

When Critics Go Stale: Benchmarking Safety Guardrail Degradation Under Environment Drift

Fatemeh Taheri Dezaki Lakshman Kolasani Ankita Bhaumik
Siddhartha Sathymoorthy Rao Ayush Jain
Amazon

Abstract

LLM-based critics are widely deployed as safety guardrails for agentic systems, yet their reliability under realistic operating conditions remains underexplored. We demonstrate that critics calibrated against a fixed system snapshot degrade as tools are added and application domains expand, with no external signal indicating weakened detection capability. We formalize this problem along controlled axes and introduce DRIFT (**D**etecting **R**eliability **I**ssues **F**rom **T**ool-scaling), a benchmark for measuring critic robustness under environment change. Building on SafeToolBench’s multi-domain tool ecosystem, we construct verified cross-app composition scenarios, both violating and compliant, where policy violations emerge only from data flow across application boundaries, and evaluate 9 models across 6 provider families on a two-dimensional scaling matrix separating intra-app noise from cross-app noise. We find that (1) all 9 models degrade under cross-app scaling, with UPR increases ranging from +0.083 to +0.296, (2) tool scaling shifts critics toward reduced discriminability, missed violations rise while false blocks simultaneously fall, and (3) cross-app noise consistently dominates intra-app noise, with cross-app deltas positive for all nine models (sign test $p = 0.004$) while intra-app deltas are directionless (4 of 9 positive, $p = 1.0$). DRIFT provides a controlled methodology for measuring critic robustness as tool ecosystems evolve, and our results argue for monitoring UPR and FBR jointly rather than aggregate accuracy alone.

1 Introduction

LLM-powered agentic systems increasingly orchestrate complex workflows by invoking tools across multiple application domains (Yao et al., 2023; Schick et al., 2023; Qin et al., 2024; Liu et al., 2024b), from e-commerce product management (Kolasani and Taheri Dezaki, 2026) to customer service and travel booking (Yao et al., 2024).

As these systems grow in capability, *safety critics*, LLM-based guardrails that evaluate whether an agent’s proposed actions comply with domain policies, have become a standard safeguard against harmful tool use (Bai et al., 2022; Zheng et al., 2023; Yuan et al., 2024).

Consider a multi-app assistant that launches with a handful of tools spanning payments and messaging. Over successive release cycles, integrations are added for healthcare records, insurance claims, travel booking, and smart-home control; a growth pattern reflected in SafeToolBench’s 16-application ecosystem (Xia et al., 2025). A critic calibrated at launch now evaluates action sequences in an environment it was never tested against, yet no monitoring signal alerts operators that its detection capability has silently degraded. A critical but underexplored question is: **do safety critics maintain their detection capability as the environment evolves around them?**

We demonstrate that the answer is no. Through controlled experiments on 9 models spanning 6 provider families, we show that critics silently degrade under tool-scaling pressure through a specific mechanism: *reduced discriminability*. As the tool catalog grows, critics shift toward approving everything, both violations and compliant sequences. This manifests as rising Unsafe Pass Rate (UPR) alongside falling False Block Rate (FBR), confirming that critics become less discriminating rather than selectively losing detection ability.

These failure modes are particularly dangerous because they are *silent*: the critic continues to produce confident judgments while its detection rate deteriorates without external signal. In production systems where tools are regularly onboarded, this means safety guardrails can quietly become ineffective without any monitoring signal alerting operators.

To study this phenomenon systematically, we introduce DRIFT, a benchmark with three design prin-

ciples: (1) **compositional violations** that require cross-app reasoning, each individual action appears compliant in isolation; (2) **template-driven generation** with independent quality verification; and (3) a **two-dimensional scaling matrix** that separately controls intra-app and cross-app noise, enabling precise attribution of degradation sources.

Our contributions are:

1. **DRIFT benchmark:** A two-dimensional scaling matrix (intra-app $\times$ cross-app) with 1,022 verified compositional scenarios, enabling controlled measurement of critic degradation under tool-ecosystem growth. The benchmark methodology generalizes to any agentic system, practitioners can substitute their own tool catalogs and policies.
2. **Discriminability finding:** Empirical evidence across 9 models and 6 families that tool scaling shifts critics toward reduced discriminability (UPR $\uparrow$, FBR $\downarrow$ simultaneously). Cross-app deltas are positive for all nine models (mean +0.139, sign test $p = 0.004$); intra-app deltas are small and directionless (4 of 9 positive, $p = 1.0$).
3. **Deployment guidance:** Actionable mitigation strategies derived from the empirical patterns: joint UPR/FBR monitoring to detect silent permissiveness shifts, dynamic tool-scope narrowing to recover detection capability without retraining, and re-evaluation triggers at integration milestones, applicable to any production critic pipeline.

2 Related Work

Tool-use safety benchmarks. SafeTool-Bench (Xia et al., 2025) evaluates tool utilization safety across four risk categories using 16 applications and 116 APIs. Their multi-app scenarios bundle multiple risky actions but do not test *compositional* violations where each action is individually compliant. ToolSword (Ye et al., 2024) examines safety across tool learning stages. InjecAgent (Zhan et al., 2024) targets prompt injection in tool-integrated agents. R-Judge (Yuan et al., 2024) benchmarks safety risk awareness. ToolEmu (Ruan et al., 2024) emulates tool execution to identify risks. AgentSafetyBench (Zhang et al., 2024) covers 2,000+ test cases across 8 risk categories. TraceSafe (Chen et al., 2026) benchmarks guardrails on multi-step tool-calling

trajectories across 12 risk categories, finding that structural data competence matters more than semantic alignment. ToolSafe (Mou et al., 2026) proposes proactive step-level guardrails with reinforcement learning. None of these study how critic performance *degrades* as the tool ecosystem scales; they evaluate at a fixed tool-set size.

Agent guardrails. GuardAgent (Xiang et al., 2025) proposes knowledge-enabled reasoning for monitoring agent actions. Llama Guard (Inan et al., 2023) introduces LLM-based input/output safeguards. NeMo Guardrails (Rebedea et al., 2023) provides programmable guardrails. ShieldAgent (Chen et al., 2025) enforces safety policy compliance via logical reasoning over action trajectories. Safiron (Huang et al., 2025) builds a foundational pre-execution guardrail for agentic systems using synthetic data. AgentTrust (Yang, 2026) provides runtime interception for agent tool calls. These propose guardrail *architectures* but do not benchmark their degradation under environment drift.

Multi-tool planning. ToolLLM (Qin et al., 2024) provides infrastructure for 16,000+ APIs. AppBench (Wang et al., 2024a) evaluates cross-app planning. τ -bench (Yao et al., 2024) provides realistic multi-turn interactions. We build on SafeTool-Bench’s tool ecosystem but introduce a fundamentally different evaluation target: measuring how the *critic’s* judgment degrades as tools scale.

LLM-as-judge reliability. Zheng et al. (2023) establish the LLM-as-judge paradigm. Subsequent work reveals position bias (Wang et al., 2024b), length bias (Dubois et al., 2024), and limited human agreement (Shen et al., 2023). DRIFT extends this to safety critics under controlled scaling pressure.

Context scaling effects. “Lost in the Middle” (Liu et al., 2024a) shows LLMs underutilize information in long contexts. Many-shot jailbreaking (Anil et al., 2024) shows context filling degrades safety. These mechanisms explain *why* tool scaling degrades critics, more tool descriptions dilute attention over policy-relevant information.

3 The DRIFT Benchmark

3.1 Problem Formulation

Figure 1 provides an overview of the benchmark construction and evaluation methodology. A

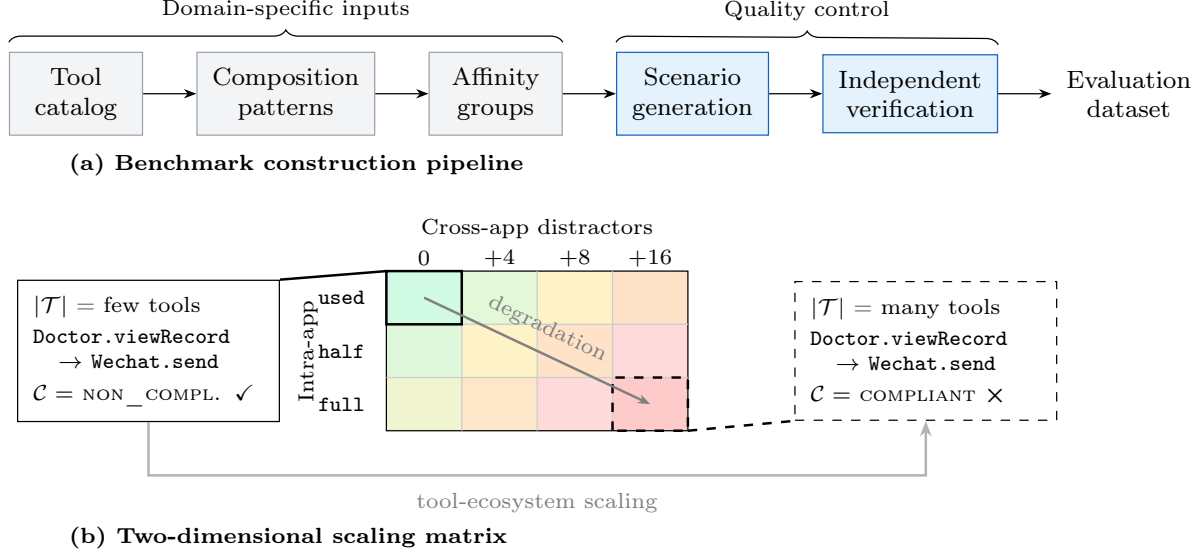


Figure 1: Overview of DRIFT. (a) Construction pipeline from domain inputs to verified dataset. (b) Scaling matrix: the same violation is detected at low tool visibility but missed as the ecosystem grows.

safety critic $\mathcal{C}$ receives a policy p , visible tools $\mathcal{T}$, and action sequence $\mathbf{a}$, outputting $\mathcal{C}(p, \mathcal{T}, \mathbf{a}) \in \{\text{COMPLIANT}, \text{NON_COMPLIANT}\}$. We measure:

$$\text{UPR} = \frac{\text{violations missed}}{\text{total violations}} \quad (\text{lower is better}) \quad (1)$$

$$\text{FBR} = \frac{\text{compliant blocked}}{\text{total compliant}} \quad (\text{lower is better}) \quad (2)$$

We also report Balanced Accuracy $\text{BA} = \frac{(1-\text{UPR}) + (1-\text{FBR})}{2}$ and Combined F1, defined as F1 on the violation class (NON_COMPLIANT as the positive label), with precision and recall computed over the pooled violation and compliant sets.

We decompose tool-scaling into two orthogonal axes:

- **Intra-app:** Adding unused APIs from the same applications involved in the scenario.
- **Cross-app:** Adding tools from entirely unrelated application domains.

3.2 Scenario Construction

We build on SafeToolBench’s 16 applications (116 APIs) spanning finance, health, transport, shopping, and smart home domains. These domains mirror real multi-app orchestration platforms where tools are onboarded incrementally and cross-app data flows create emergent policy-relevant interactions (e.g., a medical record retrieval followed by a messaging action, or a balance check followed by a cross-platform transfer).

Composition patterns. 20 interaction patterns describe neutral cross-app data flows, each with a measurable policy and clear violation/compliant boundaries.

App affinity groups. 18 groups of naturally co-occurring apps. Compatible (pattern, group) pairs are derived from API signatures, yielding 174 valid combinations.

Hardness constraints. Each action’s arguments appear innocent without context from other actions, and the sequence must actually perform the violation, not merely express intent. A separate generation model (excluded from evaluation to avoid bias) produces both violations and compliant scenarios.

Verification. An independent LLM judge (distinct from both the generator and all evaluated models) re-labels each scenario by assessing whether the action sequence concretely violates the stated policy. The generator and judge agreed on 80.7% of scenarios (825/1,022); the judge’s label was taken for the remaining 197 (167 violation→compliant, 30 compliant→violation). The judge emits a confidence label: 92.0% of scenarios are high-confidence. Human review covered all 197 disagreements plus a random sample of approximately 50 scenarios per class (~ 300 of 1,022, $\sim 29\%$); annotators judged each scenario against the policy text independently, with majority vote resolving disagreements. Final dataset: **483 violations** and **539 compliant** (1,022 total).

Intra-app	Cross-app (distractor apps)			
	0	+4	+8	+16
used_only	3	31	61	96
half_app	9	37	67	102
full_app	16	44	74	109

Table 1: Average visible tools per matrix cell.

3.3 Two-Dimensional Scaling Matrix

Each scenario is evaluated under a 3×4 matrix (Table 1). Affinity-group apps are excluded from cross-app distractors to avoid semantic contamination. Distractor sampling on each axis uses independent randomization to ensure effects are attributable to a single scaling dimension.

3.4 Generalizability

The DRIFT methodology is parameterized by three domain-specific inputs: a tool catalog, a set of policies, and an affinity mapping between co-occurring applications. Because no component of the construction process (pattern discovery, compatible-pair filtering, hardness-constrained generation, and independent verification) depends on the specific tool semantics of SafeToolBench, the framework transfers to arbitrary agentic systems. The two-dimensional decomposition (intra-app vs. cross-app) is itself domain-agnostic; it requires only a notion of application boundaries, enabling evaluation of critic robustness against system-specific growth trajectories rather than reliance on a single fixed-size benchmark.

4 Experimental Setup

We evaluate 9 models spanning 6 provider families (Table 2), representing the range of cost-efficient LLMs commonly deployed as always-on critics in production, where inference cost scales linearly with request volume. All models receive an identical static prompt instructing evaluation of the full action sequence against the policy, with structured JSON output. This is a research experiment; no models were deployed in production.

Output parsing and context limits. Critic outputs are parsed for a JSON judgment field; if JSON parsing fails, a substring fallback searches for NON_COMPLIANT/NON-COMPLIANT in the raw text. Two cells of the matrix cannot be evaluated by every model: at half_app $\times$ +16 and full_app $\times$ +16 the prompt averages 8.4–9.5k tokens, exceeding Llama 3 70B’s 8192-token context limit. To

Model	Provider	Size
Ministral 8B	Mistral	8B
Nemotron 12B	NVIDIA	12B
Mistral Small	Mistral	~22B
Gemma 3 27B	Google	27B
Nemotron 30B	NVIDIA	30B
Qwen3 32B	Qwen	32B
Kimi K2.5	Moonshot AI	32B MoE
Mixtral 8x7B	Mistral	46B MoE
Llama 3 70B	Meta	70B

Table 2: Models evaluated via a unified inference API at temperature 0.

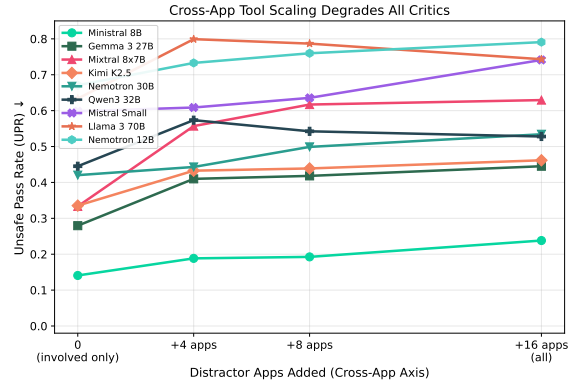


Figure 2: UPR under cross-app scaling (used_only row). All 9 models degrade as unrelated tools are added, with increases from +0.083 to +0.296.

keep per-model averages comparable, the aggregate columns of Table 3 are computed over the ten cells common to all models. Cross-App Δ and Figures 2–4 derive from the used_only row and are unaffected by this choice.

5 Results

5.1 All Models Degrade Under Cross-App Scaling

Figure 2 and Table 3 present the primary results. We observe a consistent increase in UPR from baseline to maximum scaling for all 9 models, with Cross-App Δ (UPR at +16 distractor apps minus UPR at 0) ranging from +0.083 (Qwen3 32B) to +0.296 (Mixtral 8x7B). The 95% confidence intervals between baseline and worst-case conditions are non-overlapping across all models, indicating statistically reliable degradation. Notably, no model family is immune: the effect holds across dense architectures (Ministral 8B, Llama 3 70B), mixture-of-experts models (Mixtral 8x7B, Kimi K2.5), and models from six independent providers, ruling out provider-specific artifacts.

Model	Averages		Cross-App Δ		Combined		UPR Scaling (used_only)			
	UPR	FBR	UPR	FBR	F1	BA	0	+4	+8	+16
Ministral 8B	.151	.665	+.097	-.195	.655	.592	.14	.19	.19	.24
Gemma 3 27B	.363	.315	+.166	-.163	.641	.661	.28	.41	.42	.45
Kimi K2.5	.384	.307	+.126	-.124	.629	.655	.34	.43	.44	.46
Mixtral 8x7B	.508	.326	+.296	-.269	.530	.583	.33	.56	.62	.63
Nemotron 30B	.423	.357	+.114	-.043	.584	.610	.42	.44	.50	.53
Qwen3 32B	.502	.249	+.083	-.102	.561	.625	.45	.57	.54	.53
Mistral Small	.624	.190	+.143	-.089	.473	.593	.60	.61	.64	.74
Llama 3 70B	.756	.099	+.110	-.069	.361	.573	.63	.80	.79	.74
Nemotron 12B	.708	.162	+.120	-.076	.397	.565	.67	.73	.76	.79

Table 3: Main results. Averages, F1, and BA are computed over the ten matrix cells evaluable by every model, excluding half_app $\times$ +16 and full_app $\times$ +16 (see §4). Cross-App Δ = UPR change from 0 to +16 distractor apps (used_only row). All models show positive UPR Δ and negative FBR Δ , confirming reduced discriminability.

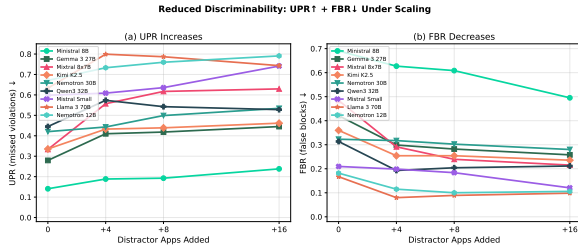


Figure 3: UPR rises (left) while FBR falls (right) under cross-app scaling, revealing reduced discriminability.

5.2 Tool Scaling Reduces Discriminability

Joint analysis of UPR and FBR reveals a consistent pattern across all models (Figure 3): tool scaling does not selectively impair violation detection; rather, it reduces overall discriminability. Every model exhibits negative FBR Cross-App Δ (range: -0.043 to -0.269) concurrent with positive UPR Δ , indicating a systematic shift toward permissive judgments on both violating and compliant inputs. This co-movement has an important methodological implication: aggregate accuracy metrics can remain stable while the critic’s operating point shifts toward missed violations, rendering standard evaluation insufficient for detecting this failure mode.

5.3 Cross-App Noise Dominates Intra-App Noise

To isolate the relative contribution of each scaling axis, we compare Cross-App Δ to Intra-App Δ (UPR at full_app minus UPR at used_only, with cross-app held at 0). Cross-app deltas are positive for all nine models (mean $+0.139$, sign test $p = 0.004$). Intra-app deltas are small and directionless: 4 of 9 are positive, with a mean near zero (sign test $p = 1.0$). This asymmetry is not an artifact of the different numbers of tools added per axis (~ 93 cross-app vs. ~ 13 intra-app): normalized per

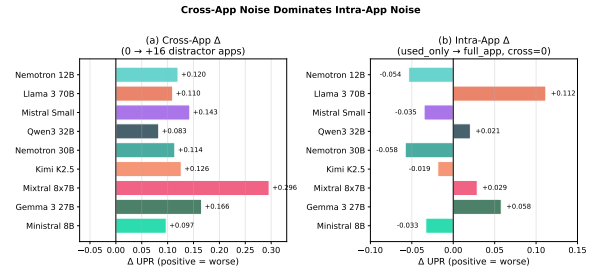


Figure 4: Cross-app degradation (left) vs intra-app (right). Cross-app dominates for models with positive deltas on both axes.

tool added, cross-app slopes remain positive for all nine models ($+0.089$ to $+0.318$ UPR per 100 tools) while intra-app slopes remain directionless. The asymmetry also does not reflect ceiling effects from high baseline UPR: expressed as a fraction of remaining headroom, cross-app degradation is positive for all nine models (mean 26%, range 11–44%), and baseline UPR does not predict delta magnitude (Spearman $\rho = -0.20$, $n = 9$). The pattern suggests that semantically unrelated tool descriptions are substantially more disruptive than additional APIs from familiar domains, consistent with an attention-dilution account where novel vocabulary competes for the model’s attention over policy-relevant data flows.

5.4 Full Scaling Matrix

Figure 5 visualizes the complete 3×4 UPR matrix for each model, confirming that degradation concentrates along the cross-app axis across all model families.

6 Discussion

Why cross-app noise hurts more. The asymmetry between axes can be understood through the

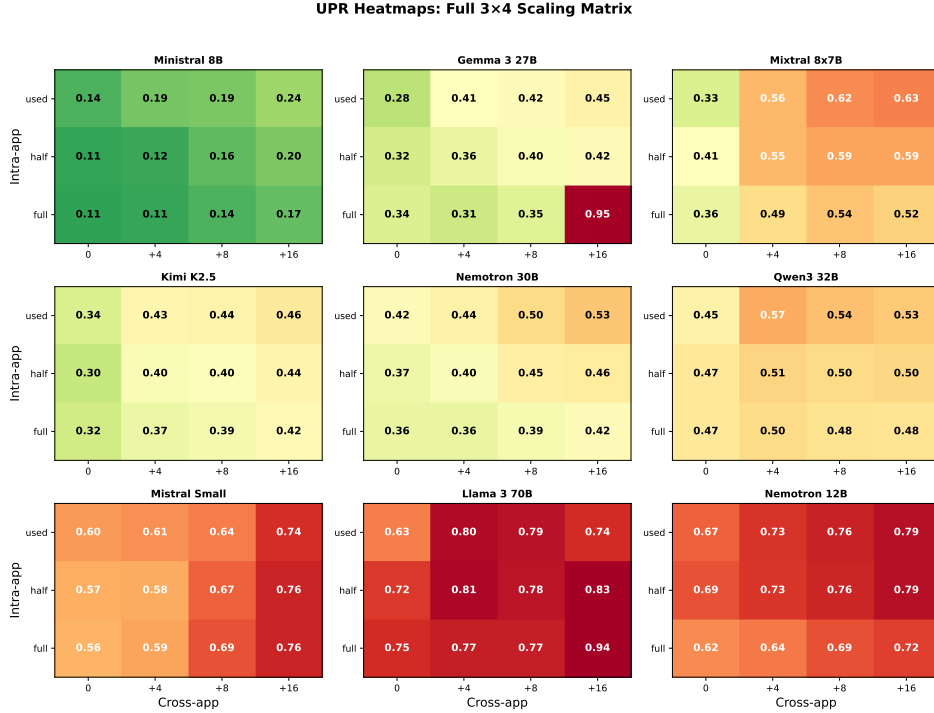


Figure 5: Per-model UPR heatmaps. Green = low UPR (good); red = high (bad). Degradation is primarily horizontal (cross-app).

lens of contextual interference (Liu et al., 2024a): intra-app tools share lexical and functional overlap with the relevant APIs, contributing minimal novel information to the context window. Cross-app tools, by contrast, introduce semantically disjoint descriptions that compete for the model’s attention budget without contributing signal relevant to the policy under evaluation. The monotonic degradation observed in §5 is consistent with an attention-dilution mechanism where the ratio of policy-relevant to irrelevant tokens decreases as cross-app tools are added.

The discriminability shift. The universal co-occurrence of $\text{UPR}\uparrow$ and $\text{FBR}\downarrow$ indicates that critics do not become selectively worse at one class; rather, they lose their decision boundary under context pressure, defaulting toward permissive judgments. From a safety-systems perspective, this is the more dangerous failure mode: silent under-detection is harder to detect operationally than noisy over-blocking, which generates user complaints.

Implications for deployment. Our findings suggest three actionable strategies. First, *joint metric monitoring*: the discriminability pattern ($\text{UPR}\uparrow$, $\text{FBR}\downarrow$) is invisible to systems that track only over-

all accuracy; production systems should monitor both metrics over time, particularly after tool onboarding events, where a drop in FBR without corresponding UPR improvement signals increasing permissiveness rather than better calibration. Second, *dynamic tool-scope narrowing*: comparing the used_only baseline to worst-case cells shows that restricting the critic’s visible tools to only those relevant to the current action sequence can substantially recover detection capability without model retraining; this requires only filtering the tool catalog before critic invocation. Third, *re-evaluation at scale milestones*: our data shows that cross-app additions of 4+ unrelated domains produce measurable degradation for vulnerable models, suggesting periodic re-evaluation after integration milestones rather than continuous testing.

Future work. A natural extension is drift-aware repair, iteratively patching the critic specification based on observed failure patterns. Policy drift, conversation depth, and intent distribution shifts represent additional axes worth investigating.

7 Conclusion

We introduced DRIFT, a benchmark revealing how LLM safety critics silently degrade under tool-scaling pressure. Across 9 models and 6 provider

families, we find universal degradation: every model’s UPR increases while FBR decreases as unrelated tools are added, confirming reduced discriminability rather than selective detection loss. Cross-app deltas are positive for all nine models (sign test $p = 0.004$) while intra-app deltas are directionless ($p = 1.0$), confirming that the axis of scaling matters, not just the total token count. DRIFT provides an evaluation framework for this failure mode, and we recommend that production systems adopt joint UPR/FBR monitoring, dynamic tool-scope narrowing, and periodic re-evaluation at integration milestones to detect this silent degradation before it impacts end users.

Limitations

We evaluate only the tool-scaling axis of drift; policy revision, conversation depth, and intent distribution shifts are additional axes worth investigating. Our scenarios are synthetically generated from SafeToolBench’s tool ecosystem rather than extracted from production traces; while independently verified and constructed to mirror realistic multi-app workflows, real deployments may exhibit additional complexity from user-specific context and long conversation histories. All critics receive a static zero-shot prompt; retrieval-augmented configurations that dynamically surface relevant policy excerpts may exhibit different resilience characteristics.

Ethical Considerations

This work studies failure modes in safety guardrails with the goal of improving, not circumventing, defensive mechanisms. We disclose the vulnerability of silent degradation under tool scaling because awareness is a prerequisite for mitigation; practitioners cannot defend against failure modes they do not know exist. All scenarios in DRIFT involve synthetic tool interactions over public API schemas from SafeToolBench; no real user data, personal information, or production traces are used. The policies and violations are constructed to represent realistic safety boundaries (e.g., data sharing limits, transaction caps) without encoding actual customer policies. We evaluate only publicly available open-weight models and do not expose proprietary model internals or deployment configurations. This was a research experiment; no models were deployed in production systems as part of this study. The SafeToolBench API schemas are used under their

original CC BY 4.0 license; all evaluated open-weight models are used under their respective publicly available licenses (Apache 2.0, MIT, Llama 3 Community License, and Qwen License, as applicable).

References

- Cem Anil, Esin Durmus, and 1 others. 2024. Many-shot jailbreaking. *arXiv preprint*.
- Yuntao Bai, Saurav Kadavath, and 1 others. 2022. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.
- Yen-Shan Chen, Sian-Yao Huang, Cheng-Lin Yang, and Yun-Nung Chen. 2026. Tracesafe: A systematic assessment of LLM guardrails on multi-step tool-calling trajectories. *arXiv preprint arXiv:2604.07223*.
- Zhaorun Chen, Mintong Kang, and Bo Li. 2025. Shield-agent: Shielding agents via verifiable safety policy reasoning. *arXiv preprint arXiv:2503.22738*.
- Yann Dubois, Balazs Galambosi, and 1 others. 2024. Length-controlled alpacaEval: A simple way to debias automatic evaluators. In *COLM*.
- Yue Huang, Hang Hua, Yujun Zhou, and 1 others. 2025. Building a foundational guardrail for general agentic systems via synthetic data. *arXiv preprint arXiv:2510.09781*.
- Hakan Inan, Kartikeya Upasani, and 1 others. 2023. Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*.
- Lakshman Kolasani and Fatemeh Taheri Dezaki. 2026. Progressive fine-tuning for cost-effective structured attribute generation in e-commerce. In *ACL Industry Track*.
- Nelson F. Liu, Kevin Lin, and 1 others. 2024a. Lost in the middle: How language models use long contexts. *TACL*.
- Xiao Liu, Hao Yu, Hanchen Zhang, and 1 others. 2024b. Agentbench: Evaluating llms as agents. In *ICLR*.
- Yutao Mou, Zhangchi Xue, Lijun Li, Peiyang Liu, Shikun Zhang, Wei Ye, and Jing Shao. 2026. Tool-safe: Enhancing tool invocation safety of LLM-based agents via proactive step-level guardrail and feedback. *arXiv preprint arXiv:2601.10156*.
- Yujia Qin, Shihao Liang, Yining Ye, and 1 others. 2024. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *ICLR*.
- Traian Rebedea, Razvan Dinu, and 1 others. 2023. Nemo guardrails: A toolkit for controllable and safe llm applications with programmable rails. In *EMNLP (Demo)*.

- Yangjun Ruan, Honghua Dong, and 1 others. 2024. Identifying the risks of llm agents with an llm-emulated sandbox. In *ICLR*.
- Timo Schick, Jane Dwivedi-Yu, and 1 others. 2023. Toolformer: Language models can teach themselves to use tools. In *NeurIPS*.
- Chenhui Shen, Liying Cheng, and 1 others. 2023. Large language models are not yet human-level evaluators for abstractive summarization. In *Findings of EMNLP*.
- Hongru Wang, Rui Wang, and 1 others. 2024a. App-bench: Planning of multiple apis from various apps for complex user instruction. In *Proceedings of EMNLP*.
- Peiyi Wang, Lei Li, and 1 others. 2024b. Large language models are not fair evaluators. In *ACL*.
- Hongfei Xia, Hongru Wang, Zeming Liu, Qian Yu, Yuhang Guo, and Haifeng Wang. 2025. Safetool-bench: Pioneering a prospective benchmark to evaluating tool utilization safety in LLMs. *arXiv preprint arXiv:2509.07315*.
- Zhen Xiang, Linzhi Zheng, and 1 others. 2025. Guardagent: Safeguard llm agents by a guard agent via knowledge-enabled reasoning. In *ICML*.
- Chenglin Yang. 2026. Agenttrust: Runtime safety evaluation and interception for AI agent tool use. *arXiv preprint arXiv:2605.04785*.
- Shunyu Yao, Noah Shinn, and 1 others. 2024. τ -bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*.
- Shunyu Yao, Jeffrey Zhao, and 1 others. 2023. React: Synergizing reasoning and acting in language models. In *ICLR*.
- Junjie Ye, Sixian Li, and 1 others. 2024. Toolsword: Unveiling safety issues of large language models in tool learning across three stages. In *Proceedings of ACL*.
- Tongxin Yuan, Zhiwei He, Lingzhong Dong, Yiming Wang, Ruijie Zhao, Tian Xia, Lizhen Xu, Binglin Zhou, Fangqi Li, Zhuosheng Zhang, Rui Wang, and Gongshen Liu. 2024. R-judge: Benchmarking safety risk awareness for LLM agents. In *Findings of the Association for Computational Linguistics: EMNLP 2024*.
- Qiusi Zhan, Zhixiang Liang, and 1 others. 2024. Injecagent: Benchmarking indirect prompt injections in tool-integrated large language model agents. In *Findings of ACL*.
- Zhexin Zhang, Shiyao Cui, and 1 others. 2024. Agentsafetybench: Evaluating the safety of llm agents. *arXiv preprint arXiv:2412.14470*.
- Lianmin Zheng, Wei-Lin Chiang, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *NeurIPS*.