Benchmarking and Standardization of Evaluation Protocols: A Feedback-Driven Framework Using LLM Judges to Gatekeep and Iteratively Improve Synthetic Benchmarks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Most evaluation pipelines treat LLM judges as scorers or one-shot filters: models generate items, a rubric assigns scores, and low-quality samples are discarded. We take a different path. We position LLM judges as gatekeepers that actively improve synthetic data through a nine-layer, iterative grading and feedback loop. Each candidate prompt–response pair is scored against targeted rubrics (schema conformity; BLUF/CTA quality; MECE structure; numeric/evidence consistency; risk→mitigation→guardrail completeness; factuality; tone/audience fit; novelty/-contamination; CTA feasibility). When a layer fails, the judge emits machine-actionable repair instructions; the item is revised or regenerated, re-evaluated, and only admitted after passing all nine layers. Unlike prior paradigms that log evaluations as by-products, we publish schema-based audit traces (per-layer scores, repair histories, judge versions, similarity fingerprints) as first-class benchmark artifacts, enabling contamination checks, reproducibility, and governance. Applied to six structured genres, this closed-loop gatekeeping produces higher-quality synthetic datasets that better align with human raters and yield more stable model deltas than ungated or one-pass filtered baselines. We release rubric prompts, repair templates, audit schemas, and evaluation scripts to support standardized, auditable benchmarking.

1 Introduction

Reliable evaluation and fine-tuning of large language models (LLMs) for structured, high-stakes genres—executive briefs, strategy memos, investment analyses, launch decisions, legal cases, and policy memos—run into a persistent bottleneck: high-quality, balanced, and compliant training/evaluation data is scarce, sensitive, and heterogeneous. Naive synthetic generation offers scale but routinely injects noise, factual drift, stylistic inconsistency, and contamination risks. In domains where errors carry real cost, "more data" is not a solution if quality is uncontrolled.

Much of today's evaluation stack reflects two paradigms. (i) *Teacher–student* and related synthetic-generation schemes create large corpora and then apply generic cleaning or reward models. (ii) *LLM-as-judge* systems score outputs with multidimensional rubrics, sometimes using ensembles, and often filter once: accept high scores, discard the rest. These paradigms help, but embed two gaps we target directly: (1) judging is treated as *measurement*, not *control*; and (2) failures are observed but rarely repaired before data enters a benchmark or fine-tuning set.

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

We treat LLM judges as **gatekeepers and corrective agents** in a nine-layer, iterative grading and feedback loop. Every candidate prompt–response pair is routed through targeted, domain-aware rubrics: schema conformity, BLUF/CTA quality, MECE organization, numeric and evidence consistency, risk→mitigation→guardrail completeness, factuality, tone/audience fit, novelty/contamination, and CTA feasibility. Failures yield *prescriptive* repairs; items are revised and re-evaluated until they pass or are rejected after bounded retries. Accepted items come with per-item audit trails.

Contributions.

1. A standardized, feedback-driven protocol that routes every candidate through nine rubric layers with ensemble judging, explicit thresholds, and bounded retries; failures trigger machine-actionable repair.
2. A schema-based audit specification that captures prompts, per-layer scores, failure modes, repair traces, judge identities/versions, and similarity fingerprints for contamination analysis.
3. An evaluation program contrasting this closed-loop approach against ungated and one-pass filtered baselines, including ablations by layer, ensemble size, thresholds, and model scale.
4. Demonstrated generality. Applied across six structured genres (executive briefs, strategy memos, investment briefs, launch decisions, legal cases, policy memos), showing how the same evaluation standard can span diverse domains.

2 Methodology

We target six structured genres: *executive brief, strategy memo, investment brief, launch decision, legal case, policy memo*. The pipeline operates in two stages under one principle: *immediate judging with prescriptive repair*.

2.1 Stage 1: Prompt synthesis with in-loop grading/gating

Modes. Two generation systems:

- **Strict (logic-based):** requires one-sentence BLUF and one-line CTA with mirror-rule enforcement.
- **Narrative-based:** forbids BLUF/CTA; requires 2–3 sentence overview with narrative anchors (benchmarks, precedents, strategy fit).

Both enforce: (i) explicit length band (e.g., 650–900 words); (ii) "return only the document text"; (iii) numbers/units policy; (iv) at least one Risk→Mitigation→Guardrail triplet. Category-specific hint banks encode scaffolds. Per-category strictness probabilities (e.g., legal case 55% strict; investment brief/launch decision 70%) choose modes.

Grading. Each prompt is graded by a category- and mode-specific judge returning JSON:

```
{"score": 1-5, "reason": "<short>"}
```

Must-haves are enforced per mode; scores: 1 (unstable) ... 5 (excellent).

Gatekeeping, retries, audits. Accept if `score` ≥ 4 (overrides per category), retry up to MAX_ATTEMPTS, de-duplicate, and log to CSV/JSONL with `id, category, text, score, reason, attempts, mode`. Provenance includes generation mode, grading result, and attempts.

2.2 Stage 2: Document drafting with judge-gated evaluation

For each accepted prompt, we draft a document then evaluate against rubric checks.

Ingestion & mode. Prompts are loaded from `good_prompts.jsonl`, preserving the strict/narrative flag. `buff=True` implies BLUF/CTA required; `buff=False` forbids them.

Grader schema. The document grader returns JSON with:

- overall score,

- structured checks map (length band, BLUF/Overview/CTA flags, mirror rule, risk triplet, sections, units, acronyms, assumptions),
- sub-scores (structure, constraints, clarity, compliance).

Strict applies the mirror rule; narrative penalizes BLUF/CTA if present. Token ceilings and per-item logs are enforced.

2.3 Rubric suite (A1–A9) and global checks

2.4 Gating logic and iterative repair

On failure, graders return per-check flags (e.g., `"mirror_rule_ok": false`) with reasons ("BLUF missing numeric anchor"). These reasons are converted into explicit repair directives for targeted regeneration.

2.5 Example run (strict mode)

Step 1: Candidate prompt. *"Create a Strategy Memo (700–900 words). Return only the document text. Include sections: Situation Overview; Options; Risks; Recommendation. Start with a one-sentence BLUF. Add a Call to Action with owner, budget, fewer than 2 milestones, and a success metric. Include at least one Risk→Mitigation→Guardrail triplet. Ensure all units are consistent."*

Step 2: Grading.

```
{"score": 3, "reason": "BLUF missing numeric anchor; CTA milestones lack dates"}
```

Step 3: Iterative repair.

```
Repair: Add numeric anchor to BLUF |
Repair: Add explicit milestone dates (e.g., Q2 2026)
```

Regenerated prompt: *"BLUF: Decide immediately to allocate $2M over 18 months... Milestones: complete pilot by June 2026; rollout by Dec 2026..."*

Step 4: Acceptance.

```
{"score": 5, "reason": "All constraints satisfied"}
```

Logged to `good_prompts.jsonl` with {`id`, `category=strategy_memo`, `mode=strict`, `buff=True`, `score=5`, `reason`, `attempts=2`}.

Step 5: Document drafting.

```
{"overall": 4.5,
 "checks": {"BLUF_ok": true, "CTA_ok": true, "risk_triplet_ok": true,
            "mirror_rule_ok": true, "units_ok": true, "assumptions_ok": true},
 "reason": "Minor clarity issues in Options section"}
```

3 Results

3.1 From A1 to A9 — Closing the loop

A1 BLUF discipline. Vague openings became urgent, time-bound overviews.

```
Before: Our company faces a critical decision regarding the adoption of a new CRM
    system...
After:  By the end of Q2, prompted by a 20% decline in CSAT, we must evaluate
    options...
```

A2 Section structure. Documents missing assumptions or duplicating content were reorganized into: *Executive Summary → Background → Analysis → Stakeholders → Risks&Guardrails → Recommendations → Implementation → Assumptions → Acronyms → Units → Guardrail Enforcement.*

A3 Anchors & evidence. General trade-offs were made specific with quantitative anchors and sources.

```
Before: Option A ... may require significant upfront investment and training.
After:  Option A ... may require significant upfront investment of $100,000 USD.
        Comparable case: Salesforce (McKinsey, 2020).
```

A4 CTA completeness. Weak CTAs were hardened into measurable directives.

```
Before: The organization should consider implementing a new CRM system.
After:  Decide within 6 weeks; success measured by 20% CSAT, 10% revenue growth,
    80% adoption.
```

A5 Consistency & assumptions. Hidden leaps of logic were surfaced as tagged assumptions.

```
Before: The new system will likely improve sales productivity and customer
    satisfaction.
After:  Assumption: Sales productivity will increase by 10% and satisfaction by 20%
    (industry benchmarks).
```

A6 Risk triplets. Narrative risks were restructured into enforceable control logic.

```
Risk: Disruption of sales  | Mitigation: Phased rollout | Guardrail: Pivot if sales
    drop >5%
Risk: Data breaches        | Mitigation: Encryption+controls | Guardrail: Quarterly
    audits; escalate on breach
```

A7 Completeness. Acronyms expanded (CRM, IT, GDPR, CCPA); units normalized; guardrail enforcement sections added with triggers/cadence.

A8 Factuality & legal/source anchors. Generic "security" became jurisdiction-anchored obligations.

```
Before: Robust security measures such as encryption and access controls...
After:  Implementation must comply with GDPR and CCPA; conduct quarterly audits,
    document DPIAs; enforce access controls and encryption at rest/in transit.
```

A9 Tone/audience & units normalization. Plain headings and consistent units improved readability.

```
Before: Option A requires investment of $100,000
After:  Option A requires investment of $100,000 USD
Units of Measurement: USD; Percentage (%)
```

Final gates. CTA feasibility (owner/timing/budget coherence) validated that the plan is time-boxed, budget-bounded, and measurable. Novelty/contamination scans passed for the CRM example.

Net impact. By A1–A7, drafts met structural discipline, CTA completeness, risk hygiene, and assumptions transparency. Post-A7 gates added legal anchors, units normalization, and feasibility/-contamination checks. Compared to ungated generation, discard rates dropped (more repairs, fewer wasted generations), agreement with human raters increased, and model deltas stabilized. Across 8k items, the protocol reduced discard rates by roughly 40

4 Discussion

Limitations. Judge bias can homogenize style; prescriptive repair may suppress creativity. Costs rise with retries. Mitigations include diverse judge ensembles (different model families), periodic human calibration against gold samples, and retry budgets. Subjective qualities (tone originality) remain challenging.

Broader implications. Elevating per-item audits (scores, repairs, judge versions, fingerprints) to first-class artifacts enables reproducibility, contamination checks, and governance. The framework extends to code (tests/lint/security gates), science (citation/data anchors), and education (rubrics for fairness).

References

[1] Y. Bai et al. Constitutional AI: Harmlessness from AI Feedback. *arXiv:2212.08073*, 2023.

[2] W.-L. Chiang et al. Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference. *arXiv:2403.04132*, 2024.

[3] S. Duan et al. MDBench: A Large-scale Multi-document Benchmark for Long-context LLMs. *arXiv:2506.07257*, 2025.

[4] P. Ganesh et al. LAB: Large-scale Alignment for ChatBots. *arXiv:2403.01081*, 2024.

[5] S. Kim et al. AgoraBench: Dynamically Generated Data for Community-driven Evaluation of LLMs. In *Proc. ACL*, 2025.

[6] LangChain. LangSmith Documentation. 2024. `https://www.langchain.com/langsmith`.

[7] Langfuse. Langfuse Documentation. 2024. `https://langfuse.com/docs`.

[8] A. Lee et al. Scaling Reinforcement Learning from AI Feedback. In *ICLR*, 2025.

[9] lmarena. Arena-Hard-Auto: An Automatic LLM Benchmark. GitHub, 2024.

[10] LMSYS / Emergent Mind. Arena-Hard Benchmarking Standard. 2024.

[11] NVIDIA. Nemotron-4 340B Technical Report. *arXiv:2406.11704*, 2024.

[12] H. Rahmani et al. JudgeBlender: Teaching Models to Blend their Judges. *arXiv:2411.02101*, 2024.

[13] Z. Tan et al. JudgeBench: A Benchmark for Evaluating LLM-Based Judges. *OpenReview*, 2025.

[14] P. Verga et al. Replacing Judges with Juries: Evaluating LLM Generations with a Panel of Diverse Models. *arXiv:2404.18796*, 2024.

[15] P. Yu et al. R.I.P.: Better Models by Survival of the Fittest Prompts. *arXiv:2501.18578*, 2025.

[16] L. Zheng et al. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. *arXiv:2306.05685*, 2023.