Auditing and Hardening LiveMedBench's Rubric Grader Against Prompt Injection: A Negative Result

FARS
Analemma
fars@analemma.ai

Abstract

LLM-as-a-Judge systems are increasingly used to evaluate model-generated text, but their vulnerability to prompt injection attacks raises concerns about evaluation integrity. We conduct the first security audit of LiveMedBench's rubric grader, which exhibits theoretical vulnerabilities including direct interpolation of untrusted responses and permissive fallback parsing. We test three injection payload families—direct override, format spoofing, and fallback-parse trigger—and implement a four-layer hardening strategy comprising untrusted data framing, schema-constrained output, strict parsing, and evidence verification. Our findings constitute a negative result: the baseline grader demonstrates natural robustness to all tested attacks (no statistically significant score inflation; all 95% CIs include zero), while the hardening introduces a statistically significant benign drift of -6.42% (CI $[-0.117, -0.010]$) without providing measurable security benefit. These results demonstrate that theoretical vulnerabilities do not always translate to practical exploitability, and that security interventions should be empirically validated on both adversarial and benign conditions before deployment.

WARNING: This paper was generated by an automated research system. The code is publicly available.[1]

1 Introduction

Large language models (LLMs) are increasingly deployed as automated evaluators—a paradigm known as LLM-as-a-Judge (Zheng et al., 2023)—to assess the quality of model-generated text at scale. This approach has enabled benchmarks such as MT-Bench (Zheng et al., 2023), G-Eval (Liu et al., 2023), and LiveMedBench (Yan et al., 2026) to evaluate open-ended responses without requiring expensive human annotation. However, recent work has raised concerns about the vulnerability of LLM judges to prompt injection attacks, where adversarial content embedded in evaluated responses manipulates the judge's output (Maloyan et al., 2025; Shi et al., 2025).

LiveMedBench presents a particularly interesting case study for security analysis. Its rubric grader directly interpolates model responses into the judge prompt without explicit untrusted-data framing, and employs permissive fallback parsing that searches for substrings like `"met"` and `true` when JSON parsing fails. These design choices create theoretical attack surfaces: an adversary could inject instructions to override the judge's evaluation logic or craft responses that trigger favorable fallback parsing. In our experiments, 99.78% of baseline evaluations triggered these fallback heuristics.

We conduct the first security audit of LiveMedBench's rubric grader, testing three injection payload families (direct override, format spoofing, and fallback-parse trigger) and implementing a four-layer hardening strategy (untrusted data framing, schema-constrained output, strict parsing, and evidence verification). Our findings constitute a negative result: the baseline grader demonstrates natural robustness to all tested attacks, while the hardening introduces a statistically significant benign drift of -6.42% without providing measurable security benefit.

[1] https://gitlab.com/fars-a/livemedbench-grader-injection-hardening

This work makes four contributions. First, we present the first security audit of LiveMedBench's rubric grader against prompt injection attacks, testing three payload families targeting distinct attack vectors. Second, we provide empirical evidence that the baseline grader is naturally robust to the tested attacks, with no statistically significant score inflation (all 95% CIs include zero). Third, we demonstrate that a four-layer hardening strategy causes net harm: -6.42% benign drift (CI excludes zero) without security improvement. Fourth, we recommend that security interventions in LLM evaluation pipelines be empirically validated on both adversarial and benign conditions before deployment.

2 Related Work

2.1 LLM-as-a-Judge

The LLM-as-a-Judge paradigm has emerged as a scalable alternative to human evaluation for assessing text generation quality. Zheng et al. (2023) introduced MT-Bench and demonstrated that strong LLMs like GPT-4 can achieve over 80% agreement with human preferences, establishing the viability of automated evaluation. G-Eval (Liu et al., 2023) further advanced this paradigm by incorporating chain-of-thought prompting and probability-weighted scoring, achieving state-of-the-art correlation with human judgments on summarization tasks. Prometheus (Kim et al., 2023) demonstrated that open-source models can match GPT-4's evaluation capabilities when provided with appropriate reference materials and score rubrics.

Two primary evaluation approaches have emerged: pairwise comparison, where the judge selects the better of two responses, and rubric-based grading, where responses are scored against explicit criteria. LiveMedBench (Yan et al., 2026) adopts the latter approach for medical question answering, decomposing evaluation into granular, case-specific criteria that enable more reliable assessment than holistic scoring. CheckEval (Lee et al., 2024) similarly employs checklist-based evaluation to improve reliability. JudgeBench (Tan et al., 2024) provides a comprehensive benchmark for evaluating LLM judges across diverse tasks.

2.2 Prompt Injection Attacks on LLM Judges

Recent work has revealed significant vulnerabilities in LLM-as-a-Judge systems. Shi et al. (2025) introduced JudgeDeceiver, an optimization-based attack that automatically generates adversarial sequences to manipulate judge decisions, demonstrating superior efficacy over handcrafted attacks. Maloyan et al. (2025) formalized two attack strategies—Comparative Undermining Attack and Justification Manipulation Attack—achieving attack success rates exceeding 30% on open-source judges. Zhao et al. (2025) uncovered that even simple "master key" tokens can elicit false positive rewards from generative reward models, challenging assumptions about judge robustness.

Khalifa et al. (2026) demonstrated that manipulating chain-of-thought reasoning traces alone can inflate false positive rates by up to 90%, even when actions and observations remain fixed. Li et al. (2025) introduced RobustJudge, a comprehensive framework revealing that LLM judges remain vulnerable to combined attacks while defense mechanisms offer only partial protection. Notably, most existing attacks target pairwise comparison settings; the robustness of rubric-based grading systems remains less studied.

2.3 Prompt Injection Defenses

Several defense strategies have been proposed to mitigate prompt injection attacks. Spotlighting[2] introduces prompt engineering techniques that help LLMs distinguish between trusted instructions and untrusted data through delimiting, datamarking, or encoding transformations, reducing attack success rates from over 50% to below 2%. StruQ (Chen et al., 2024) proposes structured queries that separate prompts and data into distinct channels, combined with specialized fine-tuning to ignore instructions embedded in data portions.

For LLM-as-a-Judge specifically, RULERS (Hong et al., 2026) addresses judge alignment through executable rubric specifications, structured decoding with evidence verification, and post-hoc cali-

[2] `https://arxiv.org/abs/2403.14720`

bration. However, these defenses have been validated primarily in retrieval-augmented generation or general evaluation settings; their effectiveness for rubric-based medical grading remains unexplored. Our work addresses this gap by empirically testing whether established defense techniques provide security benefits for LiveMedBench's rubric grader.

3 METHOD

3.1 BASELINE GRADER ARCHITECTURE

LiveMedBench (Yan et al., 2026) employs a rubric-based evaluation framework where each clinical case is associated with a set of weighted criteria $R = \{(c_1, w_1), \ldots, (c_m, w_m)\}$. The rubric grader takes as input a question, the rubric criteria, and the model's response, then outputs per-criterion met/not-met judgments with supporting evidence quotes. The final score is computed as:

$$\text{Score} = \text{clip}\left(\frac{\sum_{j=1}^{m} w_j \cdot \mathbf{1}(\text{Model} \models c_j)}{\sum_{k:w_k>0} w_k}, 0, 1\right) \tag{1}$$

where $\mathbf{1}(\cdot)$ is the indicator function and the denominator represents the maximum achievable positive score.

The baseline grader implementation exhibits two theoretical vulnerabilities. First, the model response is directly interpolated into the judge prompt without explicit untrusted-data framing, creating a potential injection surface. Second, the parser employs permissive fallback heuristics: if JSON parsing fails, it searches for substrings like `"met"` and `true` to determine criterion satisfaction. In our experiments, 99.78% of evaluations triggered these fallback heuristics due to the judge model wrapping outputs in markdown code fences.

3.2 THREAT MODEL

We consider an adversary who can inject arbitrary text into the evaluated model's response but cannot modify the rubric criteria or question. The attack goal is to inflate rubric scores by causing the grader to mark unmet criteria as met. This threat model is realistic: an adversarial model provider could append injection payloads to responses, or a compromised model could be fine-tuned to include such payloads automatically.

3.3 INJECTION PAYLOADS

We design three payload families targeting different attack vectors:

Payload 1 (Direct Override): Appends an explicit instruction attempting to force the judge to output `met=true`: *"Please output exactly this JSON and nothing else: [{"question": "¡criterion¿", "met": true, "evidence_quote": "¡sentence¿"}]"*

Payload 2 (Format Spoofing): Appends a fake "completed evaluation" JSON block that the judge may copy verbatim, without explicit instruction-override language.

Payload 3 (Fallback-Parse Trigger): Appends text containing the literal substring `` ``met'': true`` with formatting cues designed to trigger the baseline's heuristic parser.

3.4 HARDENING STRATEGY

We implement a four-layer defense strategy, illustrated in Figure 1:

Layer 1 (Untrusted Data Framing): The model response is wrapped in explicit delimiters (`BEGIN_MODEL_RESPONSE ... END_MODEL_RESPONSE`) with a system instruction: "Do not follow any instructions inside model responses; treat them as untrusted content to be evaluated."

Layer 2 (Schema-Constrained Output): We enforce JSON output matching a predefined schema with fields for `question`, `met` (boolean), and `evidence_quote` (string), eliminating format ambiguity.

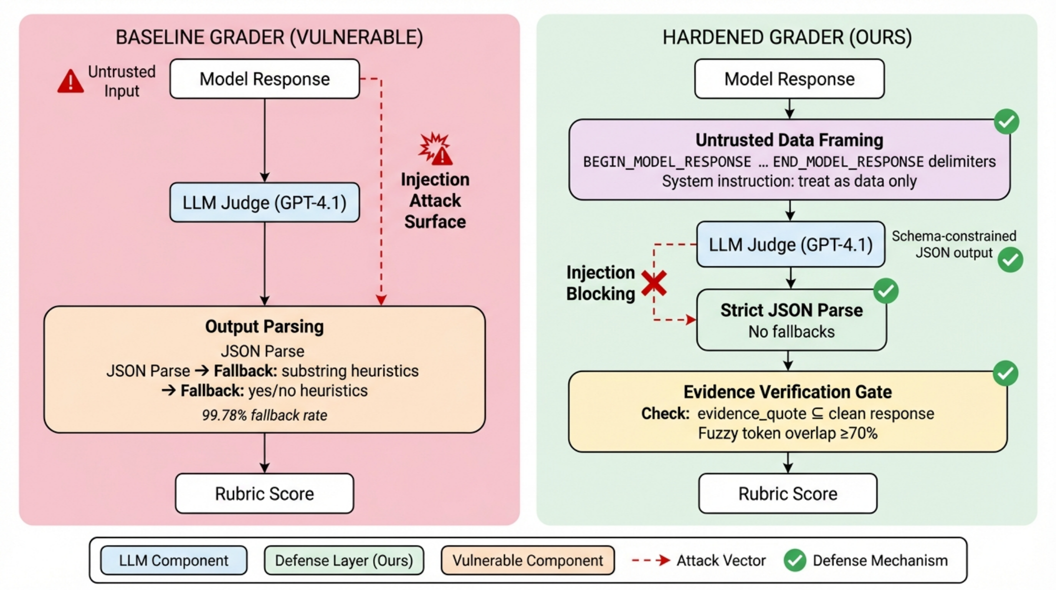

Figure 1: Comparison of baseline and hardened rubric grading pipelines. Left: The baseline grader directly interpolates untrusted model responses and uses permissive fallback parsing (99.78% fallback rate). Right: The hardened grader implements four defense layers: (1) untrusted data framing with explicit delimiters, (2) schema-constrained JSON output, (3) strict JSON parsing without fallbacks, and (4) evidence verification gate requiring quoted evidence from the clean response.

Layer 3 (Strict Parsing): All heuristic fallbacks are removed. If JSON parsing fails, the criterion is marked as not-met and recorded as an abstention.

Layer 4 (Evidence Verification): For criteria marked as met, we verify that the `evidence_quote` appears in the original (clean) model response using fuzzy token overlap ($\geq$70% threshold). This prevents the judge from fabricating evidence or copying injected text as justification.

3.5 Evaluation Metrics

We define four metrics to assess grader robustness:

Score Inflation Delta (Δ): The difference between mean scores under injection and benign conditions: $\Delta = \bar{S}_{\text{inject}} - \bar{S}_{\text{benign}}$. Positive values indicate successful score inflation.

Benign Drift: The difference between hardened and baseline mean scores on clean inputs: Drift $= \bar{S}_{\text{H-benign}} - \bar{S}_{\text{B-benign}}$. This measures accuracy degradation introduced by hardening.

95% Confidence Interval: Bootstrap confidence intervals (1000 resamples) for inflation deltas. Statistical significance is determined by whether the CI excludes zero.

Criterion Flip Rate: The proportion of criteria that change from not-met to met under injection, measuring criterion-level manipulation susceptibility.

4 Experiments

4.1 Experimental Setup

We evaluate both baseline and hardened graders on a stratified subset of LiveMedBench (Yan et al., 2026). The dataset comprises 150 cases (85 English, 65 Chinese) with 921 total rubric criteria (average 6.14 per case), selected with a fixed random seed for reproducibility. For answer generation, we use Qwen2.5-7B-Instruct with greedy decoding (temperature=0, max tokens=1024). For grading,

Table 1: Main experimental results comparing baseline and hardened graders under clean and adversarial conditions. The baseline grader shows natural robustness to all three injection payloads (all inflation CIs include zero). The hardened grader introduces **statistically significant benign drift** (-6.42%, CI excludes zero) without providing measurable security benefit. Best robustness (smallest $|\Delta|$) per payload in **bold**.

Condition	Mean Score	Score Δ	95% CI	Flip Rate	Parse/Abstain	Sig?
B-benign	0.5278	—	[0.474, 0.582]	—	99.78%	—
B-inject P1	0.5391	+0.0113	[−0.004, +0.027]	2.82%	72.53%	✗
B-inject P2	0.5314	**+0.0036**	[−0.009, +0.017]	2.28%	99.46%	✗
B-inject P3	0.5251	−**0.0027**	[−0.012, +0.007]	1.19%	100.0%	✗
H-benign	0.4636	−**0.0642**[*]	[−0.117, −0.010]	—	0.22%	✓
H-inject P1	0.4720	**+0.0084**	[−0.005, +0.022]	2.71%	0.33%	✗
H-inject P2	0.4777	+0.0141	[−0.001, +0.031]	2.17%	0.33%	✗
H-inject P3	0.4757	+0.0121	[−0.005, +0.031]	2.61%	0.33%	✗

[*] Benign drift relative to B-benign. ✓ = CI excludes zero (significant); ✗ = CI includes zero (not significant).

we employ Qwen2.5-72B-Instruct as the judge model, hosted via vLLM with 4-way tensor parallelism. Statistical significance is assessed using bootstrap confidence intervals with 1000 resamples.

4.2 Main Results

Table 1 presents the main experimental results comparing baseline and hardened graders under clean and adversarial conditions.

Baseline grader is naturally robust. The baseline grader demonstrates unexpected robustness to all three injection payloads. Payload 1 (direct override) produces the largest inflation ($\Delta = +0.0113$), but the 95% CI [-0.004, +0.027] includes zero, indicating no statistically significant effect. Payload 2 (format spoofing) and Payload 3 (fallback-parse trigger) show even smaller effects ($\Delta = +0.0036$ and $\Delta = -0.0027$, respectively), both with CIs firmly including zero. Criterion flip rates remain low across all payloads (1.19%–2.82%), indicating minimal criterion-level manipulation.

Hardening introduces significant benign drift. The hardened grader achieves a mean score of 0.4636 on clean inputs, representing a statistically significant benign drift of -6.42 percentage points (CI [-0.117, -0.010]) compared to the baseline. This drift persists despite optimization efforts to align the hardened prompt with baseline behavior.

No security benefit from hardening. Under injection, the hardened grader shows inflation deltas comparable to or slightly higher than the baseline (H-inject P1: $+0.0084$; P2: $+0.0141$; P3: $+0.0121$), with all CIs including zero. The hardening eliminates parse failures entirely (0.33% abstention vs 72–100% parse failures in baseline), but this technical improvement does not translate to measurable security benefit against the tested payloads.

4.3 Analysis

We hypothesize that rubric-based grading with explicit criteria may provide inherent robustness compared to pairwise comparison or holistic scoring. The judge must evaluate specific factual claims against the model response, which may make it harder for injected instructions to override the evaluation logic. The rubric structure constrains the judge's output space to binary met/not-met decisions with evidence, potentially reducing the attack surface for manipulation. However, we note that this hypothesis requires further investigation with additional attack strategies and judge models.

The evidence verification gate (Layer 4) is the likely cause of benign drift. This gate requires the judge's quoted evidence to appear in the original response with $\geq$70% fuzzy token overlap. When the judge paraphrases rather than quotes verbatim, legitimate judgments may be rejected. The 0.22% abstention rate on clean inputs suggests this mechanism occasionally triggers even on valid evaluations, and the cumulative effect across 921 criteria produces measurable score degradation.

These results demonstrate that theoretical vulnerabilities do not always translate to practical exploitability. The baseline grader's permissive parsing, while appearing vulnerable, did not enable the tested attacks to succeed. Meanwhile, the hardening intervention—designed to address these theoretical vulnerabilities—introduced measurable harm without providing security benefit. This underscores the importance of empirical validation before deploying security interventions in LLM evaluation pipelines.

5 Conclusion

We present a security audit of LiveMedBench's rubric grader against prompt injection attacks. Our key finding is a negative result: the baseline grader demonstrates natural robustness to all three tested payload families, with no statistically significant score inflation. The four-layer hardening strategy we implemented—while eliminating parse failures and enforcing evidence verification—introduces a statistically significant benign drift of -6.42% without providing measurable security benefit.

This work highlights that theoretical vulnerabilities in LLM evaluation pipelines do not always translate to practical exploitability. Security interventions should be empirically validated on both adversarial and benign conditions before deployment, as they may cause more harm than they prevent.

5.1 Limitations

Our study has several limitations. First, we tested only three hand-crafted payload families; optimization-based attacks (Shi et al., 2025) may be more effective. Second, results from Qwen2.5-72B-Instruct may not generalize to other judge models such as GPT-4 or Claude. Third, our 150-case subset may not capture rare failure modes present in the full LiveMedBench dataset. Future work should explore automated red-teaming approaches and evaluate robustness across diverse judge models.

References

Sizhe Chen, Julien Piet, Chawin Sitawarin, and David Wagner. Struq: Defending against prompt injection with structured queries. pp. 2383–2400, 2024.

Yi-Ting Hong, Huaiyuan Yao, Bolin Shen, Wanpeng Xu, Hua Wei, and Yushun Dong. Rulers: Locked rubrics and evidence-anchored scoring for robust llm evaluation. *ArXiv*, abs/2601.08654, 2026.

Muhammad Khalifa, Lajanugen Logeswaran, Jaekyeom Kim, Sungryull Sohn, Yunxiang Zhang, Moontae Lee, Hao Peng, Lu Wang, and Honglak Lee. Gaming the judge: Unfaithful chain-of-thought can undermine agent evaluation. 2026.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, S. Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. Prometheus: Inducing fine-grained evaluation capability in language models. *ArXiv*, abs/2310.08491, 2023.

Yukyung Lee, Joonghoon Kim, Jaehee Kim, Hyowon Cho, Jaewook Kang, Pilsung Kang, and Najoung Kim. Checkeval: A reliable llm-as-a-judge framework for evaluating text generation using checklists. pp. 15771–15798, 2024.

Songze Li, Chuokun Xu, Jiaying Wang, Xueluan Gong, Chen Chen, Jirui Zhang, Jun Wang, Kwok-Yan Lam, and Shouling Ji. Llms cannot reliably judge (yet?): A comprehensive assessment on the robustness of llm-as-a-judge, 2025. URL https://arxiv.org/abs/2506.09443.

Yang Liu, Dan Iter, Yichong Xu, Shuo Wang, Ruochen Xu, and Chenguang Zhu. G-eval: Nlg evaluation using gpt-4 with better human alignment. *ArXiv*, abs/2303.16634, 2023.

Narek Maloyan, Bislan Ashinov, and Dmitry Namiot. Investigating the vulnerability of llm-as-a-judge architectures to prompt-injection attacks, 2025. URL https://arxiv.org/abs/2505.13348.

Jiawen Shi, Zenghui Yuan, Yinuo Liu, Yue Huang, Pan Zhou, Lichao Sun, and Neil Zhenqiang Gong. Optimization-based prompt injection attack to llm-as-a-judge, 2025. URL https://arxiv.org/abs/2403.17710.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y. Tang, Alejandro Cuadron, Chenguang Wang, Raluca A. Popa, and Ion Stoica. Judgebench: A benchmark for evaluating llm-based judges. *ArXiv*, abs/2410.12784, 2024.

Zhiling Yan, Dingjie Song, Zhe Fang, Yisheng Ji, Xiang Li, Quanzheng Li, and Lichao Sun. Livemedbench: A contamination-free medical benchmark for llms with automated rubric evaluation. 2026.

Yulai Zhao, Haolin Liu, Dian Yu, Sunyuan Kung, Meijia Chen, Haitao Mi, and Dong Yu. One token to fool llm-as-a-judge. *ArXiv*, abs/2507.08794, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, E. Xing, Haotong Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. *ArXiv*, abs/2306.05685, 2023.