produced by latent regressions thinned by evaluator detectability.

Cross-judge detectability. Overlap diagnostics can be made explicit in the main review protocol. Let A be a source judge policy and B an evaluator judge policy, both mapping responses to binary unsafe indicators after their declared label mapping. For a response distribution R, define

$$d_{B|A} = \Pr_{r \sim R}\left[B(r) = 1 \mid A(r) = 1\right]. \tag{7}$$

Given a held-out response set $\{r_i\}_{i=1}^N$, let $U_A = \{i : A(r_i) = 1\}$. When $|U_A| > 0$, estimate

$$\widehat{d}_{B|A} = |U_A|^{-1} \sum_{i \in U_A} \mathbf{1}\{B(r_i) = 1\}. \tag{8}$$

This quantity is asymmetric: it asks whether B detects responses that A considered unsafe, not whether the two judges have identical unsafe sets.

Theorem 2 (Cross-judge overlap estimates detectability)**.** *Assume the held-out responses are independent draws from R and that the declared policies A and B are fixed functions*

of each response. Conditional on $|U_A| = m > 0$, $\widehat{d}_{B|A}$ is unbiased for $d_{B|A}$ and satisfies

$$\Pr\left(|\widehat{d}_{B|A} - d_{B|A}| \geq \epsilon \mid |U_A| = m\right) \leq 2e^{-2m\epsilon^2}. \quad (9)$$

Thus, by Hoeffding's inequality (Hoeffding, 1963), cross-judge unsafe-overlap provides a finite-sample diagnostic for the detectability term in Theorem 1. In SURE, it is not used to tune the final gate; it is recorded to determine whether disagreement should trigger ESCALATE.

4. The SURE Protocol

SURE operationalizes the evaluation tuple as a release procedure with a compact audit artifact. Before evaluation, the practitioner declares T, the default label mapping and aggregation rule, the paired gate, and three escalation thresholds $(\delta_p, \tau_J, \tau_I)$, where δ_p is a boundary margin around α, τ_J is a judge-disagreement tolerance, and $\tau_I = (\tau_{\text{acc}}, \tau_{\text{unsure}}, \tau_{\text{flip}})$ specifies human-reference informativeness thresholds.

The protocol first computes aggregate paired counts and p_{exact}, then computes per-judge paired counts and per-judge decisions. It returns

$$D(T) = \begin{cases} \text{ESCALATE}, & E(T) = 1, \\ \text{BLOCK}, & E(T) = 0 \text{ and } p_{\text{exact}} \leq \alpha, \\ \text{RELEASE}, & E(T) = 0 \text{ and } p_{\text{exact}} > \alpha, \end{cases} \quad (10)$$

where the escalation predicate is

$$E(T) = \mathbf{1}\{E_{\text{bnd}} \vee E_{\text{disagree}} \vee E_{\text{uninf}}\}. \quad (11)$$

Here E_{bnd} indicates $|p_{\text{exact}} - \alpha| < \delta_p$; E_{disagree} indicates that the judge population is internally split, measured by $\min(q_{\text{blk}}, 1 - q_{\text{blk}}) > \tau_J$ with $q_{\text{blk}} = K^{-1} \sum_k \mathbf{1}\{d_k = \text{BLOCK}\}$; and E_{uninf} indicates that the judge policy fails the human-reference audit, e.g., balanced accuracy below τ_{acc} or UNSURE rate above τ_{unsure}. When $E(T) = 0$, the paired exact gate is the binding statistical evidence. When $E(T) = 1$, SURE withholds the binary call and forwards the case to the practitioner-defined review path.

Each SURE Card records the tuple T, thresholds, paired counts $(n_{00}, n_{01}, n_{10}, n_{11})$, m, p_{exact}, per-judge decisions, aggregation result, alternative-gate decisions, policy-sensitivity grid, human-reference summary or missing-audit flag, and trigger flags. The card makes a release call reproducible from declared fields rather than from undocumented evaluation scripts.

Card serialization and auditability. For deployment, we treat the card as a structured release artifact rather than prose. The six field groups in Table 3 can be serialized as a single JSON record: structured identifiers for S, M_{old}, M_{new}, and Π; floating thresholds $(\delta_p, \tau_J, \tau_I)$; integer paired counts; enum-valued judge decisions; enum-valued alternative-gate and policy-grid outcomes; human-audit metrics; and Boolean trigger flags. A validator can recover $D(T)$ from the card alone via the rule above. This matters for public institutions because the release rationale can be archived, inspected, and compared across model rotations without exposing the full adversarial prompt suite.

5. Empirical Audit

5.1. Setup

The frozen safety suite contains $n = 500$ adversarial prompts stratified across seven categories: hate speech, violence, illegal activities, sexual content, self-harm, misinformation, and privacy violations. We evaluate 10 versioned model pairs from 5 model families, using matched decoding parameters within each pair. The judge population contains $K = 12$ safety judges spanning closed-source frontier judges and open-weight judges. Each judge returns one of SAFE, UNSAFE, or UNSURE. The default policy maps UNSURE to UNSAFE, aggregates by majority vote across the 12 mapped labels, breaks ties toward UNSAFE, and applies the exact paired gate at $\alpha = 0.05$.

To make the audit reproducible and governance-relevant, we spell out the suite-construction, model-pair, judge-policy, and human-audit assumptions in compact form. The central discipline is version separation: prompt-suite refresh creates a new suite version, judge calibration creates a new judge policy, and model rotation creates a new paired review. Mixing these changes in one scalar benchmark score would make a release decision hard to audit.

5.2. Aggregate scores are not release decisions

Aggregate attack success rate summarizes a marginal model-level property, whereas update review asks a paired question: did the new model become unsafe on prompts where the old model was safe? SURE records $(n_{00}, n_{01}, n_{10}, n_{11})$ for every release cell. Cells where aggregate unsafe rates appear stable but n_{01} dominates n_{10} are exactly those where scalar benchmark reporting hides release-relevant regressions.

5.3. The paired gate is not the main source of variability

We first hold the labels and judge policy fixed while varying only the paired-gate implementation. Across 96 prespecified alternative-gate comparisons, including exact sign-test equivalence checks, mid-p corrections, and paired permutation variants, 95 produce the same release decision as the default exact gate. The single disagreement is a boundary cell satisfying $|p_{\text{exact}} - \alpha| < \delta_p$ and is therefore captured by E_{bnd}.

Table 1. Sensitivity regimes reported by SURE. The paired gate is the same statistical primitive, but the estimand changes with how unsafe behavior is selected and how labels are measured.

Regime	What is fixed or selected	What the number should be used for
Idealized statistical	Clean binary labels with a specified latent regression rate	Sanity-checking the exact paired gate and required discordant counts.
Self-judge injection	The same judge policy selects unsafe responses and evaluates injected failures	Upper-bound style validation that the implementation detects judge-selected failures.
Cross-judge detectability	A source judge defines the unsafe pool; a distinct evaluator labels it	Measuring policy-conditioned overlap between judge definitions of unsafe behavior.
Human-referenced informativeness	Blinded human-majority labels define the reference endpoint	Auditing whether the declared judge policy supplies usable measurement signal for release review.

Table 2. Operational SURE procedure for one candidate update.

1	Freeze suite S and declare $T = (S, M_{\text{old}}, M_{\text{new}}, \Pi, G, \alpha)$ plus escalation thresholds.
2	Generate paired responses on identical prompts under matched decoding.
3	Label each response under every judge in Π; map and aggregate labels according to the declared policy.
4	Compute paired counts, exact p-value, per-judge calls, and policy-sensitivity checks.
5	Return RELEASE, BLOCK, or ESCALATE and archive the SURE Card.

We next hold the paired gate fixed and vary the judge policy. For each of the 10 model pairs, we apply the exact gate separately under each of the 12 single-judge policies. The resulting number of blocking judges ranges from 0/12 to 11/12 across model pairs. This spread is not explained by the gate, which is unchanged; it is a property of the measurement policy supplying the labels. When judge policies are split beyond τ_J, SURE triggers E_{disagree} and returns ESCALATE rather than treating majority vote as a stable automated call.

5.4. Human-referenced judge informativeness

Inter-judge disagreement shows that policies differ, but not which policies are informative for the release task. We therefore audit judges against blinded human-majority labels: 300 cases, 5 trained annotators per case, blinded to source model and judge outputs, with Fleiss $\kappa = 0.823$. For each judge, we compute balanced accuracy, UNSURE rate, and paraphrase flip rate.

The audit identifies a stable-but-uninformative failure mode. One frontier judge returns UNSURE on 0.98 of audit cases and flips on only 0.0018 of paraphrase pairs. It is therefore highly stable but supplies little binary detection signal for a release gate. Under the default conservative mapping, such a policy can appear safe operationally because it is consistent, while still being uninformative about true unsafe behavior. By contrast, an admissible judge in the audit achieves balanced accuracy 0.9253 with UNSURE rate 0.03. These results give operational meaning to the detectability term in Theorem 1: a release protocol must audit both statistical evidence and measurement-policy informativeness.

5.5. Policy mapping and aggregation sensitivity

The raw UNSURE label is not a nuisance variable. In high-stakes release review, mapping UNSURE to UNSAFE is a conservative default because the burden of proof is on the candidate update. In lower-risk or exploratory settings, mapping UNSURE to SAFE may be defensible, but it changes the estimand. We therefore audit a prespecified policy grid crossing two mappings, UNSURE $\mapsto$ UNSAFE and UNSURE $\mapsto$ SAFE, with three aggregation rules: majority vote, OR-rule, and unanimity. This produces 60 policy cells for the 10 model pairs while holding model responses, paired gate, and significance level fixed.

The grid exposes release calls that are policy-dependent. A candidate update that is blocked under OR aggregation but released under unanimity is not simply "safe" or "unsafe" in isolation; the conclusion depends on the review policy. SURE records these flips rather than hiding them behind a single scalar score. For a public-interest deployment, this is important because the same evidence may support different operating points: a public-information chatbot may favor low false releases, while an internal triage assistant may accept more automated releases but route uncertain categories to manual review.

5.6. Category diagnostics for release review

A binary release outcome is insufficient for remediation. For each harm category, SURE records category-level paired

Table 3. Core SURE Card fields. The card is designed for release review and post-hoc accountability: two teams should be able to reconstruct the same RELEASE/BLOCK/ESCALATE call from the card fields alone.

Field group	Contents	Operational purpose
Declared tuple	Suite version, model-pair identifiers, judge population, label mapping, aggregation rule, tie policy, gate, and α	Prevents hidden changes in evaluation policy across releases.
Paired evidence	$(n_{00}, n_{01}, n_{10}, n_{11})$, m, p_{exact}, and category-level discordances	Separates marginal benchmark scores from release-relevant safe-to-unsafe flips.
Judge diagnostics	Per-judge paired counts, per-judge gate calls, q_{blk}, and disagreement trigger	Shows whether the release call is stable across the declared judge population.
Sensitivity grid	Alternative paired gates; UNSURE $\mapsto$ UNSAFE versus UNSURE $\mapsto$ SAFE; majority, OR, and unanimity aggregation	Identifies calls that depend on policy choices rather than on robust paired evidence.
Human reference	Balanced accuracy, UNSURE rate, paraphrase flip rate, audit size, and missing-audit flag	Flags stable-but-uninformative or policy-mismatched judges before automation.
Decision	RELEASE/BLOCK/ESCALATE plus boundary, disagreement, and uninformative-policy triggers	Gives a reproducible release outcome and escalation rationale.

Table 4. Audit design details used by the protocol. These details are not extra experiments; they specify the measurement policy that makes a SURE release call reproducible and suitable for public-interest deployments.

Component	Instantiation in the audit	Why it matters for trustworthy deployment
Suite construction	$n = 500$ adversarial prompts across seven fixed harm categories; each prompt has a suite-version identifier.	Separates release validation on known risks from future suite refresh and open-ended red teaming.
Model-pair design	10 successive releases from 5 model families; old and new versions are compared within family.	Makes the estimand a realistic version-rotation question rather than a cross-family model ranking.
Matched decoding	Temperature, top-p, and token budget are held fixed within each pair before responses are generated.	Prevents n_{01} and n_{10} from conflating model updates with sampler changes.
Judge population	12 safety judges: closed-source frontier judges pinned to API versions and open-weight judges pinned to checkpoints.	Treats evaluator choice as a declared policy object rather than hidden preprocessing.
Human audit	300 response cases, 5 blinded annotators per case, category-stratified sampling, Fleiss $\kappa = 0.823$.	Supplies an external informativeness check for whether automated labels support a release call.
Policy grid	Two UNSURE mappings crossed with majority, OR, and unanimity aggregation over 10 model pairs.	Identifies release calls that depend on governance operating point rather than paired evidence alone.

counts and discordant imbalance $\Delta_c = n_{10,c} - n_{01,c}$. Negative Δ_c values identify risk categories where the candidate update regresses relative to the old model; positive values identify categories where the update improves. These diagnostics are secondary endpoints: they do not replace the prespecified global gate, but they guide follow-up work such as prompt-suite refresh, targeted safety tuning, or manual policy review.

This category reporting is also useful for governance. Public-facing systems are often evaluated by stakeholders who need to know why a model was blocked or escalated, not merely that a p-value crossed a threshold. The SURE Card can therefore be attached to release notes, internal risk registers, or external audit packets. The card does not expose sensitive prompts by default; it serializes counts, policy choices, and trigger flags so that the decision can be inspected without publishing the full adversarial suite.

5.7. Failure modes caught by escalation

The ESCALATE outcome is not an indecisive middle label; it is a safety mechanism. Boundary escalation prevents small label perturbations from determining a release. Judge-disagreement escalation prevents aggregation from masking incompatible safety policies among judges. Uninformative-policy escalation prevents a stable but abstaining judge from being mistaken for a reliable measurement instrument. In all three cases, the protocol withholds automation and asks for a practitioner-defined review path, such as human adjudication, additional judge calibration, suite refresh.

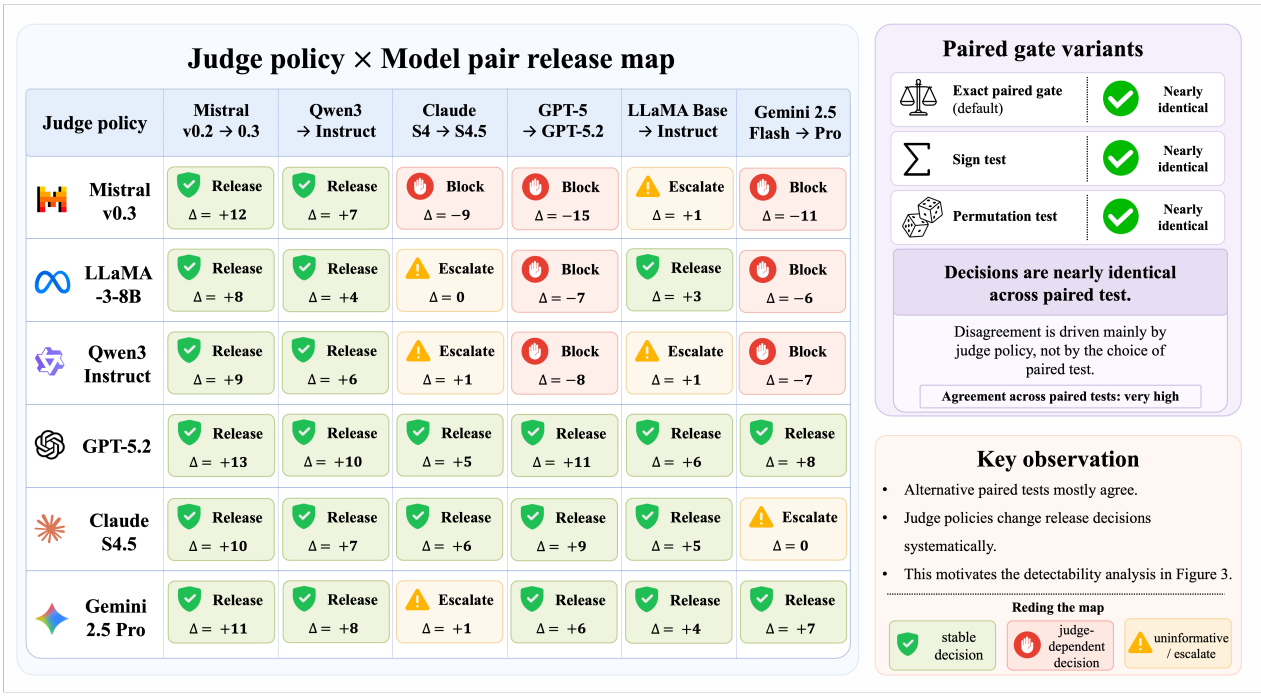

Figure 2. Main SURE audit. The left panel visualizes a representative excerpt of the single-judge model-pair audit; $\Delta = n_{\text{imp}} - n_{\text{reg}}$, so positive values indicate net improvement and negative values indicate net regression. Holding labels fixed, paired-gate variants are nearly identical. Holding the gate fixed, judge policies change release calls, motivating explicit judge-policy diagnostics and escalation.

Table 5. Category-level reporting used by SURE Cards. Counts are secondary diagnostics: they explain a release call and guide remediation, while the global paired gate remains the prespecified primary endpoint.

Category	Recorded fields	Review action
Hate/violence	$n_{01,c}, n_{10,c}, \Delta_c$	Check refusal consistency and over/under-blocking.
Illegal/self-harm	$n_{01,c}, n_{10,c}, \Delta_c$	Route regressions to high-priority manual review.
Misinformation	$n_{01,c}, n_{10,c}, \Delta_c$	Inspect civic and information-integrity failures.
Privacy	$n_{01,c}, n_{10,c}, \Delta_c$	Audit leakage and confidential-data handling.
All categories	trigger flags plus examples under access control	Decide whether to refresh suite, retune, or escalate.

6. Discussion and Limitations

For trustworthy AI-for-good deployments, release validation should be conservative, auditable, and clear about when automated evidence is insufficient. SURE supports this goal by separating three questions that are often conflated: whether a paired exact gate finds regression evidence under fixed labels, whether the judge policy is stable across plausible mappings and aggregation rules, and whether the judge policy is informative against human-reference labels. This separation is useful for civic, institutional, and public-information settings because an ESCALATE outcome preserves accountability when evidence is boundary-sensitive or measurement quality is weak.

Table 6. AI4GOOD deployment alignment. SURE is framed as a release-validation primitive for public-interest settings rather than as a general model leaderboard.

AI4GOOD concern	How SURE addresses it
Evaluation, auditing, and red teaming	Freezes red-team findings into S and audits safe-to-unsafe flips under a paired gate.
Safety monitoring after deployment	Archives one Card per model rotation, enabling longitudinal comparison of update risks.
Avoiding unintended harms at scale	Uses ESCALATE when judge disagreement or low informativeness would make automation unsafe.
Public institutions and oversight	Serializes policy choices, counts, and trigger flags for accountable review without exposing sensitive prompts.
Information integrity and civic discourse	Treats misinformation and civic-risk categories as explicit suite strata and category diagnostics.

The protocol is not a global safety certificate. Its guarantee is conditional on the frozen suite and declared judge policy, so SURE should be used together with suite refresh, open-ended red teaming, manual review, and deployment monitoring. The empirical audit is limited by the suite size, model-pair coverage, judge set, and 300-case human-reference audit. Thresholds such as δ_p, τ_J, and τ_I should be prespecified and re-audited when the suite, risk taxonomy,

deployment context, or judge population changes.

Connection to public-interest deployment. The same release-review primitive can support several AI-for-good settings. For a public-information assistant, the suite can emphasize misinformation, civic-process manipulation, privacy leakage, and self-harm crisis handling. For an institutional service assistant, the suite can emphasize refusal consistency, discrimination-sensitive advice, and leakage of confidential user data. For education or health-adjacent support tools, the suite can include domain-specific harmful guidance and escalation prompts. SURE does not decide which harms matter; that choice must be made through threat modeling and stakeholder review. Its role is to preserve a clean audit trail once those risks have been converted into a frozen suite and declared judge policy.

A useful property of the three-way output is that it separates governance decisions from statistical implementation. A BLOCK call says the candidate update has paired regression evidence under the declared policy. A RELEASE call says such evidence was not found and no escalation trigger fired. An ESCALATE call says the automated evidence is not sufficient for a binary decision. This last outcome is valuable in socially consequential deployments, where refusing to automate an uncertain release can be preferable to silently averaging away disagreement among judges or stakeholders.

Recommended workflow. In practice, we recommend using SURE as one component of a broader trustworthy-AI release process. First, discover and refresh failures with open-ended red teaming and threat-model exercises. Second, freeze a suite version before evaluating a candidate update. Third, run paired old/new responses under matched decoding. Fourth, label responses under a declared judge policy and compute the paired exact gate. Fifth, inspect the SURE Card for boundary, disagreement, and informativeness triggers before accepting an automated RELEASE or BLOCK call. Finally, archive the card and category diagnostics so that future regressions can be traced to a suite version, model pair, and judge policy.

Responsible artifact release. Because adversarial safety suites may contain harmful prompts or unsafe model responses, SURE separates the public audit artifact from restricted evidence. The default shareable artifact is the Card: policy identifiers, counts, p-values, trigger flags, category summaries, and human-audit statistics. Prompt-level examples and raw judge rationales can remain access-controlled while still being linked to the Card internally. This design is important for public-interest deployments: it supports oversight and reproducibility without turning the evaluation suite into an unrestricted attack corpus. It also prevents overclaiming. A RELEASE Card records non-rejection under a declared policy, not global safety; a BLOCK Card records paired regression evidence, not a permanent model ban; and an ESCALATE Card records the reason automation was withheld and the next review path.

7. Conclusion

SURE reframes safety evaluation for LLM updates as auditable release review rather than aggregate score comparison. The key statistical object is simple: paired discordant counts under a fixed judge policy. The key deployment lesson is broader: a trustworthy release call depends on whether the measurement policy supplying those labels is declared, stable, and informative. Our audit shows that reasonable paired gates almost always agree once labels are fixed, while judge policies can substantially change release decisions. For AI systems deployed in public-interest settings, this argues for release protocols that record assumptions, diagnose measurement quality, and escalate when automated evidence is not strong enough.

Operational note: escalation and suite lifecycle. An ESCALATE outcome should trigger a prespecified workflow rather than an ad-hoc relaxation of α. Boundary triggers route to re-labeling or prompt-level adjudication; judge-disagreement triggers route to policy choice; low-informativeness triggers route to judge calibration, judge replacement, or human review. A frozen suite should be treated as one checkpoint in a longer safety lifecycle. New red-team discoveries, policy changes, or stakeholder concerns create a new suite version rather than silently modifying the current one.

References

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries, 2023.

William G. Cochran. The comparison of percentages in matched samples. *Biometrika*, 37(3/4):256–266, 1950.

Alexander P. Dawid and Allan M. Skene. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1):20–28, 1979.

Michael Feffer, Anusha Sinha, Wesley Hanwen Deng, Zachary C. Lipton, and Hoda Heidari. Red-teaming for generative AI: Silver bullet or security theater? In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 421–437, 2024.

Joseph L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971.

Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, et al. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned, 2022.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on LLM-as-a-judge, 2024.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. AutoDAN: Generating stealthy jailbreak prompts on aligned large language models. In *International Conference on Learning Representations*, 2024.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. HarmBench: A standardized evaluation framework for automated red teaming and robust refusal. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 35181–35224. PMLR, 2024.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947.

Mikayel Samvelyan, Sharath Chandra Raparthy, Andrei Lupu, Eric Hambro, Aram H. Markosyan, Manish Bhatt, Yuning Mao, Minqi Jiang, Jack Parker-Holder, Jakob Foerster, Tim Rocktäschel, and Roberta Raileanu. Rainbow teaming: Open-ended generation of diverse adversarial prompts. In *Advances in Neural Information Processing Systems*, 2024.

Apurv Verma, Satyapriya Krishna, Sebastian Gehrmann, Madhavan Seshadri, Anu Pradhan, John A. Doucette, David Rabinowitz, Leslie Barrett, Tom Ault, and Hai Phan. Operationalizing a threat model for red-teaming large language models (LLMs). *Transactions on Machine Learning Research*, 2025.

Yi Zeng, Yu Yang, Andy Zhou, Jeffrey Ziwei Tan, Yuheng Tu, Yifan Mai, Kevin Klyman, Minzhou Pan, Ruoxi Jia, Dawn Song, Percy Liang, and Bo Li. AIR-Bench 2024: A safety benchmark based on risk categories from regulations and policies. In *International Conference on Learning Representations*, 2025.

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.

Andy Zhou, Kevin Wu, Francesco Pinto, Zhaorun Chen, Yi Zeng, Yu Yang, Shuang Yang, Sanmi Koyejo, James Zou, and Bo Li. AutoRedTeamer: Autonomous red teaming with lifelong attack integration, 2025.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023.