

RAG Faithfulness Evaluator

A Sentence-Level AI Control Monitor for Grounded Generation in High-Stakes Domains

Mouad Benmansour | Apart Research AI Control Hackathon, March 2026 | Track: Control Protocol Design

Keywords: RAG . AI Control . Faithfulness Evaluation . Hallucination Detection . Drift Analysis . LLM Monitor

Abstract

RAG is everywhere now, and organisations are adopting it precisely because it is supposed to be accurate. In the context of this hackathon I wanted to build something that holds RAG systems accountable to their primary role: faithfulness. And when they fall short, the system provides visualisations, sentence-level scores, and a detailed diagnostic analysis, itself powered by a RAG system, that shows what went wrong, where it went wrong, and what pattern the drift followed. It took a lot of trial and error and a willingness to keep questioning the approach to get to this prototype. It works, it catches real failures, and there is a clear path to making it significantly more powerful.

1. The Problem

The whole point of RAG is grounding. You retrieve documents, you pass them to a model, and the model answers using only what those documents say. In practice that breaks down quietly. The model paraphrases correctly in one sentence, then pulls from its training data in the next. It runs out of relevant content halfway through an answer and starts filling in gaps. It adds connective phrases for fluency that have no grounding anywhere. None of these are catastrophic failures. They are the ordinary drift that makes RAG unreliable in domains where every claim needs to be traceable.

Existing tools like RAGAS, TruLens and DeepEval measure faithfulness at the answer level. They give you one score and tell you the system failed. They do not tell you which sentence failed, why it failed, or what the pattern of failure looked like across the answer. That information gets lost in the average.

2. The Core Idea

Faithfulness is not a static property of an answer. It is a signal that changes sentence by sentence, and the shape of that change tells you something important. A score that starts high and drops progressively means the model ran out of retrieved context and started interpolating. A single dip surrounded by grounded sentences usually means a fluency bridge, a transitional phrase generated from language patterns rather than evidence. Uniformly low scores usually mean retrieval failed entirely.

These are different problems with different causes and different fixes. An aggregate score throws all of that away. Sentence-level scoring keeps it.

3. How It Works

For every sentence in a generated answer, the system computes two scores.

The first is semantic similarity: cosine similarity between the sentence embedding and the embeddings of all retrieved chunks, using the all-mpnet-base-v2 sentence transformer. The second is lexical overlap: Jaccard similarity between the meaningful keywords of the sentence and those of each chunk after removing stopwords. This catches cases where semantic similarity underestimates groundedness due to structural paraphrasing. The two scores are combined in an 85/15 weighted average favouring

semantic similarity. This split is an initial heuristic established through empirical testing on the ACL injury case study, where higher lexical weight consistently penalised correctly paraphrased sentences; a full grid search across domains remains an important next step.

Chunk size was tuned to 150 words with 30-word overlap after testing showed larger chunks diffuse meaning and reduce scoring accuracy. The grounding threshold was set at 0.35 after empirical calibration. Each sentence is also scored for relevance to the original query, distinguishing between a sentence that is unfaithful but still trying to answer the question and one that has drifted off-topic entirely. The scores are plotted sequentially as a drift timeline, making the pattern of failure immediately readable.

The Diagnostic Layer and the Watchmen Problem

On top of the statistical layer, the system passes the full context to an LLM with a structured prompt. The LLM reasons against a human-authored diagnostic guide of six known drift patterns: parametric override, context exhaustion, fusion hallucination, fluency bridging, confident refusal, and topical drift.

A natural concern here is the 'who watches the watchmen' problem: if the diagnostic layer is itself an LLM, who monitors the monitor? This is addressed through task decomposition and constitutional constraint. The diagnostic LLM is not performing open-ended generation. It is pattern-matching against a fixed, human-authored taxonomy it cannot modify or override. Its output space is bounded to six named diagnoses, making its behaviour auditable and its failure modes predictable. This is the same principle behind Constitutional AI: constrain the model's action space rather than relying on its judgment. Even if the diagnostic LLM drifts, it drifts within a human-specified, inspectable set of options. The diagnostic guide is itself a RAG knowledge base. The monitor that catches grounding failures is itself grounded. That is not an accident.

4. What It Found

Two results are worth highlighting.

In the first, a query about ACL injury and the menstrual cycle caused the model to produce a sentence drawing from training knowledge. This sentence scored 0.731 on semantic similarity but only 0.14 on lexical overlap, meaning its specific vocabulary was not present in any retrieved chunk. The statistical scorer marked it grounded. The LLM diagnostic layer correctly identified it as a parametric override and flagged it. Without the second layer, this false positive would have gone undetected.

In the second, a broad query caused the model to correctly refuse to answer rather than fabricate. The diagnostic layer identified this as a confident refusal, a correct control outcome, and recommended expanding the knowledge base rather than adjusting the model. The system does not penalise appropriate restraint.

5. Limitations

The threshold of 0.35 was tuned on one medical domain and should be validated against a labelled dataset using a precision-recall curve. HaluEval is the obvious benchmark for this. The lexical scorer does not handle synonyms or domain abbreviations well, and named entity recognition would fix most of this. The most important missing piece is entailment-based scoring, checking whether retrieved chunks logically entail each sentence rather than just measuring vocabulary proximity. That would catch

parametric override cases the current statistical layer misses.

6. Where This Goes

What excites me most about this is not the prototype itself but what it points toward. Every organisation deploying RAG in a high-stakes domain needs something like this. A real-time faithfulness monitor that does not just score but diagnoses, that tells an operator not just that something failed but what kind of failure it was and what to do about it, does not yet exist as a production-ready tool.

The path forward is concrete: validate the threshold against HaluEval, add entailment-based scoring as a third layer, build chunk attribution so every grounded sentence traces back to its exact source document, and extend the drift pattern guide through systematic study across multiple domains and model families. Each of these is a publishable contribution. The drift timeline alone, as a diagnostic framing rather than just a visualisation, is a idea I believe deserves a full research paper. I want to pursue this seriously, and the opportunity to do that with the right mentorship and infrastructure would be transformative.

7. Conclusion

RAG systems are being trusted with real decisions in high-stakes domains. The assumption that they stay within their retrieved context is not always justified, and the tools currently available to verify that assumption are not granular enough to be genuinely useful. This system is a step toward something better: a monitor that reads the shape of faithfulness drift, names what it sees, and tells you what to fix. The prototype works. The path forward is clear.

References

- [1] ReDeEP: Detecting Hallucination in RAG Systems via Mechanistic Interpretability. ICLR 2024.
openreview.net/forum?id=ztzZDzgfrh
- [2] Hallucination Mitigation for Retrieval-Augmented Large Language Models. MDPI Mathematics, 2025.
mdpi.com/2227-7390/13/5/856
- [3] RAG Hallucinations Explained: Causes, Types and How to Reduce Them. Mindee, 2025.
mindee.com/blog/rag-hallucinations-explained
- [4] Constrained Medical RAG and Hallucination Reduction. PubMed, 2024. pubmed.ncbi.nlm.nih.gov/40934488/

Code: github.com/mouadbenmansour/RAG-Faithfulness-Evaluator | Contact: mouadbenmansour.dev | Submitted: Apart
Research AI Control Hackathon, March 2026