

Application Tested

Tester

Public Road Safety Chatbot

How LLMs were used in application?

- Summarisation
- Retrieval augmented generation
- Classification and recommendation
- Multi-turn chatbot
- Data extraction from unstructured source

What Risks Were Considered Relevant And Tested?

-  **Primary business risk**
 - Reputational harm from an under-performing tool
-  **Associated technical risks**
 - Hallucination
 - Excessive guardrails/false refusal
 - Response Variance/inconsistency
 - Toxic and harmful outputs
 - Prompt injection and jailbreaks
 - Discrimination and bias

Multinational Insurer



The assured firm is a Multinational Insurer.
The use case is a publicly accessible chatbot, designed to promote better driving habits through expert guidance.



Verify AI is a firm that specialises in quantitative risk evaluations of AI model applications.



How Were The Risks Tested?


Approach

-  **Custom benchmarking for hallucinations**
 - Baseline accuracy testing: Questions autogenerated from chatbot’s designated knowledge base (using LLM)
 - Stress testing for consistency and non-refusal: Additional questions from other domain-relevant inputs (e.g., external docs, general knowledge from LLMs)
-  **Red Teaming**

Automated adversarial tests to probe for jailbreaks/harmful content/other worst-case failures

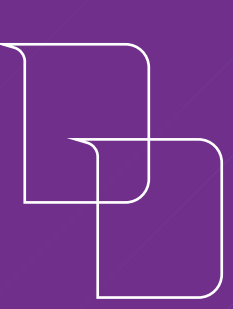
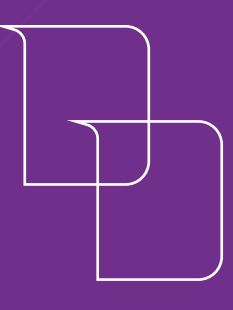
Evaluators

-  **Baseline accuracy testing**
 - Semantic match of answer with what exists in the knowledge base (using LLM as a judge)
 - Accuracy of citation provided (using ground truth)
 - Refusal rate, with manual investigation of validity of refusal
-  **Stress testing**

Above, plus disaggregation of failure modes along the lines of identified risks (e.g., fairness, security and compliance)
-  **Red Teaming**

Attack success rate

Challenges

-  Automatic generation of adversarial prompts: Substantial design effort and compute needed
-  Verifying insights from LLMs as a judge: Need for human oversight to ensure that these LLM judges perform well. Sequential model checking employed to build formal proofs of statistical correctness of evaluation

Insights

- 01
Clear, testable definitions needed for each technical risk (particularly in “no box” situations where only access to the final outputs of the chatbot are available)
- 02
Difficult to verify and understand the root-causes of hallucinations without visibility into the retrieval process
- 03
LLM as a judge viable, but unreliable without additional safeguards; statistical model checking can help

Public Road Safety Chatbot

Multinational Insurance Company

Use Case

The assured firm is an anonymous Insurance Company

A publicly accessible chatbot, that is available without login on the company website. The chatbot is designed to offer road safety advice. Its core value lies in promoting better driving habits through expert guidance. It is therefore critical that the chatbot consistently delivers accurate and responsible advice and does not recommend unsafe or dangerous practices. From the insurance company's perspective, the impact of the deployed chatbot on its reputation and the trust of its customers is key.

High-level Architecture

The system under consideration is a RAG-based LLM architecture that allows users to access safe driving advice. The document serving as the primary knowledge base for the RAG is publicly available and around 200 pages long. The exact version and training of the underlying foundation model is confidential and was unknown to the assurer, Verify-AI, during the audit. The chatbot supports inputs of no more than 300 characters and supports multi-turn conversation. However, only ten consecutive queries will be answered before the model refuses to answer further. Additional guardrails have been implemented by the deployer. The specification and nature of these additional guardrails are confidential and were not shared with the assurer before the audit.



The assurer is Verify-AI, a firm that specialises on quantitative risk evaluations of AI model applications.

Verify-AI uses a combination of open-source and proprietary software to perform automatic evaluations of black-box AI models at scale. To this end, Verify-AI uses its own application-specific risk and audit framework.

A primary challenge of this pilot is that the Verify-AI audit team must work in a **no-box setting** meaning that **only access to the final outputs of the chatbot are available**. This differs critically from the black box setting where it is commonly assumed that outputs and output probabilities are available. It differs even more considerably from a white box setting where auditors have access to the model architecture, weights, system prompts, safeguard details, etc.

Risk Assessment

Businesswise, the insurance company is concerned with reputational harm from an underperforming tool that provides incorrect driving recommendations. There is also a secondary operational risk associated with a potential increase in insurance claims, but this is considered negligible.

Based on the identified business risks, the following technical risks and failure modes have been considered in the pilot:

- **Hallucination**
The RAG generates facts not grounded in any evidence, retrieved or otherwise. For example, the chatbot’s outputs contradict official road rules.
- **Excessive Guardrails & False Refusal**
The chatbot refuses to respond to legitimate queries about road safety.
- **Response Variance & Inconsistency**
Repeated queries with identical input yield different or conflicting driving advice.
- **Toxic & Harmful Outputs**
The chatbot generates abusive or inappropriate content in response to edge-case or adversarial prompts related to driving behaviour or identity.
- **Discrimination & Bias**
The chatbot systematically favours or neglects particular geographic regions, demographics or user categories.
- **Prompt Injection & Jailbreaks**
Malicious prompts circumvent guardrails and manipulate the system into ignoring constraints.

Testing Scope

Each of these technical risks were evaluated under up to three of the following evaluation modes:

- **Benchmarking**
Tests the chatbot on inputs directly covered by the retrieval document. Questions were automatically generated, and answers evaluated for factual accuracy and correct citation. This establishes baseline performance under ideal conditions.
- **Stress-Testing**
Domain-relevant but out-of-distribution inputs to assess behaviour when retrieval is partial or absent. Focus was on identifying issues with hallucination, refusal, and inconsistency.
- **Red Teaming**
Automated and manual adversarial tests to probe for jailbreaks, harmful content, and other worst-case failures. Adversarial prompts were generated using surrogate models and transfer-tested on the deployed system.

Technical tests were designed to specifically address the identified risks, combining automated and manual methods:

Benchmarking

To ensure system operates as intended in a best-case scenario the team first broke the base document down into chunks of text of various sizes and with various offsets. The team then tasked an LLM with automated question generation based on the source text. Accuracy of the system returned answers is then determined by two criterias:

- Does the provided answer match semantically with what exists in the text?
- Does the LLM provide the correct citation?

The former is quantified with an LLM as a judge and the latter quantified using string-matching techniques such as regular expressions (vs. ground truth). In addition to these two metrics, the team also carefully tracked how often the LLM refuses to answer. This is automated by checking against a pre-filtered dataset where all questions are assumed valid. The team found this refusal rate concerning, as many instances counted as errors in the accuracy scores are in fact due to the model refusing to answer rather than providing an incorrect or hallucinated response.

Stress-Testing (In-Domain but Out-of-Distribution Tests)

To explore the more general use case of the tool, the team considered queries that are part of the safe driving domain, e.g., pertain to laws, regulations, and habits of safe driving but which may not be explicitly covered within the text itself. For this, a more comprehensive test generation is done including use of external documents, general knowledge from LLMs, manual construction, and automated exploration. At the general level, the team considered the same metrics as above but also focus on disaggregation of failure modes along the lines of the identified risks including fairness, security and compliance.

Red-Teaming (Adversarial and Out-of-distribution Tests)

To understand the worst-case failures of the model under consideration, Verify-AI undertook an extensive automated red-teaming test. This involved the training of several surrogate LLMs that guide the generation of adversarial prompts and the synthesis of novel attack strategies which are computed to have the highest chance of success when transferred to the in-production model.

Evaluators used a mix of techniques:

- **Rule-based checks**
for citation correctness and refusal detection
- **LLM as a judge**
for semantic answer quality
- **Human-in-the-loop**
calibration of LLM judges to ensure reliability

Verify-AI combined human evaluators to certify the accuracy and quality of LLM judges. Verify-AI used a statistically grounded framework that verifies the reliability of LLM judges relative to human annotators. Verify-AI also employed “entropy-prioritised” judge selection to identify and rely on the most consistent LLM judges. These techniques aim to ensure strong quality control over automatic assessments while keeping human annotation costs low.

Execution of Tests

01

Tests were executed using a bespoke platform. As model access was restricted custom interaction scripts were created to record answers and control model interactions. The results of tests were quantified using LLM as a judge in nearly all instances. Several hundred thousand queries were made. Advanced statistical model checking techniques were used to inspect and build provable confidence in the reported results. The testing was conducted in a secure staging environment with strict access controls.

Data Used in Testing

02

- 10K benign queries were used, 100k out of distribution queries were used, 50k adversarial queries were used.
- Queries origin was both extracted from known documents, generated by LLMs, synthesised using surrogate models of observed system performances, and optimised using adversarial approaches.

Cost of Testing

03

The testing process involved significant time allocation from the:

- Deployer’s technical team ~ 8 hours of meetings and discussions and
- Tester’s technical and expert teams ~ 8 hours for platform setup, 10 hours for test design, and 20 hours analysis

Additionally, there were direct costs associated with LLM usage, particularly token costs for generating test cases using a mixture of for-profit foundation models (totalling less than 1k GBP) along with the use of open-source models on existing computational hardware.

Challenges in Implementation

04

- Automatically synthesising adversarial prompts from prior model interactions requires substantial design effort and compute including training of surrogates and hyper-parameter optimisation of the adversarial attack process.
- Another key challenge was in the formal verification of insights from LLMs as a judge. Human oversight is required to ensure that judges perform well and that the risk signal from benchmarking processes is not lost due to judge error. To tackle this problem, Verify-AI deployed a novel approach leveraging sequential model checking to build formal proofs of the statistical correctness of their evaluation.

Insights on Risk Assessment

01

The no-box setting emphasised the need for clear, testable definitions of each technical risk. Without internal access, abstract concerns like hallucination or refusal had to be formalised into measurable behaviours. Hallucinations were frequent but difficult to verify due to a lack of visibility into the retrieval process. Overly aggressive refusals emerged as a distinct failure mode, often blocking reasonable safety queries and reducing system utility.

Lessons from Test Design

02

Most existing RAG evaluation tools assume access to internal retrieval scores or retrieved sections, which makes them unsuitable for no-box audits. Verify-AI developed our own benchmarking framework using external chunking and automated question generation. For red teaming, Verify-AI trained surrogate models to approximate system behaviour and guide adversarial prompt creation. This approach proved effective but required substantial infrastructure.

Lessons Learned from Test Implementation

03

Interpretability was a key challenge. Without insight into model internals, understanding why a failure occurred was slow and uncertain. LLM-based evaluation (LLM as a judge) was viable but unreliable without additional safeguards; Verify-AI used statistical model checking to ensure evaluation robustness. Overall, future audits would benefit from clearer expectations about audit load-sharing and minimal transparency standards to make output-only auditing practical.