Runtime Safety Mechanisms for Clinical Large Language Model Agents: A Review and Empirical Comparison

Antony Pradeesh D[1]

[1]Lincoln University College, Malaysia

pdf.antonypradeesh@lincoln.edu.my

Abstract: Language models (LLMs) that reason and act through tools are close to clinical deployment, where an irreversible action of prescribing an inappropriate drug cannot be undone by reasoning. This paper explores if any of the existing mechanisms can mitigate this risk. We review four categories of defense in the model, context, orchestration, and action planes of the agent stack, grouping them against five categories of known failures. The review discovers that the defense is abundant in the model and context planes and entirely absent in the action plane that is the only place where any irreversible harm happens, as well as no existing solution generates a deterministic decision log necessary for clinical audit. Finally, we compare the reviewed methods against the data from 1,846 agent instances programatically evaluated for six different base models. Deterministic filtering of tool calls resulted in 67.1% reduction of unsafe clinical actions and elimination of prompt injection attacks for all models tested, while context plane intervention was effective in reverse direction.

Keywords: Agentic AI; Large language models; Runtime safety; Prompt injection; Clinical decision support.

Introduction

Large language models have evolved from question-answering systems to action-performing systems. An agent breaks down the goal, calls external tools, witnesses the effects of its actions, and iterates until it deems the goal achieved. Protocol standardization has hastened this shift while also narrowing the attack surface, since a standardized interface implies a standardized set of actions an agent can be prompted to perform [1]. In medicine, this development is furthest along and least forgiving: reviews of autonomous medical agents describe a field moving from reactive prediction toward agentic action [2], [3].

The issue addressed by this review is not simply that models can make mistakes, but that some incorrect actions cannot be reversed. A misclassification produces a result that a clinician can disregard; an erroneous order, by contrast, enters the medication record, and subsequent reasoning cannot undo it. Such actions can arise through three mechanisms: adversarially induced hallucination in decision support [4], compliance with instructions embedded in untrusted tool outputs [5], and loss of coherence across long trajectories [6]. Within a clinical workflow, each pathway can ultimately lead to the same outcome: an irreversible tool call that should never have been executed.

This paper reviews the runtime safety measures proposed to date, organizes them according to the layer of the agent stack where they intervene, and identifies the gap that each approach leaves unresolved.

Related work

Four broad classes of defence dominate the literature. Verifier-LLM oversight uses a second model to assess the output of the first and is the most prominent approach proposed for clinical applications [3]. Representation-level interruption acts on the model's internal states to stop harmful computation. Information-flow control restricts which information can affect particular actions and provides the

strongest available guarantees. Context and memory management aims to improve the quality of an agent's trajectory through mechanisms such as bounded working memory and transactional validation [7].

Each class faces a structural barrier to clinical deployment rather than simply a lack of engineering effort. Because the verifier is itself a language model, it remains vulnerable to the hallucination and prompt-injection failures it is intended to prevent. It also roughly doubles inference cost at each step and produces judgements that are not reproducible, making them unsuitable as clinical audit records [8]. Representation-based approaches require access to the model's internal weights, which rules out the hosted closed models used in most clinical deployments. Information-flow control requires the agent to be redesigned around a custom interpreter. Context management can improve the agent's reasoning process but does not impose a constraint at the point where an action is executed [7].

Security research further demonstrates that the underlying vulnerability persists. Indirect prompt injection can bypass prompt-level filtering because attacks may arrive through carriers that the filter does not account for [5]. In multi-agent systems, injected content can also propagate through internal communication channels [9]. Deterministic runtime verification offers categorical guarantees, but existing implementations have been developed primarily for industrial control systems with predefined action spaces rather than for dynamically generated tool calls [10]. Table 1 compares the four defence families reviewed here.

Table 1. Comparison of runtime safety mechanisms for LLM agents against the requirements of clinical deployment

Approach	**Training-free**	**Black-box models**	**Gates actions**	**No 2nd-model cost**	**Audit trail**
Verifier-LLM oversight [3], [8]	Yes	Yes	Partial	No	Partial
Representation interruption	No	No	Partial	Yes	No
Information-flow control	Yes	Partial	Partial	Yes	Yes
Context / memory management [7]	Partial	Yes	No	Yes	No
Prompt-level filtering [5]	Yes	Yes	No	Yes	Partial
Runtime verification (control) [10]	Yes	Yes	Yes	Yes	Yes
Deterministic action gating (this work)	Yes	Yes	Yes	Yes	Yes

Key Contribution

This review makes three contributions to the existing literature. First, it categorizes runtime safety methods according to the layer of the agent stack where they intervene, revealing a pattern that has not previously been articulated: interventions are concentrated at the model and context layers, while the action layer receives almost no attention, despite being the only layer where irreversible harm can actually occur. Second, it highlights auditability as a requirement frequently emphasized in compliance-focused research but not implemented by any of the surveyed methods, since probabilistic components cannot produce a decision trace that can be independently recomputed. Third, the review presents an empirical

finding, described in the following section, that helps distinguish architectural guarantees from heuristic ones using a common evidence base.

Method, Experiments and Results

To compare the surveyed families on an equal footing, we analyzed 1,846 programmatically scored agent episodes involving six base models from five organizations and six independent experimental runs. The episodes covered three benchmark suites addressing long-horizon reliability, clinical safety in irreversible settings, and indirect prompt injection. None of the reported results involved human grading. We compared two conditions: agents with no restriction at the tool interface, including context-management configurations, and agents for which every proposed irreversible tool call could proceed only when a set of deterministic policy predicates, evaluated against the environment state rather than generated text, was satisfied.

Deterministic action gating lowered executed unsafe clinical actions from 0.1135 to 0.0373 per episode, corresponding to a 67.1% relative reduction (bootstrap 95% confidence interval for the difference [-0.1263, -0.0261]; p = 0.0295). Executed prompt-injection attacks decreased from 6.42% to 0.00%, with no events observed across 134 gated episodes (p = 0.0027). Task success showed no statistically significant change on either suite (p = 0.696 and p = 0.685). Token cost increased by 8.2%, while the gating mechanism introduced no additional model inference. Table 2 provides the breakdown by model.

Table 2. Executed unsafe clinical actions per episode and attack execution rate, by base model

Base model	Unsafe acts, ungated	Unsafe acts, gated	Attacks executed, ungated	Attacks executed, gated
DeepSeek-V3	0.149	0.076	6.2%	0.0%
Llama-3.3-70B	0.113	0.000	0.0%	0.0%
GPT-4o-mini	0.033	0.000	14.4%	0.0%
Pooled	0.1135	0.0373	6.42%	0.00%

The direction of effect was negative for every model family, reaching complete elimination on two of three. By contrast, a context-management intervention evaluated on the same corpus improved outcomes on one model and degraded them on another, reversing direction between families.

Discussions

The consistency of the result is more important than its magnitude. A defence whose effectiveness changes with the underlying model cannot readily be certified for clinical use: if an institution validates a system on one model and later switches providers, the original validation no longer establishes the same safety properties. Deterministic gating performs consistently across model families because its guarantee does not depend on the model, whereas an intervention mediated by the model necessarily inherits model-specific variability. This is the practical difference between an architectural safeguard and a heuristic one.

A second finding concerns the nature of the prevention itself. Gated agents did not generate fewer unsafe actions; rather, the unsafe actions they proposed were blocked from execution. We consider this distinction important. Attempting to make a stochastic generator reliably safe is an open-ended research problem without a clear completion criterion. Restricting the actions that the generator is permitted to execute, by contrast, is an engineering problem that can be defined through explicit specifications and

recorded in a log. Because the audit record produced by a deterministic gate is determined by the proposed action and the environment state, a reviewer can reproduce each decision exactly. A verifier-LLM cannot provide the same property: its refusal is a probabilistic output, while its explanation is a post-hoc narrative [8].

Two limitations temper these conclusions. First, containment applies only to violations represented in the predicate set; 0.0373 unsafe actions per episode remained, with these residual failures concentrated in one model family. Second, the evaluated environments are simulated, so the external validity of the findings for real electronic health records remains unestablished [2].

Conclusions

1. **Problem statement.** Clinical LLM agents may perform irreversible actions, while current runtime defences operate at layers that modify the likelihood of unsafe behaviour without guaranteeing that unsafe actions cannot be executed.
2. **Method employed.** A structured review of four defence families was conducted according to their layer of intervention and failure class, and the findings were compared with evidence from 1,846 programmatically scored agent episodes involving six base models.
3. **Main findings.** The action layer remains poorly covered relative to the consequences of failures occurring there, and none of the reviewed methods provides a reproducible audit trail. Deterministic gating of tool calls reduced executed unsafe clinical actions by 67.1% ($p = 0.0295$) and prevented all executed prompt-injection attacks ($p = 0.0027$), consistently across model families and without a significant loss of utility.
4. **Limitations and future work.** Containment is limited to violations represented by the specified predicate set, the evaluation environments are simulated, and no clinician adjudicated the blocked actions. Future work should derive predicate sets systematically from clinical guidelines, validate the approach in a standards-compliant sandbox, and assess its performance against adaptive adversaries.

References

[1] X. Hou, Y. Zhao, S. Wang and H. Wang, "Model Context Protocol (MCP): Landscape, security threats, and future research directions," ACM Trans. Softw. Eng. Methodol., 2026. *https://doi.org/10.1145/3796519*

[2] S. A. Alhazba, M. H. Ali, D. A. Alawi, M. A. Al-Betar and M. Abd Elaziz, "From reactive AI to agentic systems: A review of autonomous medical AI agents in healthcare," Arch. Comput. Methods Eng., 2026. *https://doi.org/10.1007/s11831-026-10655-y*

[3] S. Vatsal, H. Dubey and A. Singh, "Agentic AI in healthcare and medicine: A seven-dimensional taxonomy for empirical evaluation of LLM-based agents," IEEE Access, vol. 14, pp. 4840-4863, 2026. *https://doi.org/10.1109/ACCESS.2026.3651218*

[4] M. Omar et al., "Multi-model assurance analysis showing large language models are highly vulnerable to adversarial hallucination attacks during clinical decision support," Commun. Med., vol. 5, no. 1, 2025. *https://doi.org/10.1038/s43856-025-01021-3*

[5] A. Milani, V. Franzoni and E. Florindi, "Indirect prompt injection in large language models," Neural Comput. Appl., vol. 38, no. 13, 2026. *https://doi.org/10.1007/s00521-026-12266-x*

[6] C. A. Martin, J. M. Torres, R. M. Aguilar, S. Alayon and M. A. Bacallado, "Agentic knowledge curation versus full-context retrieval: An empirical study of retrieval failure topology in long-context LLM systems," Appl. Sci., vol. 16, no. 13, p. 6793, 2026. *https://doi.org/10.3390/app16136793*

[7] E. Y. Chang and L. Geng, "SagaLLM: Context management, validation, and transaction guarantees for multi-agent LLM planning," Proc. VLDB Endow., vol. 18, no. 12, pp. 4874-4886, 2025. *https://doi.org/10.14778/3750601.3750611*

[8] S. Raza, R. Sapkota, M. Karkee and C. Emmanouilidis, "TRiSM for agentic AI: A review of trust, risk, and security management in LLM-based agentic multi-agent systems," AI Open, vol. 7, pp. 71-95, 2026. *https://doi.org/10.1016/j.aiopen.2026.02.006*

[9] F. E. Yagoubi, G. Badu-Marfo and R. Al Mallah, "AgentLeak: A benchmark for internal-channel privacy leakage in multi-agent LLM systems," IEEE Access, vol. 14, pp. 94960-94978, 2026. *https://doi.org/10.1109/ACCESS.2026.3704541*

[10] Q. Li, Y. Li, X. Mao, T. Wang and T. Li, "A framework for runtime safety of industrial control systems through runtime verification," IEEE Internet Things J., vol. 12, no. 11, pp. 15587-15599, 2025. *https://doi.org/10.1109/JIOT.2025.3529887*