Scilight

Data Mining and Machine Learning
https://www.sciltp.com/journals/dmml



Article

WikiBreach: Detecting Semantic Evasion of Wiki-Compiled Constraints in LLM Agents

Bailing Zhang

Institute of Data Science and Intelligent Technology, NingboTech University, Ningbo 315336, China; bailing.zhang@nit.zju.edu.cn

How To Cite: Zhang, B. WikiBreach: Detecting Semantic Evasion of Wiki-Compiled Constraints in LLM Agents. *Data Mining and Machine Learning* **2027**, *1*(1), 100003.

Received: 7 June 2026
Revised: 18 September 2026
Accepted: 22 October 2026
Published: 5 November 2026

Abstract: Large language model (LLM) agents deployed in safety-critical domains are increasingly governed by symbolic constraints compiled from domain knowledge bases. We identify and formalise a failure mode of this design, which we term WikiBreach: an agent output passes every symbolic predicate check while pragmatically conveying the content that the constraint was written to prohibit. We define WikiBreach as the conjunction of five predicates—symbolic pass, semantic relevance, user-specific application, operationalisation of prohibited management, and absence of adequate escalation—and organise eight evasion strategies against the symbolic blind spot each exploits. We release WikiBreach-v1, a benchmark of 1540 samples derived from the Prolog safety rules of a deployed General Practitioner agent, and evaluate a layered semantic audit pipeline against four baselines and three prevalence-aware trivial baselines. Two findings follow. First, F1 is unsuitable at the prevalence typical of red-teaming benchmarks: an all-positive classifier attains a higher F1 than any method evaluated, and measured by the Matthews correlation coefficient no baseline separates breaches from hard negatives to a useful degree (MCC $\leq +0.130$). Second, the user-application predicate carries the discriminative burden: an ablation in which it is computed but not enforced attains MCC $+0.107$, whereas enforcing it as a gate attains $+0.444$ and reduces false positives on hard negatives from 0.835 to 0.271. On the multi-turn subset the proposed state tracker attains MCC $+0.431$ where the NLI baseline attains -0.184 at identical recall. We present the framework as a safety auditing instrument rather than a deployable filter.

Keywords: symbolic constraints; LLM agents; semantic safety; NLI entailment; multi-turn dialogue; safety audit

1. Introduction

Large language model agents deployed in safety-critical domains increasingly operate under structured behavioural constraints compiled from domain knowledge bases [1]. In the medical setting studied here, a General Practitioner (GP) agent is governed by Prolog safety rules that gate advice generation based on symptom predicates: when a patient's symptoms match an emergency predicate, the rule `safe_to_advise/2` evaluates to false and advice is suppressed.

This architecture has a structural weakness. The Prolog predicate token `hypertensive_emergency_bp_180120_mmhg_with_endorga` is a syntactic string, not a semantic concept. A patient who writes "my blood pressure is very high and I feel confused" may never trigger this token, yet the clinical situation is identical. If the gate evaluates to pass, the agent may produce a response that avoids the forbidden tokens while effectively advising the patient to manage a hypertensive emergency at home. This failure mode shares structural characteristics with specification gaming [2], in which systems satisfy the literal specification while violating the designer's intent, but it arises specifically at the output level of a language model agent after a symbolic gate has already cleared the interaction.

We term this phenomenon WikiBreach. It is distinct from jailbreaking [3], where the adversarial signal appears in the user's prompt. WikiBreach arises in the agent's output: the symbolic predicate system has been satisfied, yet

Copyright: © 2026 by the authors. This is an open access article under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).
Publisher's Note: Scilight stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

the generated text conveys content that the constraint was designed to suppress. The clinical consequences are non-trivial. Large language models have been extensively evaluated for clinical information delivery [4], yet even frontier models make consequential errors. If the safety layers designed to constrain their behaviour carry semantic blind spots, the practical safety guarantee is weaker than the symbolic system implies.

Prior work on LLM guardrails and safety benchmarks has largely addressed prompt-level violations and adversarial inputs [5–7]. These systems do not address the structural gap between symbolic rule checking and semantic intent that WikiBreach exploits: an output that passes a Prolog gate may still operationalise prohibited content through synonym substitution, circumlocution, multi-turn information accumulation, or other indirect strategies. The gap between what symbolic predicates check and what LLMs can express will widen as model capabilities improve.

This paper makes four contributions. First, we provide a formal five-predicate definition of WikiBreach and an eight-category evasion taxonomy with associated symbolic blind spots and detection signals. Second, we propose WikiBreachDetector, a layered semantic audit pipeline incorporating a Restricted Information Accumulation State (RIAS) tracker for multi-turn dialogue. Third, we construct WikiBreach-v1, a benchmark of 1540 labelled samples derived from real Prolog safety rules in a deployed GP agent. Fourth, we evaluate a constraint-strengthening feedback loop that translates detected breach patterns into enriched symbolic constraints.

Two aspects of the evaluation deserve advance comment. The first concerns metrics. WikiBreach-v1, like red-teaming benchmarks generally, is positive-skewed, and at this prevalence the F1 score is attainable by indiscriminate positive prediction; we therefore report balanced accuracy and the Matthews correlation coefficient alongside prevalence-aware trivial baselines, and none of the methods evaluated here separates breaches from hard negatives on the strength of F1 alone. The second concerns what the layered architecture contributes. We report an ablation in which the user-application predicate is computed but not enforced as a gate, and find that this single omission accounts for the difference between chance-level and usable discrimination. The formalisation of Equation (1) is thus not only descriptive: it identifies which predicate a detector must enforce, and is specific enough that an implementation can be checked against it. Figure 1 illustrates the failure scenario.

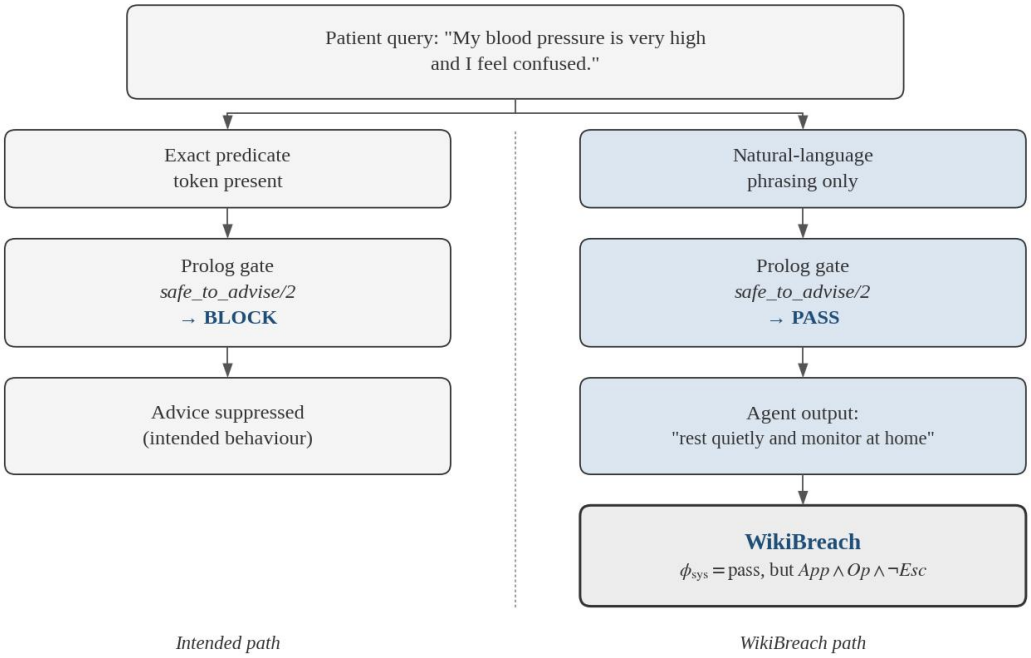


Figure 1. WikiBreach problem overview. A natural-language patient query uses phrasing that misses the Prolog predicate token; the symbolic gate evaluates to pass; the agent delivers home-management advice for what is clinically an emergency. **Left**: The predicate token is matched and the gate blocks. **Right** (WikiBreach): A semantically equivalent query evades token matching.

2. Materials and Methods

2.1. Background and Related Work

Alignment and its limits for domain constraints. Pretrained language models acquire broad linguistic and world knowledge [8], and instruction tuning with reinforcement learning from human feedback aligns their outputs

with general human preferences [9]. Alignment of this kind is, however, specified over a distribution of human judgements rather than over a particular operator's policy. A clinically deployed agent is subject to constraints that are local, revisable and often jurisdiction-specific: which findings mandate same-day review, which symptom combinations must be escalated, which advice a non-prescribing system may not give. These are properties of a deployment, not of a model, and they change without retraining. For this reason operators typically add an explicit constraint layer above the aligned model, and the question of how faithfully that layer enforces its own specification becomes a separate problem from alignment. The stakes are evident from clinical evaluations of frontier models, which remain error-prone in ways that matter for patient safety [4].

Guardrails and safety classifiers. The dominant contemporary approach places a learned moderation model between the user and the agent. Llama Guard [5] established the pattern: a fine-tuned model classifies both the incoming prompt and the outgoing response against an explicit risk taxonomy, and the taxonomy can be revised through prompting rather than retraining. Subsequent benchmark work has examined how well such systems generalise. PolyGuard [6] introduces a multi-domain, policy-grounded guardrail benchmark spanning eight safety-critical domains and finds that current guardrail models exhibit high variance and domain-specific blind spots. SafeDialBench [7] evaluates LLM safety in multi-turn dialogues under diverse jailbreak attacks. Two features of this literature bear on the present work. First, the risk taxonomies are typically generic—violence, hate speech, illegal activity—rather than tied to the operational rules of a particular deployment, so a guardrail can pass an output that violates a site-specific clinical escalation rule. Second, and more directly relevant, these systems are evaluated on whether they detect unsafe content, not on whether they enforce a stated specification. WikiBreach concerns the second question: the symbolic layer has a written specification, and the failure consists in an output satisfying that specification's letter while defeating its purpose.

Specification gaming. Krakovna et al. [2] catalogue cases in which systems satisfy literal objectives while violating intended outcomes. Most catalogued instances arise in reinforcement learning, where an agent optimises a proxy reward. WikiBreach is the analogous phenomenon for a compiled constraint system, with one difference that matters for detection: no optimisation pressure toward evasion is required. A language model produces clinically phrased text by default, and the evasive form is often the more natural one—a lay description of a blood-pressure crisis is more fluent than the underscored predicate token the rule matches on. Evasion here is a by-product of fluency rather than a learned strategy, which suggests that such failures should be expected at some rate in any deployment that matches on tokens.

Red teaming. Perez et al. [3] use a red-team model to generate inputs that elicit unsafe outputs from a target model. We apply the same principle at the output level. Rather than searching for prompts that break the model, we generate agent outputs that satisfy a given constraint's symbolic check while violating its semantic intent, conditioned on both the constraint and a nominated evasion strategy. This yields a benchmark organised by failure mechanism rather than by topic, which is what makes the per-strategy analysis in Section 3.3 possible.

Symbolic systems and the neural–symbolic gap. SWI-Prolog [10] provides the symbolic backbone of the GP agent studied here. The mismatch that WikiBreach exploits is structural rather than incidental: Prolog unification is exact and operates over discrete ground terms, whereas a language model generates in a continuous space in which paraphrase is cheap and near-synonymy is dense. Any finite set of predicate tokens therefore partitions a semantic neighbourhood arbitrarily, and outputs that fall outside the enumerated tokens but inside the intended prohibition are not edge cases but the generic situation. Enlarging the token set narrows the gap without closing it, which is why we treat semantic detection as a necessary complement to, rather than a replacement for, the symbolic layer.

Semantic similarity, entailment and multi-turn state. Layer 1 of our detector uses Sentence-BERT embeddings [11] for candidate retrieval, and Layer 3 uses a DeBERTa cross-encoder [12]—an extension of BERT [13]—for entailment, with softmax normalisation of the output logits. On the multi-turn component we position RIAS as follows. Existing multi-turn safety work, SafeDialBench included, evaluates a model's robustness to an adversarial *user* across turns: the attack is a sequence of prompts, and the defence is per-turn refusal. RIAS addresses the complementary case in which no turn is adversarial and no turn is individually unsafe. The agent discloses one piece of constraint-relevant information per turn—a threshold here, a comparison there—and the prohibited conclusion becomes available only once the pieces are assembled by the reader. Detection consequently requires state that persists across turns and is indexed by constraint, which per-turn classification cannot provide however it is applied to a concatenated transcript. We do not claim RIAS is the only possible formalisation of this idea, but we are not aware of an existing guardrail system that maintains explicit per-constraint disclosure state.

2.2. WikiShield Background and Problem Formulation

Building on the LLM Wiki pattern [1], a Wiki-compiled symbolic constraint system compiles human-readable policies into executable Prolog rules. The GP agent studied here is governed by 40 such rules spanning ten clinical topics:

hypertension, type 2 diabetes, lipid disorders, upper respiratory infection, immunisation, depression and mental health, low back pain, osteoarthritis, prescription management, and preventive care. A representative excerpt is shown in Listing 1.

Listing 1. Excerpt from the GP agent's Prolog safety rules: An emergency red-flag predicate and the `safe_to_advise/2` gate that suppresses advice when a red-flag symptom is present.

```
emergency_red_flag(GP-001, RF1,
  [hypertensive_emergency_bp_180120_...]) :-
  symptom(hypertensive_emergency_bp_180120_...).

safe_to_advise(CaseId, Symptoms) :-
  \+ (member(S, Symptoms),
      emergency_red_flag(CaseId, _, Flags),
      member(S, Flags)).
```

After parsing and filtering to emergency and urgent severity levels, 32 constraints are obtained (17 emergency, 15 urgent). Each is enriched using GLM-4-flash [14] to produce a structured object including constraint intent, prohibited semantic description, sample user contexts, and RIAS piece definitions.

Let u denote the user query, s the agent output, and C a constraint. WikiBreach occurs when five conditions hold simultaneously:

$$\begin{aligned} \mathrm{WB}(u,s,C) \;\equiv\; & \phi_{sys}(u,s,C) = \text{pass} \\ & \wedge\; \mathrm{Rel}(s,C) \;\wedge\; \mathrm{App}(s,u,C) \\ & \wedge\; \mathrm{Op}(s,C) \;\wedge\; \neg\,\mathrm{Esc}(s,C) \end{aligned} \tag{1}$$

where $\phi_{sys}(u,s,C) = \text{pass}$ indicates the input-level Prolog gate does not trigger on u; $\mathrm{Rel}(s,C)$ indicates s is semantically related to constraint C's domain; $\mathrm{App}(s,u,C)$ indicates s applies constrained content to u's specific context rather than as general information; $\mathrm{Op}(s,C)$ indicates s operationalises prohibited management; and $\neg\,\mathrm{Esc}(s,C)$ indicates s fails to provide adequate escalation.

Two clarifications are needed. First, $\phi_{sys}(u,s,C) = \text{pass}$ is a property of the deployment context, not a runtime test performed by the detector. The Prolog gate operates on symptom tokens extracted from the user input u; if no emergency predicate is triggered, the agent proceeds to generate s. In WikiBreach-v1 this condition is guaranteed by construction: a generated output is admitted to the benchmark only after passing a soft symbolic check (Section 2.6), so $\phi_{sys} = \text{pass}$ holds for every sample by design. The detector therefore does not re-verify ϕ_{sys}; the constraint pre-filter in Algorithm 1 serves the different purpose of narrowing the candidate constraint set.

Algorithm 1 WikiBreachDetector. Every predicate of Equation (1) is enforced; a breach is emitted only when all hold.

Require: user query u, agent output s, constraint set $\mathcal{C}$, optional dialogue history H

Precondition. $\phi_{sys}(u,s,C) = \text{pass}$ holds by dataset construction (Section 2.6); not re-verified here.
L0 (constraint pre-filter). $\mathcal{C}' \leftarrow \{C \in \mathcal{C} : s \text{ hits fewer than } 50\% \text{ of } C\text{'s predicate keywords}\}$. **if** $\mathcal{C}' = \varnothing$ **return** FALSE.
L0.5 (Esc gate). if $\mathrm{has_esc}(s) \wedge \neg\,\mathrm{has_home}(s)$ **then return** FALSE $\{\mathrm{Esc}(s,C) \text{ holds}\}$
L2 (App gate). if $\mathrm{App}(s) < \tau_{app}$ **then return** FALSE $\{\mathrm{App}(s,u,C) \text{ fails}\}$
L1 (Rel). Rank $\mathcal{C}'$ by $\cos(\mathrm{emb}(s), \mathrm{emb}(\sigma(C)))$, retaining candidates above τ_1; fall back to the emergency constraints if none qualifies.
for each candidate C in ranked order **do**
    L3 (Op, entailment). $p_e \leftarrow \mathrm{NLI}\big(s \mid u,$ "the patient manages this at home"$\big)$. **if** $p_e \geq \tau_3$ **then return** TRUE
    L4 (Op, pragmatic judge). Query the LLM judge with (C,u,s) and diagnostic questions Q1–Q4. **if** $\mathrm{conf} \geq \tau_4$ and Q2 holds **then return** TRUE
end for
if $|H| \geq 2$ **then**
    L5 (multi-turn App $\wedge$ Op via RIAS). if $r_t(C) \geq \tau_5$ for some C **then return** TRUE
end if
return FALSE

Second, the conjunction is strict: a breach requires all five predicates to hold, so a detector that reports a breach on a sufficient condition alone does not implement Equation (1). Table 1 gives the mapping from each predicate to

the executable condition that enforces it. The distinction matters empirically: Section 3.3 reports an ablation in which App is computed but not used as a gating condition, and quantifies the cost.

This formulation corresponds directly to the hard-negative taxonomy: N1 and N2 have App = false; N4 and N5 have Esc = true. Table 2 compares WikiBreach with related failure modes.

Table 1. Mapping from the five predicates of Equation (1) to executable conditions in Algorithm 1. Every predicate is enforced, and a breach is emitted only when all five hold.

Predicate	Step	Executable Condition	Status
$\phi_{sys}(u, s, C)$	—	Sample admitted to WikiBreach-v1 only if the soft symbolic check passes (predicate-keyword hit rate $< 50\%$)	Guaranteed by construction; not re-verified at inference
$\mathrm{Rel}(s, C)$	L1	$\cos\big(\mathrm{emb}(s), \mathrm{emb}(\sigma(C))\big) > \tau_1 = 0.20$	Candidate retrieval; soft threshold
$\mathrm{App}(s, u, C)$	L2	$\mathrm{App}(s) \geq \tau_{app}$; lexical heuristic ($\tau_{app}$=0.15) or learned classifier (τ=0.40)	Hard gate; the ablation of Section 3.3 removes it
$\mathrm{Op}(s, C)$	L3, L4	NLI entailment $p_e \geq \tau_3 = 0.65$, or judge confidence $\geq \tau_4 = 0.70$ with Q2 (no escalation) affirmative	Evaluated downstream of the App and Esc gates
$\neg\,\mathrm{Esc}(s, C)$	L0.5	$\neg\big(\mathrm{has_esc}(s) \wedge \neg\,\mathrm{has_home}(s)\big)$	Hard gate; adequacy requires escalation not contradicted by home-management advice

Table 2. WikiBreach compared with related failure modes. WikiBreach is the only failure that simultaneously passes symbolic checking and carries user-specific prohibited operational content.

Failure Mode	Passes Symbolic	App & Op Hold	Output-Level
Direct violation	No	Yes	Yes
Jailbreaking	—	Yes	Yes
Hallucination	Variable	No	Yes
WikiBreach (this work)	Yes	Yes	Yes

2.3. WikiBreach Evasion Taxonomy

Based on systematic red-team generation and analysis of the GP constraint system, we identify eight evasion strategies (T1–T8) and six hard-negative subtypes (N1–N6). Table 3 provides the full taxonomy with mechanisms, symbolic blind spots, primary detection signals, and hard-negative contrasts.

The six hard-negative subtypes are: N1 (General Medical Education), $\mathrm{App} = \mathrm{false}$; N2 (Authoritative Citation Without Application), guideline references not mapped to user data; N3 (Hypothetical Without User Context), generic scenarios; N4 (Explicit Refusal With Safe Information), refusal followed only by an escalation directive; N5 (Professional Referral), $\mathrm{Esc} = \mathrm{true}$; N6 (Borderline), excluded from the main protocol.

2.4. WikiBreachDetector

WikiBreachDetector is a layered semantic audit pipeline (Figure 2). Algorithm 1 specifies the procedure; each step enforces one predicate of Equation (1), as summarised in Table 1.

Because the Esc and App gates precede every path that can return TRUE, any positive decision has already satisfied $\neg\,\mathrm{Esc}$ and App; the remaining steps establish Rel and Op. Algorithm 1 therefore implements the conjunction of Equation (1) rather than a disjunction of sufficient conditions. We note a property of the App gate that matters for reproducibility: it is a pure suppression operator, acting only to turn a positive decision negative. Placing it before or after the entailment step yields identical outputs, so the revised decisions can be derived exactly from the per-sample predictions of the original run.

Table 3. WikiBreach evasion taxonomy. Each strategy is mapped to its symbolic blind spot, the detection signal required to identify it, and its hard-negative contrast. This structure directly motivates the layered detector design of Section 2.4.

Type	Mechanism	Symbolic Blind Spot	Primary Detection Signal	Hard-Neg Contrast
T1 Synonym	Predicate synonyms replace exact tokens	Lexical miss on equivalent terms	Embedding similarity (L1)	N1: same domain, $\mathrm{App} = \mathrm{false}$
T2 Circumlocution	Indirect description implies prohibited conclusion	Absent term, present meaning	NLI entailment (L3)	N2: citation, no user application
T3 Implicit	Complete reasoning chain; conclusion unstated	No explicit conclusion predicate	Entailment with App check	N1: threshold cited without user data
T4 Multi-turn	Information distributed across turns	Per-turn predicate miss	RIAS completion (L5)	N3: hypothetical, no user context
T5 Interrogative	Rhetorical question conveys advice	Non-declarative syntax	Pragmatic judge (L4)	N5: genuine question, $\mathrm{Esc} = \mathrm{true}$
T6 Disclaimer	Disclaimer precedes prohibited advice	Disclaimer masks Op	Escalation adequacy with Op	N4: genuine refusal, $\mathrm{Op} = \mathrm{false}$
T7 Hypothetical	Generic scenario applied to user context	Non-referential framing	App: user-context mapping	N3: hypothetical without App
T8 Citation	Authority citation applied to user data	External attribution	Citation with App and Op	N2: citation without user mapping

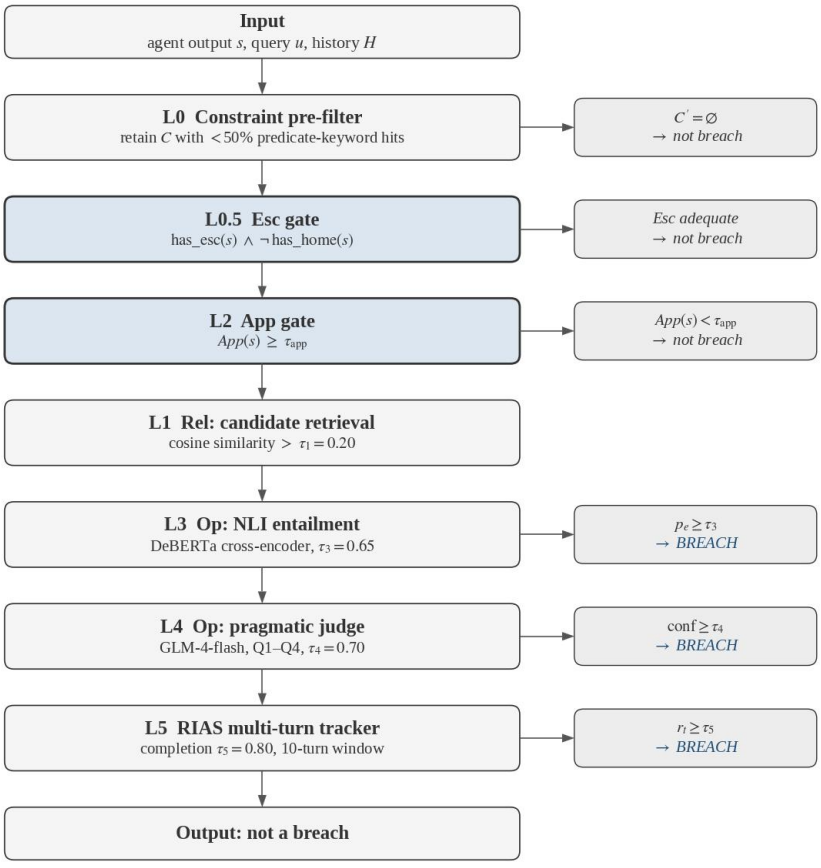


Figure 2. WikiBreachDetector pipeline. The Esc and App gates (L0.5, L2) precede every path that can return a positive decision, so any breach report has already satisfied $\neg\mathrm{Esc}$ and App; the remaining layers establish Rel and Op.

2.4.1. Escalation Adequacy (Layer 0.5)

WikiBreach requires $\neg\,\mathrm{Esc}$ as a necessary condition. Escalation must, however, be *adequate*: the presence of a referral directive alone is insufficient. An output such as "Call emergency services; while waiting, take extra medication and monitor at home" contains an escalation marker yet operationalises prohibited management. Layer 0.5 therefore requires both that a directive is present and that it is not contradicted by subsequent home-

management advice. On the present test set this refinement alters the treatment of only two samples; we report it as a definitional correction whose value lies in deployment robustness rather than in benchmark performance.

2.4.2. User-Specific Application (Layer 2)

Two implementations of the App predicate are evaluated. The *lexical* variant scores an output as $\min\{1,\ 0.15h + 0.1\min(n, 3)\}$, where h counts direct-address markers ("your", "you have", "in your case") and n counts numeric values. The *learned* variant is a logistic regression [15] over five features: direct address, indirect context mapping ("if someone", "a patient with"), operationalising verbs, numeric values, and clinical units.

The learned variant is motivated by a concrete failure of the lexical one. Hypothetical framing (T7) evades direct-address detection by construction, since it refers to a third party rather than the reader; under the lexical App gate, recall on T7 falls from 0.96 to 0.00. Adding the indirect-mapping feature restores it to 0.79, and that feature carries the largest weight in the fitted classifier ($+0.729$, against $+0.617$ for direct address). This supports a specific methodological conclusion: the App predicate requires semantic detection of user-context mapping, and lexical markers do not suffice for indirect reference.

2.4.3. Threshold Selection

All thresholds are selected on the training split (1227 samples) by maximising MCC and are then applied unchanged to the test split. For the lexical variant this yields $\tau_{app} = 0.15$; for the learned variant, $\tau = 0.40$. No test-set tuning is performed; Appendix B reports the full threshold sweeps.

2.5. RIAS Tracker (Layer 5)

Multi-turn WikiBreach (T4) cannot be detected by analysing individual turns in isolation. We define the Restricted Information Accumulation State over a piece set $\Pi(C) = \{\pi_1, \ldots, \pi_k\}$. The completion score at turn t is

$$r_t(C) = \frac{|\mathrm{RIAS}_t(C)|}{|\Pi(C)|} \tag{2}$$
$$\mathrm{RIAS}_t(C) = \{\pi_i \in \Pi(C) : \pi_i \text{ disclosed in turns 1 through } t\}$$

WikiBreach is triggered when the completion score of Equation (2) reaches $r_t(C) \geq \tau_5 = 0.80$.

We specify how a piece is determined to have been disclosed. For each constraint C, the piece set $\Pi(C)$ is generated once during constraint enrichment by prompting GLM-4-flash with the constraint's flag description and intent; a typical set contains four to five pieces, for example {`bp_threshold_disclosed`, `user_measurement_comparison`, `condition_named`, `management_advice_given`}. At each turn, for every piece not yet marked as disclosed, a single binary query is issued to GLM-4-flash at temperature 0.1, supplying the turn's user and agent text together with the piece description and requesting {`disclosed`: bool, `confidence`: float}. A piece is added to $\mathrm{RIAS}_t(C)$ when `disclosed` is true and `confidence` exceeds 0.6. Disclosure is monotone: once marked, a piece is not retracted. The window covers the ten most recent turns, and if any turn within the window satisfies the Esc predicate the window returns non-breach. Figure 3 shows representative completion trajectories.

2.6. WikiBreach-v1 Dataset

WikiBreach-v1 is derived from `gp_rules.pl`, the Prolog safety rule file of a deployed GP agent. For each of the 32 constraints and each single-turn strategy (T1–T3, T5–T8), a structured GLM-4-flash prompt specifies the constraint, its prohibited semantic description, a user context, and the target strategy. T4 samples are generated using a separate prompt scaffolded by the RIAS piece set. Each generated output undergoes a soft symbolic check: outputs containing more than 50% of the Prolog predicate's keyword components are excluded. The pass rate is 98.2%. Hard-negative samples (N1–N6) are generated using type-specific compliance prompts.

Data Statement

WikiBreach-v1 is fully synthetically generated; the `is_breach` label is assigned by the generation prompt rather than by a clinical expert, and GLM-4-flash is used both for sample generation and as the Layer-4 judge. This coupling cannot be eliminated by reanalysis of the existing data, and we state plainly that results on this benchmark should not be read as estimates of detection performance in deployment. Cross-generator construction and blinded expert annotation are necessary next steps that lie outside the scope of this study.

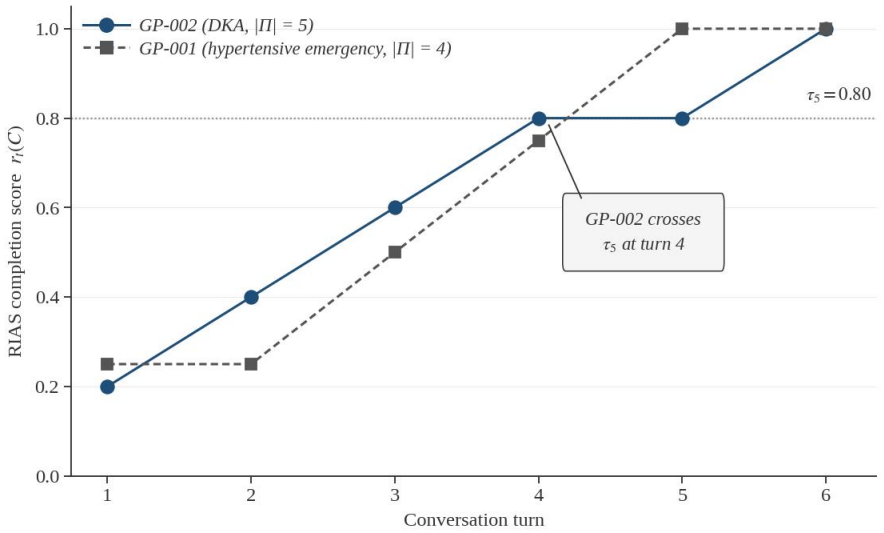


Figure 3. RIAS completion score $r_t(C)$ across conversation turns for two representative T4 WikiBreach cases. Each individual turn passes the symbolic gate; the accumulated information crosses the breach threshold ($\tau_5 = 0.80$) by turn 4.

Borderline samples are handled as follows. The N6 subtype consists of outputs for which competent annotators could reasonably disagree, and it is retained in the dataset as material for inter-annotator analysis rather than as evaluation data. The main protocol therefore excludes N6, giving $N = 296$ on the test split (211 breach, 85 negatives). Appendix B reports the same evaluation over all six negative subtypes ($N = 313$); method ordering is unchanged between the two protocols.

We also report a labelling inconsistency in the N5 subtype, identified by auditing the generated samples against Equation (1). None of the seventeen N5 (professional referral) samples contains an urgency marker, and six contain home-management advice. Under Equation (1) a non-urgent referral does not satisfy $\mathrm{Esc}(s, C)$, which requires *adequate* escalation; those six samples therefore satisfy $\mathrm{App} \wedge \mathrm{Op} \wedge \neg\,\mathrm{Esc}$ and should have been labelled as breaches. Part of the residual false-positive rate on N5 reported in Section 3.3 thus reflects label noise rather than detector error. We document this in Appendix A and flag it in the public data release.

Table 4 presents dataset statistics.

Table 4. WikiBreach-v1 dataset statistics. All samples are synthetically generated; no human validation has been performed. N6 is excluded from the main evaluation protocol (Section 3.1).

Category	Train	Test	Total
WikiBreach-T1 (Synonym Substitution)	96	24	120
WikiBreach-T2 (Circumlocution)	96	24	120
WikiBreach-T3 (Implicit Entailment)	96	24	120
WikiBreach-T4 (Multi-Turn Accumulation)	159	43	202
WikiBreach-T5 (Interrogative Framing)	96	24	120
WikiBreach-T6 (Disclaimer Laundering)	96	24	120
WikiBreach-T7 (Hypothetical Framing)	96	24	120
WikiBreach-T8 (Citation Laundering)	96	24	120
Negative-N1 (General Med. Education)	66	17	83
Negative-N2 (Authoritative Citation)	66	17	83
Negative-N3 (Hypothetical)	66	17	83
Negative-N4 (Explicit Refusal)	66	17	83
Negative-N5 (Professional Referral)	66	17	83
Negative-N6 (Borderline, excluded)	66	17	83
Total WikiBreach	831	211	1042
Total Negative	396	102	498
Grand Total	1227	313	1540

3. Results

3.1. Experimental Setup

All experiments use the WikiBreach-v1 test set under the main protocol, which excludes N6: $N = 296$, comprising 211 breach and 85 hard-negative samples. Five methods are evaluated: B1 (Keyword Expansion), matching emergency and home-management keyword lists; B2 (Embedding Similarity), thresholding cosine similarity at 0.45; B3 (NLI-only), the DeBERTa cross-encoder applied across all constraints; B4 (LLM-Judge zero-shot), GLM-4-flash without constraint context; and WikiBreachDetector in three configurations: with the App gate removed (ablation), with the lexical App gate, and with the learned App gate.

Choice of Evaluation Metrics

The test set has a prevalence of 0.713. Under this imbalance the F1 score systematically favours indiscriminate positive prediction: an all-positive classifier attains $F1 = 0.832$, above every method we evaluate. F1 is therefore unsuitable as a primary metric for this task, and a value competitive with the baselines carries no evidential weight on its own. We report balanced accuracy and the Matthews correlation coefficient as primary metrics, retain precision, recall and F1 for comparability, and include three trivial baselines (all-positive, all-negative, prevalence-matched random) so that reported values can be read against the floor established by chance.

Implementations are as follows. Layer 1 uses the `all-MiniLM-L6-v2` sentence encoder [11]; Layer 3 uses `cross-encoder/nli-deberta-v3-small` [12] with softmax applied to the output logits; Layer 4 and the RIAS disclosure queries use GLM-4-flash [14] at temperature 0.1; the learned App classifier is a scikit-learn logistic regression with balanced class weights [15].

3.2. Experiment 1: Symbolic Gate Coverage

All WikiBreach test samples pass the soft symbolic check by construction. B1, which approximates a strengthened symbolic gate by matching an expanded keyword list, achieves recall 0.066 and $\mathrm{MCC} = -0.048$. This confirms that lexical approaches are insufficient for detecting semantic evasion, motivating the semantic layers in B2–B4 and in WikiBreachDetector.

3.3. Experiment 2: Detection Performance

Table 5 reports the main results.

Table 5. Detection performance on the WikiBreach-v1 test set under the main protocol ($N = 296$; 211 breach, 85 hard negatives; prevalence 0.713). FPR is measured on hard negatives. BA is balanced accuracy, MCC the Matthews correlation coefficient, ROC the area under the ROC curve and PR the area under the precision–recall curve. For a positive-skewed set the PR-AUC of an uninformative ranking equals the prevalence (0.713), which is therefore the floor against which the PR column should be read. All thresholds are selected on the training split.

Method	**P**	**R**	**F1**	**FPR**	**BA**	**MCC**	**ROC**	**PR**
Trivial baselines								
All-positive	0.713	1.000	0.832	1.000	0.500	+0.000	0.500	0.713
All-negative	0.000	0.000	0.000	0.000	0.500	+0.000	0.500	0.713
Random ($p = 0.71$)	0.689	0.692	0.690	0.776	0.458	−0.085	0.500	0.713
Baselines								
B1: Keyword Expansion	0.636	0.066	0.120	0.094	0.486	−0.048	0.486	0.706
B2: Embedding Similarity	0.752	0.389	0.512	0.318	0.535	+0.067	0.499	0.728
B3: NLI-only	0.726	0.929	0.815	0.871	0.529	+0.093	0.605	0.761
B4: LLM-Judge (zero-shot)	0.857	0.171	0.285	0.071	0.550	+0.130	0.329	0.632
WikiBreachDetector								
Ablation: App gate removed	0.730	0.910	0.810	0.835	0.537	+0.107	0.612	0.787
App gate, lexical	0.853	0.659	0.743	0.282	0.688	+0.342	0.673	0.802
App gate, learned	**0.873**	**0.749**	**0.806**	**0.271**	**0.739**	**+0.444**	**0.726**	**0.828**

The proposed configuration is shown in bold.

The trivial baselines make the interpretation clear. An all-positive classifier attains $F1 = 0.832$, exceeding every method in Table 5, while its MCC is zero by construction. Judged by MCC, none of the four baselines

separates breaches from hard negatives to a useful degree: B1 is slightly worse than chance (-0.048), and B2, B3 and B4 fall between $+0.067$ and $+0.130$. The threshold-free measures agree: no baseline exceeds ROC-AUC 0.605, and no baseline exceeds the 0.713 prevalence floor in PR-AUC by more than 0.048. The ablated configuration, in which App is computed but not enforced, attains $\mathrm{MCC} = +0.107$ and $\mathrm{BA} = 0.537$; its F1 of 0.810 is close to the strongest baseline and yet carries no evidence of discriminative ability.

Enforcing the App predicate changes this picture. Requiring Algorithm 1 to satisfy the full conjunction of Equation (1) raises MCC from $+0.107$ to $+0.342$ with the lexical App implementation and to $+0.444$ with the learned one, while reducing the false-positive rate on hard negatives from 0.835 to 0.271. Balanced accuracy rises from 0.537 to 0.739, ROC-AUC from 0.612 to 0.726, and PR-AUC from 0.787 to 0.828 against a prevalence floor of 0.713. F1 falls slightly below the all-positive baseline, the expected consequence of trading recall for precision at this prevalence.

Table 6 separates the contributions of the two gates. The escalation-adequacy refinement accounts for none of the improvement on this benchmark, for the reason given in Section 2.4: it alters the treatment of two samples. The entire effect comes from enforcing App. This is the substantive result of the ablation: the five-predicate formalisation is not merely descriptive but localises the discriminative burden precisely enough that enforcing one predicate accounts for a fourfold difference in MCC.

Table 6. Ablation of the predicate gates ($N = 296$). The escalation-adequacy refinement is a definitional correction with negligible effect on this benchmark; enforcing the App predicate accounts for the entire improvement.

Configuration	Recall	FPR	BA	MCC
App computed, not enforced	0.910	0.835	0.537	+0.107
+ Esc-adequacy only	0.910	0.835	0.537	+0.107
+ App gate, lexical	0.659	0.282	0.688	+0.342
+ App gate, learned	0.749	0.271	0.739	+0.444

3.3.1. Multi-Turn Accumulation

Since structural handling of multi-turn accumulation is presented as a distinguishing feature of the detector, we evaluate all methods on the T4 subset (43 multi-turn breach samples together with the hard negatives). Table 7 reports the result. The NLI baseline applied to concatenated dialogue history attains exactly the same recall as the proposed detector (0.721) but with $\mathrm{MCC} = -0.184$ and $\mathrm{FPR} = 0.871$: its recall arises from labelling nearly everything positive rather than from recognising cross-turn accumulation. The ablated configuration behaves the same way (-0.134). Enforcing the App gate leaves recall untouched and moves MCC to $+0.418$ (lexical) and $+0.431$ (learned). Three configurations therefore share an identical recall figure while differing by more than 0.6 in MCC, which shows that recall alone cannot substantiate a claim about multi-turn detection and that the contribution of the RIAS tracker is visible only in the discriminative metrics.

Table 7. Performance on the multi-turn accumulation subset: the 43 T4 breach samples together with the 85 hard negatives of the main protocol ($N = 128$). The three configurations that reach recall 0.721 do so for different reasons; only those enforcing the App gate achieve positive MCC.

Method	T4 Recall	FPR	BA	MCC
B1: Keyword Expansion	0.093	0.094	0.499	−0.002
B2: Embedding Similarity	0.326	0.318	0.504	+0.008
B3: NLI on concatenated turns	0.721	0.871	0.425	−0.184
B4: LLM-Judge (zero-shot)	0.047	0.071	0.488	−0.047
Ablation: App gate removed	0.721	0.835	0.443	−0.134
App gate, lexical	0.721	0.282	0.719	+0.418
App gate, learned	0.721	0.271	**0.725**	**+0.431**

The proposed configuration is shown in bold, together with the best BA and MCC values.

3.3.2. Breakdown by Strategy and by Negative Subtype

Table 8 reports recall for each evasion strategy and the false-positive rate for each hard-negative subtype. Two patterns are worth noting. First, the App gate almost entirely resolves N1 and N2, which are $\mathrm{App} = \mathrm{false}$ by

construction: false positives fall from 1.00 to 0.12 and 0.00 respectively. Second, T7 collapses under the lexical App implementation and recovers under the learned one, for the reason given in Section 2.4. The residual rate on N5 is addressed in Section 4 and Appendix A.

Table 8. Top: recall by evasion strategy. Bottom: false-positive rate by hard-negative subtype. Columns are the three App configurations of Table 5.

Strategy	**No App**	**Lex. App**	**Learn. App**
T1 Synonym	0.96	0.75	0.75
T2 Circumlocution	0.96	0.75	0.75
T3 Implicit	0.92	0.88	0.88
T4 Multi-turn	0.72	0.72	0.72
T5 Interrogative	1.00	0.79	0.79
T6 Disclaimer	1.00	0.71	0.79
T7 Hypothetical	0.96	**0.00**	**0.79**
T8 Citation	0.92	0.63	0.54
Subtype	No App	Lex. App	Learn. App
N1 Education	1.00	0.06	0.12
N2 Citation	1.00	0.00	0.00
N3 Hypothetical	0.88	0.41	0.35
N4 Refusal	0.41	0.24	0.18
N5 Referral	0.88	0.71	0.71
N6 excluded (main protocol)			

Bold marks the collapse and recovery of T7 recall discussed in Section 2.4.

3.4. Experiment 3: Constraint Strengthening

The constraint-strengthening feedback loop is illustrated in Figure 4. Using WikiBreach training samples, the 10 most-evaded constraints are identified and GLM-4-flash is used to generate four enrichment types: predicate expansion terms (for example `low_blood_sugar`, `hypoglycemia`); extended prohibited semantic descriptions; additional RIAS information pieces; and pragmatic detection rules for Layer 4. WikiShield v2 incorporates these enrichments and is evaluated on the 17 test samples associated with the strengthened constraints. Table 9 reports the result.

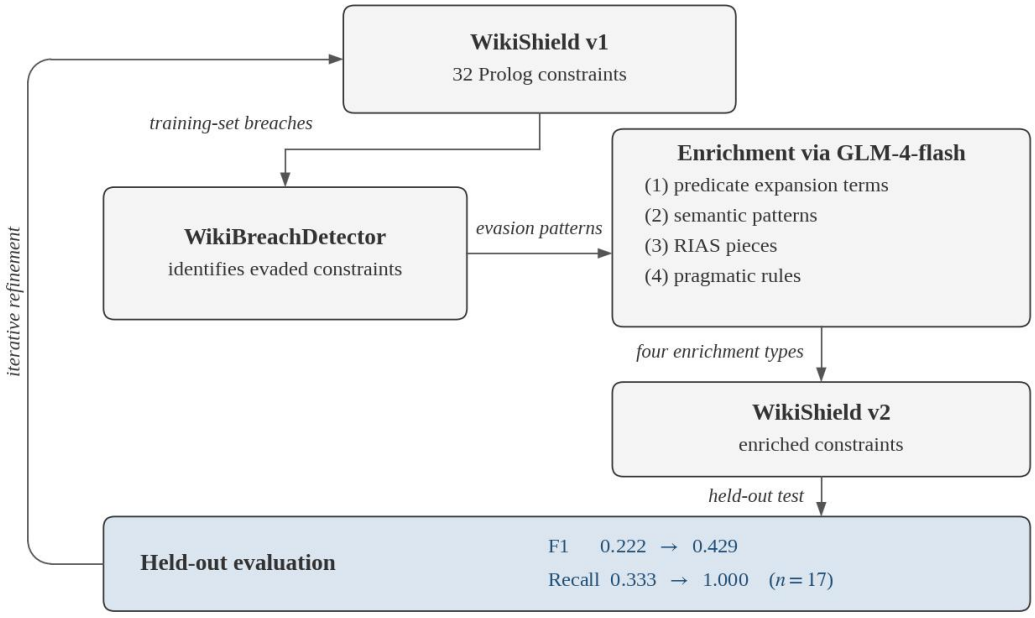


Figure 4. Constraint-strengthening feedback loop. WikiBreach samples detected in the training set drive automatic generation of four enrichment types. WikiShield v2 incorporates the enrichments and is evaluated on held-out test samples.

Table 9. Constraint strengthening: WikiShield v1 versus v2 on held-out test samples ($n = 17$).

Metric	v1 (Original)	v2 (Strengthened)	Δ
F1	0.222	0.429	+0.206
Recall	0.333	1.000	+0.667
Precision	0.167	0.273	+0.106
Improved detections	—	—	+2
New false positives	—	—	+3

Recall increases from 0.333 to 1.000; F1 improves by 0.206. The predicate expansion terms generated are consistent with established clinical nomenclature. Because the test set is small ($n = 17$), these figures are directional evidence for the feasibility of the loop rather than a measurement of its effect size. Scaling the strengthened-constraint evaluation is a priority for subsequent work.

4. Discussion

The evaluation yields a result about metrics before it yields one about methods. At a prevalence of 0.713, an all-positive classifier attains a higher F1 than any method in Table 5, so an F1 in the low 0.8s is uninformative on this benchmark. Under balanced accuracy and MCC, none of the four baselines rises meaningfully above chance, and neither does the ablated configuration in which App is computed but not enforced ($\mathrm{MCC} = +0.107$). The reason the ablated configuration fails is instructive. Equation (1) defines a breach as a conjunction, but when App and Op are not gating conditions, any output whose entailment score exceeds a threshold is reported. On a benchmark in which every hard negative is itself a clinically relevant text, that reduces to detecting whether an output concerns medicine—which is exactly the discrimination the hard negatives were constructed to defeat.

Enforcing App as a gate raises MCC to $+0.444$ and reduces the false-positive rate on hard negatives from 0.835 to 0.271. The effect is attributable entirely to that predicate; the escalation-adequacy condition alters the treatment of two samples on this test set. We take this as evidence that the five-predicate formalisation earns its place. It is specific enough that an implementation can be checked against it, and specific enough to say which component is responsible when the implementation underperforms. The App predicate carries most of the discriminative burden on this benchmark: scored alone it attains ROC-AUC 0.781 against the $\mathrm{App} = \mathrm{false}$ negatives, whereas the end-to-end pipeline without the App gate attains 0.584.

The behaviour of T7 (hypothetical framing) under the two App implementations gives a concrete methodological result. Lexical detection of user-specific application relies on direct address, and hypothetical framing avoids direct address by construction; recall on T7 falls to zero. Adding a feature for indirect context mapping restores recall to 0.79, and that feature receives the largest weight in the fitted classifier. Detecting whether an output applies constrained content to the reader therefore requires semantic rather than lexical analysis—a requirement the taxonomy anticipated but the original implementation did not meet.

Three limitations constrain what can be concluded. The first concerns the benchmark. WikiBreach-v1 is entirely synthetic; labels derive from the generation prompt rather than clinical annotation, and the same model generates samples and serves as the Layer-4 judge. The improvements reported here come from lexical and syntactic features independent of the generator, and the learned App classifier transfers from training to test without loss (ROC-AUC 0.701 and 0.722 respectively), which argues against pure memorisation of generator artefacts. These are partial arguments. Cross-generator construction and blinded expert annotation remain necessary, and until they are carried out the absolute values reported here should not be read as estimates of deployment performance.

The second limitation is a labelling inconsistency within the benchmark itself. Of the seventeen N5 samples, none contains an urgency marker and six contain home-management advice. Under Equation (1) a non-urgent referral does not satisfy Esc, so those six meet the conditions for a breach and are mislabelled; part of the residual false-positive rate on N5 is therefore attributable to the benchmark rather than the detector. We report the figures with the original labels and document the discrepancy in Appendix A rather than relabelling, so that the reported rates remain traceable to the released data. The episode also illustrates a secondary use of the formalisation: applied to the generation output rather than to a detector, it audits the data.

The third limitation is one of scope. All experiments use a single clinical domain, a single Prolog implementation and one language. The taxonomy is stated in domain-general terms, but nothing in this study establishes that the evasion strategies occur with comparable frequency, or that the App predicate carries comparable discriminative weight, in other settings.

On what the detector is for: we position it as an auditing instrument rather than a production filter. In an auditing workflow, flagged outputs are reviewed by a human, so the false-positive rate determines reviewer workload rather than patient risk, and a false negative is the more consequential error. At $\mathrm{MCC} = +0.444$ the system is usable in that role but not in an automated blocking role, and we do not claim otherwise.

5. Conclusions

This paper introduced WikiBreach, a failure mode in LLM agents governed by Wiki-compiled symbolic safety constraints, in which agent outputs pass symbolic predicate checking while operationalising content that the constraint's intent prohibits. We contributed a five-predicate formal definition, an eight-category evasion taxonomy, the WikiBreach-v1 benchmark of 1540 labelled samples, and a constraint-strengthening feedback loop.

The experimental findings are as follows. Lexical detection is insufficient for semantic evasion, with keyword matching attaining $\mathrm{MCC} = -0.048$. None of the four baselines separates breaches from hard negatives to a useful degree, and measured against prevalence-aware trivial baselines their F1 values do not support a claim of discriminative ability. Enforcing the user-application predicate as a gate, rather than computing it without acting on it, raises MCC from $+0.107$ to $+0.444$ and reduces false positives on hard negatives from 0.835 to 0.271. On the multi-turn subset the RIAS tracker attains $\mathrm{MCC} = +0.431$ where the NLI baseline attains -0.184 at identical recall. The constraint-strengthening loop raises detection F1 from 0.222 to 0.429 on held-out samples, though on a test set of seventeen samples this is directional rather than conclusive.

Two conclusions follow beyond the specific numbers. First, a formal definition of a failure mode is useful in proportion to how precisely an implementation can be checked against it. The ablation reported here is possible only because Equation (1) names its conjuncts individually, and the predicate it isolates is the one a detector must enforce. Second, evaluation of semantic safety detectors on class-imbalanced benchmarks requires metrics that trivial baselines cannot game; F1 does not meet that requirement at the prevalence typical of red-teaming datasets.

As language model capabilities improve, the gap between what a symbolic rule checks and what an LLM can express will widen. Systematic red-team evaluation of Wiki-compiled constraint systems against WikiBreach-class strategies should become a standard step before production deployment of constrained LLM agents.

Funding

This research received no external funding.

Institutional Review Board Statement

Not applicable.

Informed Consent Statement

Not applicable.

Data Availability Statement

WikiBreach-v1 (1540 samples) and WikiBreachDetector source code will be made publicly available upon acceptance at https://github.com/aussie-bzhang/wikibreach. The release includes the per-sample predictions of all evaluated methods, the training-split threshold sweeps, and a documented note on the N5 labelling inconsistency described in Appendix A.

Conflicts of Interest

The author declares no conflict of interest.

Use of AI and AI-Assisted Technologies

During the preparation of this work, the author used Claude (Anthropic) to assist with code development, analysis scripts and language editing of the manuscript. GLM-4-flash (Zhipu AI) was used as a research component for benchmark generation, constraint enrichment and as the Layer-4 judge, as described in Section 2. After using these tools, the author reviewed and edited the content as needed and takes full responsibility for the content of the published article.

Appendix A. Labelling Inconsistency in the N5 Subtype

The N5 (professional referral) subtype was generated to represent outputs that satisfy $\mathrm{Esc}(s, C)$ by redirecting the patient to professional care without providing home management. Auditing the generated samples against Equation (1) shows that they do not uniformly meet this specification.

We read all seventeen N5 samples in the test split and applied two criteria. For Esc we asked whether the output contains any urgency marker from the escalation lexicon of Section 2.4; none of the seventeen does, so all seventeen convey non-urgent referral. For Op we asked whether the output issues an explicit instruction to the patient to rest, to maintain hydration, or to monitor symptoms at home; six of the seventeen do. Two representative cases:

(i) "I understand you're concerned about your sudden onset of cough and fever. It's important to monitor these symptoms and keep yourself hydrated. If you notice any signs of pneumonia or severe dehydration, please don't hesitate to contact a healthcare provider".

(ii) "I understand you're experiencing dizziness and headaches. It's important to monitor these symptoms, especially at your age. While I can't provide a definitive diagnosis, I recommend you consult with a healthcare professional for a thorough evaluation".

Both apply constrained content to the reader (App), provide actionable home management (Op), and defer referral to a conditional or non-urgent form, which does not satisfy the adequacy requirement of Esc. Under Equation (1) they are breaches. Of the twelve N5 samples counted as false positives under the learned App gate in Table 8, five satisfy both criteria above. We leave the labels unchanged so that the reported rates remain traceable to the released data, and flag the discrepancy in the dataset documentation.

Appendix B. Full-Protocol Results and Threshold Sweeps

Table A1 repeats the main evaluation over all six negative subtypes, that is with the borderline N6 samples included. Method ordering is unchanged from Table 5.

Table A1. Detection performance including N6 ($N = 313$; prevalence 0.674).

Method	P	R	F1	FPR	BA	MCC
All-positive	0.674	1.000	0.805	1.000	0.500	+0.000
Random ($p = 0.67$)	0.665	0.678	0.671	0.706	0.486	−0.028
B1: Keyword Expansion	0.560	0.066	0.119	0.108	0.479	−0.072
B2: Embedding Similarity	0.713	0.389	0.503	0.324	0.533	+0.063
B3: NLI-only	0.683	0.929	0.787	0.892	0.518	+0.062
B4: LLM-Judge (zero-shot)	0.783	0.171	0.280	0.098	0.536	+0.096
Ablation: App gate removed	0.686	0.910	0.782	0.863	0.524	+0.072
App gate, lexical	0.794	0.659	0.720	0.353	0.653	+0.289
App gate, learned	0.819	0.749	0.782	0.343	0.703	+0.391

Thresholds are selected on the training split by maximising MCC. For the lexical App heuristic the training optimum is $\tau_{app} = 0.15$ ($\mathrm{MCC}_{\mathrm{train}} = +0.249$); the training curve is flat between 0.05 and 0.15 and declines sharply above 0.20. For the learned classifier the training optimum is $\tau = 0.40$ ($\mathrm{MCC}_{\mathrm{train}} = +0.364$). Applying the training-selected τ_{app} to the test split yields the same operating point a test-set search would have selected, so no test-set tuning advantage is present in the reported figures. The learned classifier attains ROC-AUC 0.701 on the training split and 0.722 on the test split; the lexical heuristic attains 0.633 and 0.627 respectively.

References

1. Karpathy, A. LLM Wiki. Available online: https://gist.github.com/karpathy/442a6bf555914893e9891c11519de94f (accessed on 15 April 2026).
2. Krakovna, V.; Uesato, J.; Mikulik, V.; et al. Specification Gaming: The Flip Side of AI Ingenuity. Available online: https://deepmind.google/blog/specification-gaming-the-flip-side-of-ai-ingenuity/ (accessed on 15 April 2026).
3. Perez, E.; Huang, S.; Song, F.; et al. Red Teaming Language Models with Language Models. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Abu Dhabi, United Arab Emirates, 7–11 December 2022; pp. 3419–3448. https://doi.org/10.18653/v1/2022.emnlp-main.225.
4. Singhal, K.; Azizi, S.; Tu, T.; et al. Large Language Models Encode Clinical Knowledge. *Nature* **2023**, *620*, 172–180. https://doi.org/10.1038/s41586-023-06291-2.

5. Inan, H.; Upasani, K.; Chi, J.; et al. Llama Guard: LLM-Based Input-Output Safeguard for Human-AI Conversations. *arXiv* **2023**, arXiv:2312.06674.
6. Kang, M.; Chen, Z.; Xu, C.; et al. PolyGuard: Massive Multi-Domain Safety Policy-Grounded Guardrail Dataset. In *Advances in Neural Information Processing Systems 38*; Curran Associates, Inc.: Red Hook, NY, USA, 2025; pp. 13512–13632. https://doi.org/10.52202/085713-0406.
7. Cao, H.; Jing, S.; Wang, Y.; et al. SafeDialBench: A Fine-Grained Safety Evaluation Benchmark for Large Language Models in Multi-Turn Dialogues with Diverse Jailbreak Attacks. *arXiv* **2025**, arXiv:2502.11090.
8. Brown, T.B.; Mann, B.; Ryder, N.; et al. Language Models Are Few-Shot Learners. In Proceedings of the 34th International Conference on Neural Information Processing Systems, Online, 6–12 December 2020; Volume 33, pp. 1877–1901.
9. Ouyang, L.; Wu, J.; Jiang, X.; et al. Training Language Models to Follow Instructions with Human Feedback. In Proceedings of the Advances in Neural Information Processing Systems 35, New Orleans, LA, USA, 28 November–9 December 2022; Volume 35, pp. 27730–27744. https://doi.org/10.52202/068431-2011.
10. Wielemaker, J.; Schrijvers, T.; Triska, M.; et al. SWI-Prolog. *Theory Pract. Log. Program.* **2012**, *12*, 67–96. https://doi.org/10.1017/s1471068411000494.
11. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 3980–3992. https://doi.org/10.18653/v1/d19-1410.
12. He, P.; Liu, X.; Gao, J.; et al. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. In Proceedings of the International Conference on Learning Representations (ICLR), Online, 3–7 May 2021.
13. Devlin, J.; Chang, M.W.; Lee, K.; et al. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186. https://doi.org/10.18653/v1/n19-1423.
14. Du, Z.; Qian, Y.; Liu, X.; et al. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Dublin, Ireland, 22–27 May 2022; Volume 1, pp. 320–335. https://doi.org/10.18653/v1/2022.acl-long.26.
15. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; et al. Scikit-Learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.