PT Jurnal Cendekia Indonesia
Journal of Information Systems and Business Technology
Homepage: https://journal.jci.co.id/jisbt
Vol. 02 No. 04 (2026) Page: 111-121
E-ISSN: 3109-8886

Continual Guardrail Learning for Tool-Using LLM Agents: Cross-Benchmark Jailbreak Detection, Indirect Prompt-Injection Filtering, and Utility Preservation

[1]Hao Ran, [2]Natalie Foster, [3]Bo Liang, [4]Justin Meyer

[1]Department of Computer Science, Stony Brook University, Stony Brook, NY, USA
[2]Department of Computer Science, Michigan State University, East Lansing, MI, USA
[3]Department of Computer Science, Rutgers University–New Brunswick, Piscataway, NJ, USA
[4]Department of Computer Science, University of Massachusetts Amherst, Amherst, MA, USA

Email: [1*]hran_sbu@outlook.com

Abstract

Tool-using large language model agents process untrusted text while holding permissions to communicate, book services, and move funds. This study evaluates lightweight guardrails under changing attacks while preserving authorized task utility. The text study uses all 6,899 saved runs in one internally consistent AgentDojo v1 GPT-4o pipeline and 200 JailbreakBench prompts covering 100 behavior groups. Deduplication produced 5,280 texts: 1,450 attacks and 3,830 benign records. Injection-goal grouping assigned 3,169, 1,056, and 1,055 records to training, validation, and testing without sharing an AgentDojo attack target. A normalized word-character classifier achieved 85.43% F1, 90.31% recall, 7.96% benign false-positive rate, and 0.9613 ROC-AUC. A 1,000-round group bootstrap gave a wide 59.71-97.68% F1 interval. Cross-source recall fell to 33.00% from AgentDojo to JailbreakBench and 0.16% in the reverse direction. Sequential tests compared frozen, benign-anchored, hard-replay, and soft-label-replay learners. Reanalysis of 629 matched AgentDojo attacks showed that tool filtering reduced targeted attack success from 47.69% to 6.84% while increasing safe utility from 29.73% to 52.62%. Layered guardrails are therefore necessary: replay maintains coverage under shift, while tool-level enforcement provided the strongest observed end-to-end security-utility balance.

Keywords: LLM Agents; Jailbreak Detection, Indirect Prompt Injection, Continual Learning; Guardrails, Tool Security, Utility Preservation, Agentdojo, Jailbreakbench.

Abstrak

Agen model bahasa besar (LLM) yang menggunakan alat memproses teks yang tidak tepercaya sembari memiliki izin untuk berkomunikasi, memesan layanan, dan memindahkan dana. Studi ini mengevaluasi mekanisme pengaman (guardrails) yang ringan dalam menghadapi serangan yang terus berubah, sembari tetap mempertahankan kegunaan tugas yang diizinkan. Studi teks ini menggunakan seluruh 6.899 eksekusi tersimpan dari satu alur kerja AgentDojo v1 GPT-4o yang konsisten secara internal serta 200 prompt JailbreakBench yang mencakup 100 kelompok perilaku. Proses penghapusan duplikat menghasilkan 5.280 teks: 1.450 serangan dan 3.830 catatan yang aman (benign). Pengelompokan berdasarkan tujuan injeksi membagi catatan menjadi 3.169 untuk pelatihan, 1.056 untuk validasi, dan 1.055 untuk pengujian, tanpa adanya tumpang tindih target serangan AgentDojo. Pengklasifikasi kata-karakter yang dinormalisasi mencapai skor F1 85,43%, recall 90,31%, tingkat positif palsu (false-positive rate) untuk data aman sebesar 7,96%, dan nilai ROC-AUC 0,9613. Metode bootstrap kelompok sebanyak 1.000 putaran menghasilkan rentang skor F1 yang lebar, yaitu 59,71–97,68%. Nilai recall lintas sumber turun menjadi 33,00% saat beralih dari AgentDojo ke JailbreakBench dan menjadi 0,16% pada arah sebaliknya. Pengujian sekuensial membandingkan metode pembelajaran frozen, benign-anchored, hard-replay, dan soft-label-replay. Analisis ulang terhadap 629 serangan AgentDojo yang cocok menunjukkan bahwa penyaringan pada tingkat alat (tool filtering) menurunkan tingkat keberhasilan serangan terarah dari 47,69% menjadi 6,84%, sekaligus meningkatkan kegunaan yang aman dari 29,73% menjadi 52,62%. Oleh karena itu, mekanisme pengaman berlapis diperlukan: metode replay mempertahankan cakupan saat terjadi perubahan kondisi, sementara penegakan aturan pada tingkat alat memberikan keseimbangan keamanan-kegunaan end-to-end terbaik yang teramati.

Kata kunci: Agen LLM; Deteksi Jailbreak; Injeksi Prompt Tidak Langsung; Pembelajaran Berkelanjutan (Continual Learning); Mekanisme Pengaman (Guardrails); Keamanan Alat; Pemeliharaan Kegunaan; AgentDojo; JailbreakBench.

A. INTRODUCTION

Large language models become consequential when applications let them call tools: one context can mix

Diterima Redaksi: 01-08-2026 | Selesai Revisi: 06-08-2026 | Diterbitkan Online: 07-08-2026

privileged instructions, a user goal, and untrusted email, transaction, or web data. Malicious tool content may then be read as a command. This command-data ambiguity drives indirect prompt injection [1] and creates privacy and integrity risks exposed by agent risk cards [2].

Evaluation must distinguish direct jailbreaks from indirect injections encountered during authorized work. JailbreakBench standardizes behavioral jailbreak testing [3], whereas AgentDojo measures utility and targeted attack success in stateful tool environments [4]. InjecAgent, ToolEmu, Agent Security Bench, and AgentHarm extend coverage to tool exposure, sandboxed consequences, and multi-step malicious tasks [5], [6], [7], [8]. A formal threat model makes attack and defense assumptions comparable [9]; output toxicity alone is insufficient.

Attack distributions also change. Transferable suffixes [10], iterative black-box attacks [11], and readable AutoDAN prompts [12] differ from in-the-wild recombinations [13]. HarmBench improves grading consistency [14], while StrongREJECT exposes empty refusals [15]. Continual red-teaming therefore couples discovery with online guardrail updates [16], and safe retraining requires drift signals and rollback controls [17]. A deployed guardrail must retain prior coverage without disabling ordinary tasks.

Defenses operate at multiple layers. Spotlighting marks untrusted spans [18]; StruQ separates instructions from data [19]; instruction-hierarchy training prioritizes privileged messages [20]; Task Shield checks relevance [21]; and CaMeL enforces provenance and capabilities outside the model [22]. Evidence-based self-verification [23] and multi-stage refusal with utility preservation [24] complement these controls, but adaptive attacks still bypass defaults [25]. Decoding, refusal-loss, representation, perturbation, and moderator models [26], [27], [28], [29], [30], [31] therefore require joint security-utility evaluation.

Evidence grounding offers a complementary control surface. Private-operations RAG [32], bilingual incident triage [33], word-level provenance [34], and narrative-aware claim verification [35] separate support from fluent output; evidence-constrained monitoring agents also make tool use auditable [36]. We use JailbreakBench for jailbreak strings and AgentDojo for benign tasks, poisoned tool outputs, target actions, and trajectories, analyzing text detection separately from archived defenses.

Four questions guide the evaluation: held-out-goal detection, cross-suite and cross-benchmark transfer, adaptation without forgetting, and authorized task completion. High ROC-AUC is not deployable when false blocking is costly. XSTest [37], evidence-calibrated abstention [38], model-risk calibration [39], rejection-aware tiering [40], confidence fusion [41], and trust-calibrated multilingual RAG [42] illustrate selective-decision controls.

The study contributes a leakage-controlled corpus from a complete 6,899-run AgentDojo v1 pipeline and two JailbreakBench artifacts; calibrated lexical and contextual models; continual replay; and matched defense analysis. Experience and dark replay motivate the updates [43], [44]. Controlled memory [45] motivates lifecycle monitoring across trajectory reliability [46], NIST guidance [47], hybrid-cloud deployment [48], and retrieval-quality prediction [49].

B. METHODS

Data Sources and Archive Scope

JailbreakBench package 1.0.0, commit 23dbdf6b1965052160445622 9bc1d9c4156c85c1 [50], supplied 100 GCG prompts each for Llama-2 and Vicuna; prompts at one behavior index shared a group. Because responses and per-prompt outcomes were absent, these strings measured detector recall, not jailbreak success [3].

AgentDojo 0.1.35, commit 089ed468cf3ed0322acc66b0211f26d9d90dbf60 [51], contained 36,679 JSON trajectories. Its v1 matrix has 97 user tasks, 27 injection goals, and 629 compatible cases across four suites [4]. Detector and continual tests used the complete, error-free 6,899-run gpt-4o-2024-05-13 v1 directory; other directories supported defense and model comparisons.

The selected directory held 27,752 tool messages, 26,309 nonempty. Text was positive when it contained an exact payload from the run's injections map (3,188 occurrences); other tool text and 97 unique user prompts were benign. NFKC, lowercasing, and whitespace collapse preceded deduplication, with positive precedence on conflicts. Exact matching can miss semantic transformations.

Deduplication yielded 5,080 AgentDojo texts: 1,250 attack-bearing outputs and 3,830 benign records. Adding 200 JailbreakBench artifacts produced 5,280 texts (1,450 attack; 3,830 benign); the manifest separately records 36,679 trajectories and 105 errors.

Table 1 Composition Of The Deduplicated Text Corpus By Source, Domain, And Class.

Source	Domain	Class	Texts	Groups	Median chars	P95 chars
AgentDojo injection	Banking	Attack	387	9	586.0	1,134.7
AgentDojo injection	Slack	Attack	267	5	468.0	1,100.7
AgentDojo injection	Travel	Attack	504	7	1,005.0	2,680.0
AgentDojo injection	Workspace	Attack	92	5	1,184.5	2,832.8
AgentDojo tool	Banking	Benign	157	18	595.0	1,340.6
AgentDojo tool	Slack	Benign	27	10	169.0	964.8
AgentDojo tool	Travel	Benign	1,454	26	1,215.5	3,591.3
AgentDojo tool	Workspace	Benign	2,095	43	1,307.0	9,086.2
AgentDojo user	Banking	Benign	16	16	77.5	238.8
AgentDojo user	Slack	Benign	21	21	115.0	219.0

Source	Domain	Class	Texts	Groups	Median chars	P95 chars
AgentDojo user	Travel	Benign	20	20	429.0	634.1
AgentDojo user	Workspace	Benign	40	40	139.0	324.9
JailbreakBench	Jailbreakbench	Attack	200	100	183.0	233.1

Table 2 Agentdojo V1 Task And Injection Matrix.

Suite	User tasks	Injection goals	Compatible security tests
Banking	16	9	144
Slack	21	5	105
Travel	20	7	140
Workspace	40	6	240
Total	97	27	629

Leakage-Controlled Splits and Threat Model

Stratified group folds used seed 20240804. AgentDojo positives shared suite-goal groups; benign outputs shared suite-task groups; paired JailbreakBench prompts shared behavior groups. Train/validation/test contained 3,169/1,056/1,055 records and 871/290/289 attacks. Groups, canonical texts, and AgentDojo goals never crossed partitions, yielding target-held-out evaluation within known families.

Dense task-by-goal matrices preclude holding out both axes within a suite. The primary split prioritized goals; a complementary split grouped authorized prompts and removed cross-partition text collisions, testing unseen contexts with known goal families.

The adversary controlled tool-returned text or a direct jailbreak artifact. The guardrail observed that text and, for contextual tests, the authorized prompt, blocking scores above a validation-selected threshold. At agent level, `security=true` means malicious-goal completion and contributes to ASR.

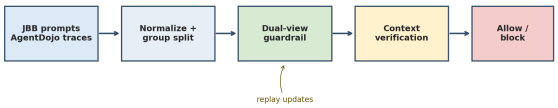

Figure 1 Experimental Pipeline Separating Grouped Text Detection, Continual Updates, And Archived Agent-Level Evaluation.

Guardrail Models

Seven methods were tested: a keyword rule; a length heuristic; word TF-IDF (12,000 features); within-word character 3-5-grams (18,000); an 8,000-word/12,000-character dual view; its punctuation-sanitized variant; and a word-character probability ensemble. Learned models used class-balanced logistic regression (C=2.0). The low-latency design parallels compact IoT filters [52] and language-guided intrusion features [53]; autoencoder traffic screening provides a label-light contrast [54]. As classifiers rather than generative reviewers, these methods trade coverage against false blocking and overhead [55].

A balanced logistic task-tool verifier combined payload probability, prompt-text TF-IDF similarity, rule score, and log length ratio. Grouping and collision removal yielded 3,682/634/667 pairs from 61/21/19 context groups with no context or candidate overlap. It is a four-feature verifier, not a second LLM [21]; budgeted multi-hop retrieval [56] and evidence-chain provenance [57] are complementary.

Thresholds, Metrics, and Statistical Analysis

A 0.002-grid validation sweep maximized recall subject to FPR <=5%, with F1 as tie-breaker. Metrics were accuracy, balanced accuracy, precision, recall, F1, ROC-AUC, FPR, false-negative rate, Brier score, and ten-bin ECE. Latency used seven repetitions; 1,000 group-bootstrap rounds produced descriptive 95% intervals. Calibration and uncertainty served as decision controls, as in confidence fusion [41], model-risk calibration [39], rejection-aware tiering [40], calibrated capacity forecasts [58].

Cross-domain tests held out one suite; cross-source tests trained on one attack source plus benign text and measured recall on the other. Both retained grouped validation and held-out benign FPR, probing adaptive attacks [25], shared-feature domain bridging [59], and cross-cloud shift [60]. Conformal demand envelopes provide a bounded-risk analogue [61].

Continual Attack Replay and Distillation

Six episodes covered direct/ignore-previous, authority-framed, identity-perturbed, agent-aware, denial-of-service, and JailbreakBench attacks. Groups split 75/25; benign groups formed fit, calibration, and test pools. A 65,536-dimensional character hash supported frozen, benign-anchored, hard-replay, and soft-label-replay learners. Hard replay kept observed positives plus matched benign memory; soft replay trained a regularized student on an averaged logistic teacher. Fixed calibration data set thresholds; untouched tests measured current/cumulative recall and benign FPR. The design joins replay [43], [44], continual red-team updates [16], controlled forgetting [45], and rollback-aware drift handling [17].

Agent-Level Utility and Security Evaluation

Five archived GPT-4o-2024-05-13 pipelines were matched: no defense, repeated user prompt, spotlighting, tool filtering, and a transformer injection detector. Each covered 97 clean tasks and 629 attacks. Clean/attack utility measured authorized completion, ASR malicious-goal completion, and safe utility completion without attack success. Clean failures were not called refusals without causal trace evidence.

Metrics were recomputed by domain, across ten 629-run attack families, and for 18 complete undefended model matrices. Five pipelines contained saved errors, so Spearman clean-utility/ASR association used pessimistic scoring and an error-excluded sensitivity check. This view complements trajectory forecasting [46], CloudTrail stage

smoothing [62], DevOps command-safety evaluation [63], and CI repair [64]. Evidence-constrained agents that separate detection from source-faithful tool use provide a governance analogue [36]. Duration was descriptive.

Reproducibility Controls

One command with seed 20240804 generated transformations, splits, models, tables, and figures under Python 3.12.13, NumPy 2.3.5, pandas 2.2.3, and scikit-learn 1.8.0. SHA-256 manifests cover inputs and outputs. Separate result objects distinguish detection, continual updates, and archived interventions. This supports NIST measurement [47], reproducible log-sequence modeling [65], and deterministic HDFS incident narratives [66].

C. RESULTS AND DISCUSSION

Corpus Structure and Primary Detection

Attacks were 27.46% of 5,280 texts: 1,250 AgentDojo injections, 3,830 benign records, and 200 JailbreakBench attacks. Travel contributed 1,454 benign/504 attack texts; workspace contributed 2,095/92. Workspace benign text had a 9,086-character 95th percentile versus JailbreakBench's 183-character median, making length unreliable.

At threshold 0.332, the sanitized dual-view classifier led at 85.43% F1. Its 1,055 tests yielded TN/FP/FN/TP counts of 705/61/28/261, 90.31% recall, 81.06% precision, 7.96% FPR, 91.17% balanced accuracy, and 0.9613 ROC-AUC. Group-bootstrap intervals were 59.71-97.68% F1, 64.55-100% recall, and 1.64-27.22% FPR. Median scoring took 2.29 ms; Brier score was 0.0652 and ECE 3.52%.

The static rule had no positive-recall point under the 5% validation-FPR limit; length missed all attacks and caused six false positives. F1/recall was 83.31%/85.47% for word TF-IDF, 83.84%/86.16% for characters, and 85.01%/89.27% for the unsanitized dual view; the ensemble reached 84.19% F1. Normalization added 0.43 F1 and 1.04 recall points. Learned test FPRs of 7.31-7.96% exceeded the validation constraint.

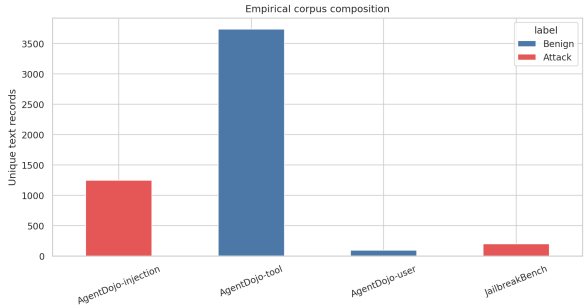

Figure 2 Class and source composition of the 5,280-text empirical corpus.

Table 3 Primary Held-Out Detector Performance, Thresholds Were Selected On Validation Data

Method	Threshold	Precision	Recall	F1	ROC-AUC	FPR	Latency (ms/item)
Static keyword rule	1.000	0.00%	0.00%	0.00%	0.6894	0.00%	0.105
Length heuristic	0.548	0.00%	0.00%	0.00%	0.6186	0.78%	0.000
Word TF-IDF classifier	0.432	81.25%	85.47%	83.31%	0.9438	7.44%	0.178
Character TF-IDF classifier	0.436	81.64%	86.16%	83.84%	0.9553	7.31%	1.651
Dual-view classifier	0.340	81.13%	89.27%	85.01%	0.9630	7.83%	2.452
Sanitized dual-view classifier	0.332	81.06%	90.31%	85.43%	0.9613	7.96%	2.289
Two-view probability ensemble	0.412	81.09%	87.54%	84.19%	0.9510	7.70%	2.965

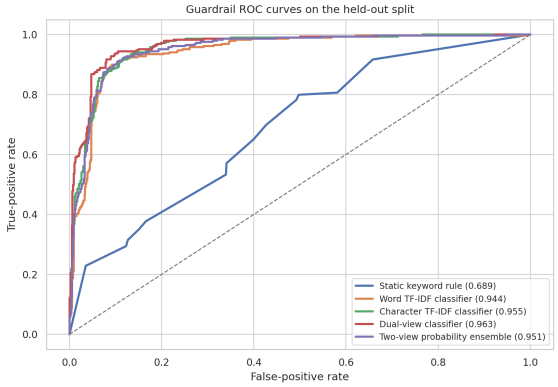

Figure 3 Held-Out Receiver Operating Characteristic Curves For Learned Text Guardrails

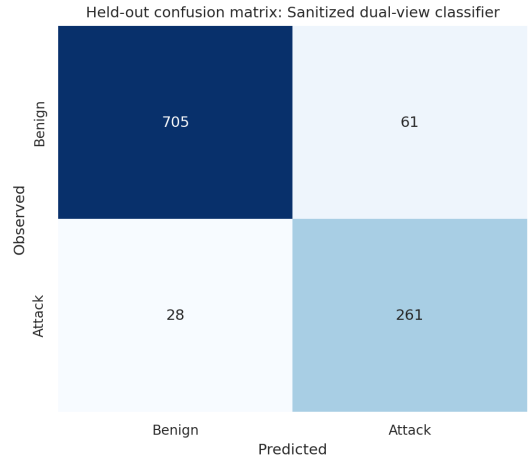

Figure 4 Confusion Matrix Of The Sanitized Dual-View Classifier At Threshold 0.332.

Domain and Source Transfer

Held-out domains transferred poorly: recall/FPR was 76.74%/53.18% for banking, 44.57%/81.25% for Slack, 86.31%/72.86% for travel, and 80.43%/51.52% for workspace. Slack ROC-AUC was 0.3887, so moving a threshold could not repair schema shift. This echoes cross-cloud shift [60] and motivates shared-feature analysis rather than assumed invariance [59].

AgentDojo-to-JailbreakBench recall was 33.00% at 2.22% benign FPR; the reverse direction recalled only 0.16% at 0.39% FPR. Direct jailbreaks and payload-bearing tool responses are distinct surfaces, so coverage requires both [10], [11], [12].

Table 4 Cross-Domain And Cross-Source Transfer At Validation-Calibrated Thresholds.

Held-out test	N	Threshold	Recall	FPR	ROC-AUC
Hold out Banking	560	0.360	76.74 %	53.18 %	0.6342
Hold out Slack	315	0.234	44.57 %	81.25 %	0.3887
Hold out Travel	1,978	0.444	86.31 %	72.86 %	0.7638
Hold out Workspace	2,227	0.238	80.43 %	51.52 %	0.7118
AgentDojo → JailbreakBench	966	0.350	33.00 %	2.22%	—
JailbreakBench → AgentDojo	2,016	0.636	0.16%	0.39%	—

Context Features and Calibration

On 667 context- and text-disjoint pairs, payload-only screening reached 96.00% F1, 98.11% recall, 1.97% FPR, and 0.9981 ROC-AUC. Adding prompt similarity, instruction cues, and length ratio yielded 96.25% F1, 96.86% recall, and 1.38% FPR; Brier score improved 0.0192 to 0.0153 and ECE 5.87% to 2.08%. This supports calibrated routing, as in selective fusion [41], evidence-grounded abstention [38], and multilingual answerability control [42], but does not prove provenance.

Table 5 User-Context- And Text-Disjoint Evaluation Of Payload-Only And Task-Tool Verification.

Method	Threshold	Recall	FPR	F1	ROC-AUC	Brier	ECE
Payload-only classifier	0.286	98.11%	1.97 %	96.00%	0.9981	0.0192	5.87 %
Context-feature verifier	0.148	96.86%	1.38 %	96.25%	0.9983	0.0153	2.08 %

Continual Learning Trade-offs

After fitting the initial direct-attack episode, all four learners had 89.31% current recall and 4.88% benign FPR. Authority-framed attacks exposed immediate shift: before updating, every learner recalled only 16.67%. After updating, all adaptive methods reached 100% current recall, but cumulative recall was 32.21% anchored, 74.50% hard replay, and 81.21% soft replay; FPRs were 8.26%, 13.43%, and 8.26%. This quantifies the retention problem addressed by continual red-team updates [16].

Before identity-perturbation updating, current recall was 95.00% anchored, 76.25% hard replay, and 58.75% soft replay; each reached 100% afterward. All three also reached 100% on agent-aware attacks, while hard replay retained 93.55% cumulative recall at 13.15% FPR. Denial-of-service recall was 56.10%, 100%, and 97.56% (N=41), showing why controlled memory policy matters [45].

Before the final JailbreakBench update, recall was 8% frozen, 6% anchored, 38% hard replay, and 20% soft replay; afterward the adaptive scores were 100%, 100%, and 88%. Final cumulative recall was 48.11%, 71.08%, 89.46%, and 72.70%; FPRs were 4.88%, 4.13%, 10.23%, and 7.23%. Hard replay maximized coverage, soft replay lowered its FPR by three points, and anchoring blocked least but forgot more. No updater dominated the XSTest trade-off [37], reinforcing rollback-aware monitoring [17].

Table 6 Continual-Learning Outcomes After Update: Replay Trajectories And Final-Method Comparison.

Episode	Method	Current recall	Seen recall	Benign FPR	Current N
Initial direct attacks	Hard-label replay	89.31%	89.31 %	4.88%	131
Authority-framed injection	Hard-label replay	100.00 %	74.50 %	13.43 %	18
Identity perturbations	Hard-label replay	100.00 %	86.46 %	10.33 %	80
Agent-aware attacks	Hard-label replay	100.00 %	93.55 %	13.15 %	50
Denial-of-service lures	Hard-label replay	100.00 %	90.00 %	10.14 %	41

Episode	Method	Current recall	Seen recall	Benign FPR	Current N
JailbreakBench artifacts	Hard-label replay	100.00 %	89.46 %	10.23 %	50
Initial direct attacks	Soft-label replay	89.31%	89.31 %	4.88%	131
Authority-framed injection	Soft-label replay	100.00 %	81.21 %	8.26%	18
Identity perturbations	Soft-label replay	100.00 %	75.55 %	9.30%	80
Agent-aware attacks	Soft-label replay	100.00 %	85.30 %	9.48%	50
Denial-of-service lures	Soft-label replay	97.56%	81.88 %	10.23 %	41
JailbreakBench artifacts	Soft-label replay	88.00%	72.70 %	7.23%	50
JailbreakBench artifacts	Frozen guardrail	8.00%	48.11 %	4.88%	50
JailbreakBench artifacts	Anchored online update	100.00 %	71.08 %	4.13%	50

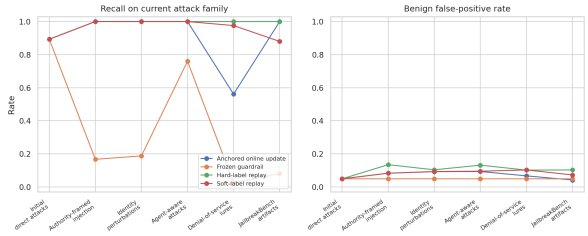

Figure 5 Recall And Benign False-Positive Rate As Attack Families Arrive Sequentially.

Archived End-to-End Defense Outcomes

Undefended GPT-4o achieved 69.07% clean utility, 50.08% attack utility, 47.69% ASR, and 29.73% safe utility. Tool filtering reached 72.16%, 56.28%, 6.84%, and 52.62%: an 85.67% relative ASR reduction and 22.89-point safe-utility gain. This supports capability-aware enforcement [21], [22] and utility-aware resistance [24]. Prompt repetition led clean/attack utility at 84.54%/67.25% but retained 27.82% ASR; safe utility was 52.62%. Spotlighting yielded 72.16%/55.64% utility and 41.65% ASR. The transformer detector cut ASR to 7.95% but clean/safe utility to 41.24%/19.24%. Low ASR can conceal poor usability, motivating evidence-based self-verification [23].

Median attacked-run duration was 6.60 s for tool filtering and 8.30 s for prompt repetition, with service and response length uncontrolled. Local scoring took 2.29 ms; end-to-end latency also includes routing, tools, inference, and review. DevOps command-safety work examines the same guidance-action boundary [63].

Table 7 Matched GPT-4o End-To-End Defense Outcomes On 97 Clean Tasks And 629 Attacks.

Defense	Clean utility	Attack utility	Targeted ASR	Safe utility	Median duration (s)
None	69.07 %	50.08 %	47.69%	29.73 %	7.13
Repeat user prompt	84.54 %	67.25 %	27.82%	52.62 %	8.30
Spotlighting with delimiters	72.16 %	55.64 %	41.65%	37.20 %	7.44
Tool filter	**72.16 %**	**56.28 %**	**6.84%**	**52.62 %**	**6.60**
Prompt-injection detector	41.24 %	21.14 %	7.95%	19.24 %	7.63

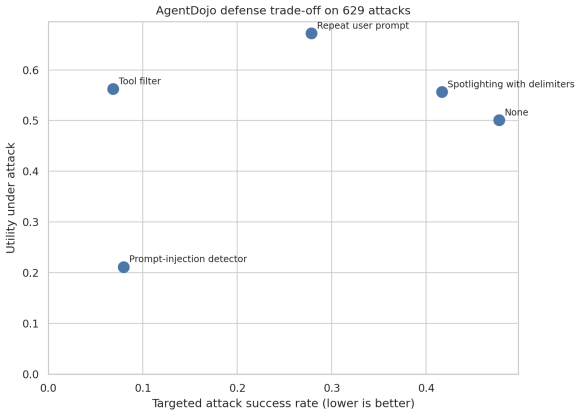

Figure 6 Utility Under Attack Versus Targeted Attack Success For Matched GPT-4o Defenses

Domain, Attack-Family, and Model Heterogeneity

Undefended ASR was 92.38% in Slack, 62.50% banking, 40.42% workspace, and 11.43% travel. Tool filtering reduced these to 8.57%, 11.11%, 4.17%, and 5.71%; safe utility was 42.86%, 47.92%, 51.67%, and 66.43%. The transformer detector reached zero travel ASR but 16.43% safe utility. Attack-stage modeling [62], diagnostic routing [67], and host-intrusion narratives [68] aid diagnosis.

Direct payload, ignore-previous, and InjecAgent ASRs were 3.66%, 5.41%, and 5.72%. Important-instructions reached 47.69%; removing names retained 44.83-46.10%, while wrong model/user names reduced ASR to 23.69%/23.21%. Tool knowledge reached 34.50%, confirming family sensitivity [25] and the value of trajectory reliability forecasts [46].

Across 18 undefended models, clean utility and ASR correlated at Spearman rho=0.480 (p=0.0436); excluding saved errors strengthened rho to 0.566 (p=0.0144). Claude-3.5-Sonnet-20241022 combined 79.38% clean utility, 1.11% ASR, and 71.54% safe utility; GPT-4-0125-preview recorded 65.98%, 56.28%, and 18.92%. This noncausal association shows that benign utility alone did not optimize security.

Table 8 Domain-Specific Targeted ASR And Safe Utility. Each Cell Reports ASR / Safe Utility.

Defense	Banking	Slack	Travel	Workspace
None	62.5% / 22.9%	92.4% / 6.7%	11.4% / 63.6%	40.4% / 24.2%
Repeat user prompt	46.5% / 41.7%	60.0% / 27.6%	7.1% / 69.3%	14.6% / 60.4%
Spotlighting with delimiters	61.8% / 30.6%	80.0% / 15.2%	3.6% / 72.1%	35.0% / 30.4%
Tool filter	11.1% / 47.9%	8.6% / 42.9%	5.7% / 66.4%	4.2% / 51.7%
Prompt-injection detector	0.7% / 31.2%	12.4% / 14.3%	0.0% / 16.4%	15.0% / 15.8%

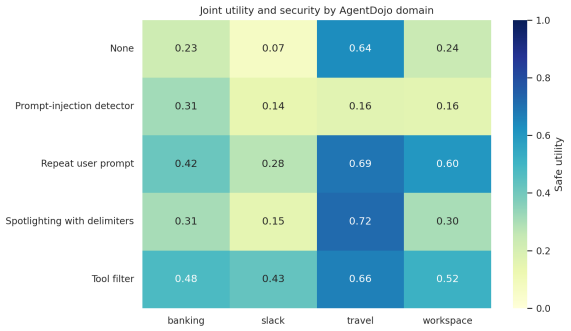

Figure 7 Domain-Level Attack Success And Safe Utility Across Archived Defenses.

Table 9 Undefended GPT-4o Results Across Ten Complete Attack Families.

Attack family	N	Attack utility	Targeted ASR	Safe utility	Median duration (s)
Direct	629	67.25%	3.66%	65.98%	6.13
Ignore previous	629	66.77%	5.41%	64.23%	5.99
Important instructions	629	50.08%	47.69%	29.73%	7.13
No model name	629	51.83%	46.10%	31.48%	7.15
No names	629	49.60%	45.79%	30.05%	6.71
No user name	629	52.15%	44.83%	33.23%	7.15
Wrong model name	629	65.82%	23.69%	53.90%	6.29
Wrong user name	629	65.02%	23.21%	52.15%	5.96
InjecAgent	629	68.52%	5.72%	67.09%	5.94
Tool knowledge	629	57.71%	34.50%	42.13%	6.37

Table 10 Representative Undefended Model Outcomes On The Complete Important-Instructions Matrix.

Model	Clean utility	Attack utility	Targeted ASR	Safe utility	Errors C/A
Claude 3.5 Sonnet (2024-10-22)	79.38%	72.50%	1.11%	71.54%	0 / 0
Claude 3 Opus (2024-02-29)	68.04%	52.46%	11.29%	46.90%	0 / 0
Command R	26.80%	30.84%	3.34%	30.21%	1 / 3
Gemini 1.5 Pro 002	61.86%	47.06%	17.01%	38.31%	0 / 0
GPT-3.5 Turbo 0125	35.05%	34.66%	10.33%	31.80%	1 / 12
GPT-4 0125 Preview	65.98%	40.70%	56.28%	18.92%	0 / 0
GPT-4o (2024-05-13)	69.07%	50.08%	47.69%	29.73%	0 / 0
Llama 3 70B Chat	34.02%	18.28%	25.60%	13.99%	6 / 35

Note: Across all 18 complete file matrices, clean utility and targeted ASR had Spearman rho = 0.480 (p = 0.0436); after excluding error-bearing runs, rho = 0.566 (p = 0.0144).

Interpretation and Limitations

The evidence favors layers. The dual-view classifier handled held-out goals, but cross-source and domain transfer depended on training surface. Updates recovered current attacks while replay traded cumulative recall against FPR. Tool filtering gave the best agent-level balance; a separate detector imposed substantial utility cost. Prompt reminders [69] do not replace enforcement.

JailbreakBench supplied prompts rather than responses, so its result is detector recall, not jailbreak success. Text guardrails were offline; end-to-end claims cover only matched archived defenses. Exact matching can miss semantic transformations, episode order is one curriculum, and leakage-controlled splits do not prove adaptive robustness. Retrieved normal prototypes [70], incident cards [71], and multi-source fault attribution [72] illustrate post-detection evidence.

Operationally, screen untrusted tool content before context construction and route ambiguity instead of authorizing sensitive actions. Cost-aware routing and admission control [73] can reserve deeper review for costly decisions; conformal alert control bounds escalation volume [74]. Reviewed attacks enter replay, followed by benign recalibration and prior-family regression tests. Monitoring must combine utility, ASR, safe utility, FPR, worst-domain recall, errors, permissions, audit, and incident response [47]. Privacy/integrity cards [2], hybrid-cloud planning [48], and scientific-search evidence cards [75] extend the loop beyond the classifier.

D. CONCLUSIONS

Continual guardrail learning improved coverage of changing attack families, but utility preservation depended on how memory was used. On 5,280 unique texts derived from JailbreakBench and a complete 6,899-run AgentDojo v1 pipeline, the best observed target-held-out detector reached 85.43% F1, 90.31% recall, and 7.96% benign FPR. Its cross-benchmark and held-out-domain transfer were poor, making static deployment inappropriate. Sequential

updates restored current-family coverage, while cumulative recall and benign blocking separated anchored, hard-replay, and soft-label strategies.

The archived agent runs established the complementary end-to-end result. Tool filtering reduced GPT-4o targeted ASR from 47.69% to 6.84% and increased safe utility from 29.73% to 52.62%, whereas the transformer detector achieved similar ASR with only 19.24% safe utility. The strongest operational design is therefore not a single universal classifier. It is a monitored stack that combines fast text screening, provenance-aware tool controls, calibrated escalation, continual attack replay, and explicit measurement of benign task completion.

E. REFERENCES

[1] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, "Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection," in Proc. ACM Workshop on Artificial Intelligence and Security, pp. 79-90, 2023, doi: 10.1145/3605764.3623985.

[2] W. Su, H. Rao, and E. Ma, "Privacy and data-integrity risk cards for LLM agents: A UI/UX design framework for secure human oversight under prompt-injection attacks," Int. J. Graph. Des., vol. 4, no. 1, pp. 186-191, Apr. 2026, doi: 10.51903/ijgd.v4i1.3699.

[3] P. Chao et al., "JailbreakBench: An Open Robustness Benchmark for Jailbreaking Large Language Models," in Advances in Neural Information Processing Systems, vol. 37, 2024.

[4] E. Debenedetti, J. Zhang, M. Balunovic, L. Beurer-Kellner, M. Fischer, and F. Tramer, "AgentDojo: A Dynamic Environment to Evaluate Prompt Injection Attacks and Defenses for LLM Agents," in Advances in Neural Information Processing Systems, vol. 37, 2024.

[5] Q. Zhan, Z. Liang, Z. Ying, and D. Kang, "InjecAgent: Benchmarking Indirect Prompt Injections in Tool-Integrated Large Language Model Agents," in Findings of the Association for Computational Linguistics: ACL 2024, pp. 10471-10506, 2024, doi: 10.18653/v1/2024.findings-acl.624.

[6] Y. Ruan et al., "Identifying the Risks of LM Agents with an LM-Emulated Sandbox," in Proc. International Conference on Learning Representations, 2024.

[7] H. Zhang et al., "Agent Security Bench (ASB): Formalizing and Benchmarking Attacks and Defenses in LLM-based Agents," in Proc. International Conference on Learning Representations, 2025.

[8] M. Andriushchenko et al., "AgentHarm: A Benchmark for Measuring Harmfulness of LLM Agents," in Proc. International Conference on Learning Representations, 2025.

[9] Y. Liu, Y. Jia, R. Geng, J. Jia, and N. Z. Gong, "Formalizing and Benchmarking Prompt Injection Attacks and Defenses," in Proc. 33rd USENIX Security Symposium, pp. 1831-1847, 2024.

[10] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, and M. Fredrikson, "Universal and Transferable Adversarial Attacks on Aligned Language Models," arXiv:2307.15043, 2023.

[11] P. Chao, A. Robey, E. Dobriban, H. Hassani, G. J. Pappas, and E. Wong, "Jailbreaking Black Box Large Language Models in Twenty Queries," in Proc. International Conference on Learning Representations, 2024.

[12] X. Liu, N. Xu, M. Chen, and C. Xiao, "AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models," in Proc. International Conference on Learning Representations, 2024.

[13] L. Jiang et al., "WildTeaming at Scale: From In-the-Wild Jailbreaks to (Adversarially) Safer Language Models," in Advances in Neural Information Processing Systems, vol. 37, 2024.

[14] M. Mazeika et al., "HarmBench: A Standardized Evaluation Framework for Automated Red Teaming and Robust Refusal," in Proc. 41st International Conference on Machine Learning, vol. 235, pp. 35181-35224, 2024.

[15] A. Souly et al., "A StrongREJECT for Empty Jailbreaks," in Advances in Neural Information Processing Systems, vol. 37, 2024.

[16] D. Zheng, C. Li, and H. Davidson, "Continual red-teaming for in-the-wild jailbreaks via online guardrail updates and guardrail distillation," J. Adv. Comput. Syst., vol. 3, no. 2, pp. 35-49, Feb. 2023, doi: 10.69987/JACS.2023.30203.

[17] H. Zhang, "DriftGuard: Multi-signal drift early warning and safe re-training/rollback for CTR/CVR models," J. Adv. Comput. Syst., vol. 3, no. 7, pp. 24-40, Jul. 2023, doi: 10.69987/JACS.2023.30703.

[18] K. Hines, G. Lopez, M. Hall, F. Zarfati, Y. Zunger, and E. Kiciman, "Defending Against Indirect Prompt Injection Attacks with Spotlighting," arXiv:2403.14720, 2024.

[19] S. Chen, J. Piet, C. Sitawarin, and D. Wagner, "StruQ: Defending Against Prompt Injection with Structured Queries," in Proc. 34th USENIX Security Symposium, pp. 2383-2400, 2025.

[20] E. Wallace, K. Xiao, R. Leike, L. Weng, J. Heidecke, and A. Beutel, "The Instruction Hierarchy: Training LLMs to Prioritize Privileged Instructions," arXiv:2404.13208, 2024.

[21] F. Jia, T. Wu, X. Qin, and A. Squicciarini, "The Task Shield: Enforcing Task Alignment to Defend Against Indirect Prompt Injection in LLM Agents," in Proc. 63rd Annual Meeting of the Association for Computational Linguistics, pp. 29680-29697, 2025, doi: 10.18653/v1/2025.acl-long.1435.

[22] E. Debenedetti et al., "Defeating Prompt Injections by Design," arXiv:2503.18813, 2025.

[23] D. Zheng, B. Zhang, and J. Geibel, "VerifySafe: Toxicity-safe agent responses under adversarial

prompts with evidence-based self-verification," J. Adv. Comput. Syst., vol. 4, no. 1, pp. 67-82, Jan. 2024, doi: 10.69987/JACS.2024.40106.

[24] D. Zheng and C. Li, "Behavior-level jailbreak resistance via multi-stage refusal + utility preservation," J. Adv. Comput. Syst., vol. 4, no. 1, pp. 83-99, Jan. 2024, doi: 10.69987/JACS.2024.40107.

[25] Q. Zhan, R. Fang, H. S. Panchal, and D. Kang, "Adaptive Attacks Break Defenses Against Indirect Prompt Injection Attacks on LLM Agents," in Findings of the Association for Computational Linguistics: NAACL 2025, pp. 7116-7132, 2025, doi: 10.18653/v1/2025.findings-naacl.395.

[26] Z. Xu, F. Jiang, L. Niu, J. Jia, B. Y. Lin, and R. Poovendran, "SafeDecoding: Defending against Jailbreak Attacks via Safety-Aware Decoding," in Proc. 62nd Annual Meeting of the Association for Computational Linguistics, pp. 5587-5605, 2024, doi: 10.18653/v1/2024.acl-long.303.

[27] X. Hu, P.-Y. Chen, and T.-Y. Ho, "Gradient Cuff: Detecting Jailbreak Attacks on Large Language Models by Exploring Refusal Loss Landscapes," in Advances in Neural Information Processing Systems, vol. 37, 2024.

[28] A. Zou et al., "Improving Alignment and Robustness with Circuit Breakers," in Advances in Neural Information Processing Systems, vol. 37, 2024.

[29] A. Robey, E. Wong, H. Hassani, and G. J. Pappas, "SmoothLLM: Defending Large Language Models Against Jailbreaking Attacks," Transactions on Machine Learning Research, 2025.

[30] S. Han et al., "WildGuard: Open One-stop Moderation Tools for Safety Risks, Jailbreaks, and Refusals of LLMs," in Advances in Neural Information Processing Systems, vol. 37, 2024.

[31] H. Inan et al., "Llama Guard: LLM-based Input-Output Safeguard for Human-AI Conversations," arXiv:2312.06674, 2023.

[32] B. Zhang, H. Rao, and D. Zhao, "Evidence-grounded RAG for cloud-native DevOps: Hallucination-resistant AIOps question answering over private operations documents," J. Adv. Comput. Syst., vol. 4, no. 3, pp. 109-125, Mar. 2024, doi: 10.69987/JACS.2024.40308.

[33] [33] G. Liu, C. Li, and E. Zhang, "OpsLLM for cloud incident triage: Bilingual RAG-based root cause analysis and alert summarization for AI infrastructure operations," J. Adv. Comput. Syst., vol. 4, no. 4, pp. 97-111, Apr. 2024, doi: 10.69987/JACS.2024.40408.

[34] C. Li, J. Bai, and S. Wang, "Evidence-chain reliable RAG: Word-level hallucination detection, source attribution, and provenance explanation for LLM applications," J. Adv. Comput. Syst., vol. 4, no. 2, pp. 76-92, Feb. 2024, doi: 10.69987/JACS.2024.40207.

[35] W. Su, S. Chen, and E. Qian, "Narrative-aware scientific claim verification agent with evidence ranking for ClimateCheck," J. Technol. Informatics Eng., vol. 5, no. 1, pp. 327-340, Apr. 2026, doi: 10.51903/jtie.v5i1.549.

[36] S. Zhou, Y. Chen, and K. Lee, "Accounting-aware evidence-constrained agents for disclosure, settlement, and secondary-market risk monitoring in tokenized RWA infrastructure," J. Technol. Informatics Eng., vol. 5, no. 2, pp. 60-74, Aug. 2026, doi: 10.51903/jtie.v5i2.544.

[37] P. Rottger, H. Kirk, B. Vidgen, G. Attanasio, F. Bianchi, and D. Hovy, "XSTest: A Test Suite for Identifying Exaggerated Safety Behaviours in Large Language Models," in Proc. NAACL 2024, pp. 5377-5400, 2024, doi: 10.18653/v1/2024.naacl-long.301.

[38] Z. S. Zhong, J. Chen, E. Zhong, and X. Sun, "Evidence-calibrated RAG for unanswerable question answering: Retrieval coverage, abstention calibration, and hallucination-proxy analysis on SQuAD 2.0," J. Technol. Informatics Eng., vol. 4, no. 2, pp. 502-520, Aug. 2025, doi: 10.51903/jtie.v4i2.536.

[39] J. Jin, T. Huang, and S. Lu, "A model-risk-friendly probability of default workflow: Calibration, distribution-free uncertainty quantification, and SHAP explanations on the UCI credit card default dataset," J. Adv. Comput. Syst., vol. 4, no. 6, pp. 74-85, Jun. 2024, doi: 10.69987/JACS.2024.40606.

[40] Y. Chen, Y. Zhang, D. Chau, and M. Sherman, "Credit card default risk tiering with probability calibration and uncertainty-driven rejection: A reproducible study on the UCI credit card clients dataset," J. Adv. Comput. Syst., vol. 3, no. 4, pp. 31-47, Apr. 2023, doi: 10.69987/JACS.2023.30403.

[41] Q. Xin, "Uncertainty-aware late fusion for 3D perception: Confidence calibration and fusion rule learning," J. Technol. Informatics Eng., vol. 4, no. 1, pp. 215-238, Apr. 2025, doi: 10.51903/jtie.v4i1.485.

[42] Y. Chen and H. Xu, "Trust-calibrated multilingual RAG for humanitarian information platforms: Empirical evaluation on OMoS-QA for migration information access," Int. J. Graph. Des., vol. 4, no. 1, pp. 141-164, Apr. 2026, doi: 10.51903/ijgd.v4i1.3552.

[43] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, and G. Wayne, "Experience Replay for Continual Learning," in Advances in Neural Information Processing Systems, vol. 32, pp. 350-360, 2019.

[44] P. Buzzega, M. Boschini, A. Porrello, D. Abati, and S. Calderara, "Dark Experience for General Continual Learning: A Strong, Simple Baseline," in Advances in Neural Information Processing Systems, vol. 33, pp. 15920-15930, 2020.

[45] [45] X. Sun, Y. Lu, and J. Chen, "Controllable long-term user memory for multi-session dialogue: Confidence-gated writing, time-aware retrieval-augmented generation, and update/forgetting," J. Adv. Comput. Syst., vol. 3, no. 8, pp. 9-24, Aug. 2023, doi: 10.69987/JACS.2023.30802.

[46] Z. S. Zhong, C. Li, and H. Rao, "Trajectory reliability prediction for generalist AI agents: Tool-use failure analysis and success forecasting on ZClawBench," J. Technol. Informatics Eng., vol. 5, no. 1, pp. 341-360, Apr. 2026, doi: 10.51903/jtie.v5i1.539.

[47] C. Autio et al., Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile, NIST AI 600-1, National Institute of Standards and Technology, 2024, doi: 10.6028/NIST.AI.600-1.

[48] Q. Xin, "Hybrid cloud architecture for efficient and cost-effective large language model deployment," J. Inf. Syst. Informatics, vol. 7, no. 3, pp. 2182-2195, Sep. 2025, doi: 10.51519/journalisi.v7i3.1170.

[49] R. Zhang, Z. Wen, C. Wang, C. Tang, P. Xu, and Y. Jiang, "Quality analysis and evaluation prediction of RAG retrieval based on machine learning algorithms," arXiv:2511.19481, Nov. 2025, doi: 10.48550/arXiv.2511.19481.

[50] JailbreakBench, "JailbreakBench v1.0.0," GitHub release, Jun. 13, 2024. [Online]. Available: https://github.com/JailbreakBench/jailbreakbench/releases/tag/v1.0.0

[51] ETH Zurich SPY Lab, "AgentDojo v0.1.35," GitHub release, Oct. 27, 2025. [Online]. Available: https://github.com/ethz-spylab/agentdojo/releases/tag/v0.1.35

[52] S. Lu and D. Zhou, "TinyLLM-assisted intrusion detection for real-time IoT networks," J. Adv. Comput. Syst., vol. 4, no. 8, pp. 72-87, Aug. 2024, doi: 10.69987/JACS.2024.40809.

[53] Y. Li and S. Lu, "Language-guided feature selection for DDoS and intrusion detection on CICIDS2017," J. Technol. Informatics Eng., vol. 4, no. 1, pp. 284-305, Apr. 2025, doi: 10.51903/jtie.v4i1.531.

[54] Q. Xin, Z. Xu, L. Guo, F. Zhao, and B. Wu, "IoT traffic classification and anomaly detection method based on deep autoencoders," Appl. Comput. Eng., vol. 69, pp. 64-70, Jul. 2024, doi: 10.54254/2755-2721/69/20241511.

[55] M. Sharma et al., "Constitutional Classifiers: Defending against Universal Jailbreaks across Thousands of Hours of Red Teaming," arXiv:2501.18837, 2025.

[56] W. Su, S. Chen, and C. Zhao, "Budgeted multi-hop retrieval agent for compositional question answering: A retrieval-policy evaluation on the official MultiHop-RAG benchmark," J. Technol. Informatics Eng., vol. 4, no. 3, pp. 649-662, Dec. 2025, doi: 10.51903/jtie.v4i3.543.

[57] J. Jin, "Evidence-chain reliable RAG: Hallucination detection, source attribution, and deterministic provenance explanations," J. Technol. Informatics Eng., vol. 4, no. 2, pp. 520-533, Aug. 2025, doi: 10.51903/jtie.v4i2.535.

[58] S. He, H. Tu, and I. Liu, "Safe PD capacity forecasting with time-series foundation models and calibrated uncertainty for heterogeneous GPU clusters," J. Adv. Comput. Syst., vol. 3, no. 4, pp. 48-66, Apr. 2023, doi: 10.69987/JACS.2023.30404.

[59] Z. S. Zhong, X. Pan, and Q. Lei, "Bridging domains with approximately shared features," in Proc. 28th Int. Conf. Artificial Intelligence and Statistics, PMLR, vol. 258, pp. 559-567, 2025.

[60] S. He, X. Chang, and E. Sun, "Cross-cloud transfer learning for AI training capacity forecasting under workload and topology distribution shift," J. Adv. Comput. Syst., vol. 4, no. 1, pp. 100-120, Jan. 2024, doi: 10.69987/JACS.2024.40108.

[61] S. Chen, S. He, and E. Sun, "Risk-bounded GPU resource oversubscription via conformal demand envelopes in production AI clusters," J. Adv. Comput. Syst., vol. 4, no. 5, pp. 119-134, May 2024, doi: 10.69987/JACS.2024.40509.

[62] J. Bai, S. Chen, D. Zheng, and M.-J. Kuo, "Interpretable attack-chain stage detection from AWS CloudTrail event sequences via linear models and HMM smoothing," Inf. Electr. Electron. Eng., vol. 6, no. 1, pp. 28-43, May 2026, doi: 10.33474/infotron.v6i1.24923.

[63] B. Zhang, X. Sun, G. Liu, and B. Zhou, "LLM-style DevOps copilot for cloud-native troubleshooting: Retrieval-augmented runbook generation and command-safety evaluation," J. Technol. Informatics Eng., vol. 5, no. 2, pp. 104-118, Aug. 2026, doi: 10.51903/jtie.v5i2.534.

[64] H. Zhang, "LLM-driven CI failure diagnosis and automated repair: From GitHub Actions logs to patch recommendation," J. Technol. Informatics Eng., vol. 4, no. 1, pp. 190-214, Apr. 2025, doi: 10.51903/jtie.v4i1.484.

[65] Q. Xin, "Self-supervised log anomaly detection with LogBERT-style transformers: Full empirical evaluation on a reproducible SynHDFS benchmark," J. Electr. Eng. Comput. Sci., vol. 11, no. 1, pp. 23-35, May 2026, doi: 10.54732/jeecs.v11i1.3.

[66] X. Sun, Z. S. Zhong, and Q. Wu, "Retrieval-grounded HDFS log anomaly detection and deterministic failure narrative generation," J. Comput. Syst. Appl., vol. 3, no. 1, pp. 15-30, Jun. 2026, doi: 10.64229/j6d7fr94.

[67] C. Li, G. Liu, and Z. Zhao, "Cost-aware LLM-style routing for AIOps log analysis: Log parsing, anomaly detection, fault diagnosis, and incident summarization on LogEval task files," J. Technol. Informatics Eng., vol. 5, no. 2, pp. 91-103, Aug. 2026, doi: 10.51903/jtie.v5i2.538.

[68] Q. Xin, "Host-based intrusion detection with system call sequences: Window localization and forensic narratives," AVITEC, vol. 8, no. 2, pp. 325-334, Aug. 2026, doi: 10.28989/avitec.v8i2.3973.

[69] [69] Y. Xie, J. Yi, J. Shao, J. Curl, L. Lyu, Q. Chen, X. Xie, and F. Wu, "Defending ChatGPT against Jailbreak Attack via Self-Reminders," Nature Machine Intelligence, vol. 5, pp. 1486-1496, 2023, doi: 10.1038/s42256-023-00765-8.

[70] [70] Q. Xin, "Explaining OpenStack failure-injection log anomalies with retrieved normal prototypes," Emerg. Inf. Sci. Technol., vol. 6, no. 2, pp. 125-146, Nov. 2025, doi: 10.18196/eist.v6i2.31232.

[71] [71] J. Nie, G. Liu, C. Li, and T. Zou, "Evidence-constrained incident visualization cards for distributed cloud logs: A UI/UX framework for turning Hadoop, OpenStack, and ZooKeeper logs into actionable SRE design interfaces," Int. J. Graph. Des., vol. 4, no. 1, pp. 179-185, Apr. 2026, doi: 10.51903/ijgd.v4i1.3703.

[72] [72] G. Liu, S. He, and I. Liu, "LLM-augmented multi-source root cause attribution for CPU and network faults in microservices," J. Adv. Comput. Syst., vol. 3, no. 6, pp. 39-57, Jun. 2023, doi: 10.69987/JACS.2023.30604.

[73] [73] S. Zhao, Y. Ren, and X. Chang, "Profit-aware spot GPU admission control with cost-sensitive loss and evidence-grounded policy memos for AI workload supply-demand matching," J. Technol. Informatics Eng., vol. 5, no. 2, pp. 45-59, Aug. 2026, doi: 10.51903/jtie.v5i2.545.

[74] [74] Q. Xin, "Log anomaly detection with conformal alert control and evidence-grounded incident ticket generation," AVITEC, vol. 8, no. 2, pp. 247-264, Aug. 2026, doi: 10.28989/avitec.v8i2.3974.

[75] [75] J. Jin, "LLM-style evidence cards for scientific search interfaces: A UI/UX design framework for retrieval transparency, ranking trust, and visual evidence hierarchy," Int. J. Graph. Des., vol. 3, no. 2, pp. 397-414, Oct. 2025, doi: 10.51903/ijgd.v3i2.3698..