AI Agent Evaluation: Building an Evaluation Platform That Scales

By: Jennifer Sikos PhD, Senior Engineering Manager

A single prompt change inside an agentic workflow can improve one agent's performance while silently degrading three others. In an agentic system, where agents select tools, plan multi-step actions, manage state, retrieve context, and produce structured outputs, the failure surface is distributed and the dependencies are often invisible until something breaks in a live environment.

A mature AI agent evaluation platform transforms agent development from an unstructured, ad hoc process into a controlled, auditable engineering discipline, one where regressions are caught early and performance is quantified.

Evaluation is the engineering discipline that makes agentic workflows verifiable. Robust evaluations bring in more authority and trust to an agentic workflow, and they make human governance over agentic behavior a reality.

Evaluation Requires Multiple Components

A common approach when evaluating an AI agent is to focus on a single output, but that alone is insufficient.

Evaluating only on the final output is similar to assessing the structure of a house without looking at its foundation, wiring, or insulation; even if it appears sturdy on the surface, there could be a number of underlying issues that need to be addressed to make it structurally sound. A complete evaluation framework covers multiple components to ensure structural soundness.

Datasets and metrics to define success: whether the data used to measure system outputs is an accurate reflection of the data used in production. Academic benchmarks are insufficient determinants of output quality in most operational workflows, because they are benchmarking fundamentally different problems and different datasets. In a robust evaluation framework, the datasets used to measure performance mirror the type of data the system sees in production.

Tool behavior: whether the correct tools are being invoked with well-formed arguments. For example, an agent asking for the weather might return the correct answer, but only after calling the wrong search tool and ignoring the structured weather API it was given. While the system

could get lucky and produce a correct answer once or twice, it is not a reliable method to count on over multiple runs.

Trajectory quality: whether the reasoning path is sound, not just whether the final output was correct. This is especially critical for workflows that require deterministic answers that depend on a clear chain of reasoning. It is like a student getting an arithmetic answer correct but the logic they applied to derive the answer was flawed. An agent can arrive at a right answer through an unsound process that fails under different conditions.

Safety boundary verification: whether forbidden actions are avoided. In environments where certain operations carry irreversible consequences, verifying the absence of unsafe behavior is as important as verifying correct behavior.

The remainder of this post focuses on the first requirement - the datasets and metrics that power robust evaluations. Treating datasets as immutable, versioned, and portable is the foundation for reliable and reproducible benchmarking in any serious agent evaluation system.

Evaluation Datasets Must Be Diverse, Immutable, and Versioned

The most common failure in evaluation infrastructure is mishandling the evaluation datasets.

Data diversity and versatility are crucial for creating a robust evaluation framework that accurately simulates production conditions. Good evaluation datasets include simple, prototypical examples as well as a full spectrum of edge cases that the agent is expected to encounter at runtime. This ensures that the agent's performance is tested not just on common success paths, but also under more ambiguous and challenging conditions. A versatile dataset includes different modalities, complexity levels, and domain-specific challenges, guaranteeing that evaluation results are a realistic predictor of operational behavior.

Too often, the datasets used for academic benchmarking are the sole determinant of what “good” looks like for agentic workflows. However, datasets used in those benchmarks rarely reflect the same conditions of operational data. Effective evaluation requires datasets that reflect real-world usage, ideally sampled directly from production environments.

For any evaluation to be effective, stability in the test dataset is essential. This dataset must serve as the foundation, accurately representing the correct outputs for the specific task being assessed. If datasets change between runs, results become incomparable. This is why datasets need to be treated as immutable during agentic evaluations. Imagine if data is added or changed in a “gold standard” dataset between test runs - it would then become impossible to assess whether improvements were caused by a change in the prompting or a change in the underlying data.

A rigorous evaluation platform treats datasets as stable, versioned, immutable artifacts. Each dataset is formatted in a consistent schema that accommodates different task types and modalities. At minimum, a dataset contains the questions or prompts to be evaluated and gold standard answers. It may also include the raw source documents required to answer those questions, and the expected tool execution sequences.

Evaluation quality is also shaped by how data is prepared before it reaches the model. The quality of pre-processed data like parsing, chunking, metadata extraction, and OCR significantly affects retrieval performance and output quality downstream. An evaluation framework should maintain records and account for the data pipeline feeding the model.

Metrics Must Reflect Real-World Objectives

Standard evaluation benchmarks were designed for research settings. Operational environments require a range of different metrics.

Metrics should reflect what the system is supposed to accomplish. This means defining custom metrics tied to operational outcomes: accuracy of the evidence provided, correctness of the final answer, task completion rate, latency under realistic conditions, and cost per workflow execution. In practice, the metrics that define "good" should be specific to the objective the agent is aiming to achieve. A summarization agent is evaluated on conciseness, coherence, and fidelity to source material. A question-answering agent over live operational data is evaluated on accuracy, source grounding, and policy compliance. Applying a single scoring rubric across both produces a number that is accurate for neither; evaluation design starts with being precise about what the system is actually supposed to accomplish.

LLM-as-a-judge is a technique where a language model is used to automatically score or critique the output of another model. It has become one of the most standard and commonly used evaluation methods, but can become problematic when implemented poorly. The technique enables evaluation of nuanced, open-ended tasks that are difficult to measure with deterministic methods. An LLM judge reads a model's response and applies a rubric, scoring correctness, helpfulness, and faithfulness, much as a human reviewer would.

The tradeoffs are significant and need to be managed explicitly. LLM judges can reward verbosity over substance, accept hallucinated content as valid, produce inconsistent results across runs, and shift in behavior as underlying models update. Managing these risks requires tight, explicit rubrics, calibration against human-labeled data, and deterministic settings: temperature at zero, prompts fixed, model version pinned. Whenever possible, a vendor-agnostic evaluation that allows for multiple LLMs to score outputs is ideal, which helps offset any bias in a single LLM-as-a-judge.

Even with these precautions, LLM judges should not be the sole gate for safety or correctness. An LLM-as-a-judge that drifts, hallucinates, or rewards verbosity over accuracy will fail silently, producing scores that look valid while masking real regressions. Deterministic, programmatic checks need to be layered in wherever the consequences of an incorrect judgment are irreversible. The combination of LLM-based judgment for qualitative assessment and deterministic checks for safety-critical boundaries is the only approach that produces reliable results at scale.

Evaluation Artifacts Must Be Decoupled From Infrastructure

Portability and vendor-agnostic design enables reliable benchmarking across disparate environments.

A critical feature of a mature AI agent evaluation platform is a vendor-agnostic design, ensuring evaluations are portable and not tied to any single model provider or execution environment. True portability means the evaluation framework can benchmark agentic systems across multiple different models, whether commercial LLMs, open-source models, or proprietary models.

Decoupling evaluation from a specific LLM or cloud platform prevents vendor lock-in, allowing organizations to switch providers, adopt new state-of-the-art models, or move to more cost-efficient alternatives without rebuilding their testing infrastructure. Using a consistent set of versioned datasets and standardized metrics also enables fair, side-by-side comparison of model performance on the same operational tasks. When evaluation is vendor-agnostic, models can be compared directly under the same conditions, enabling confident selection of the best-performing option.

Achieving vendor-agnosticism requires strict separation between the evaluation artifacts (datasets, metrics, rubrics) and the execution environment. By treating the agent's interaction as a generalized process, the platform can maintain objective standards while adapting to varied execution backends.

Testing over multiple different models is one area where flexibility in an evaluation platform is critical. Another is the portability of test data. Datasets should be grounded in raw source documents rather than references (such as indexes or caches) specific to a particular infrastructure. This means the same evaluation can be re-initialized in a clean environment, including a classified network, an air-gapped deployment, or a new cloud instance, without coupling to the infrastructure where the evaluation was first run.

Evaluation Maturity Is Visible in How Teams Behave, Not in What Platforms They Deploy

The goal is an engineering culture where performance claims are substantiated before decisions are made.

A mature, AI-driven engineering team treats AI agent evaluation as an engineering discipline requiring scientific rigor. This shift means moving away from ad hoc testing to establishing controlled, auditable processes where performance claims are substantiated with verifiable data from reproducible experiments. By treating evaluation artifacts, datasets, metrics, and rubrics, as immutable, versioned code, teams can catch regressions early, reliably quantify performance, and build the organizational trust necessary to scale agentic workflows with confidence and governance.

Legion Intelligence built its evaluation framework on these foundations: immutable versioned datasets, configurable multi-metric evaluation across all four lenses, full execution tracing, and complete environment isolation. In classified and air-gapped environments where production failures carry real consequences, the alternative to rigorous evaluation is not faster deployment. It is an uncontrolled system operating under assumed human authority rather than verified human authority.

Evaluation frameworks verify that governed agentic systems perform as designed. For the architectural foundations those systems are built on, read Agentic AI Security is an Architectural Decision.

Continue the series

Follow Legion Intelligence on LinkedIn: linkedin.com/company/legion-intel

Explore all Command Papers: legionintel.com/command-papers

New papers published every 1–2 weeks.