Scalable Oversight: Designing an Automated Evaluation Pipeline for Enterprise GenAI Reliability

MSc Software Engineering Thesis

Eneko Retolaza Ardanaz

Scalable Oversight: Designing an Automated Evaluation Pipeline for Enterprise GenAI Reliability

THESIS

submitted in partial fulfilment of the
requirements for the degree of

MASTER OF SCIENCE

in

SOFTWARE ENGINEERING

by

Eneko Retolaza Ardanaz

under the supervision of

Dr. Adam Belloum (University of Amsterdam, a.s.z.belloum@uva.nl)

July 2026

Master Software Engineering
Informatics Institute
FNWI, University of Amsterdam
Amsterdam, The Netherlands

KPN
Teleportboulevard 121
1043 EJ Amsterdam
The Netherlands

Thesis Committee

Examiner (chair):	Dr. Adam Belloum (University of Amsterdam, a.s.z.belloum@uva.nl)
Second Reviewer:	Dr. ir. Martin Bor (University of Amsterdam, m.c.bor@uva.nl)
Daily Supervisor:	Ashokkumar Shankar (KPN, ashokkumar.shankar@kpn.com)

Abstract

This thesis designs and evaluates a scalable evaluation and supervision pipeline for enterprise GenAI applications. The core challenge is that LLM-based systems are non-deterministic: small changes in prompts, models, retrieval, or orchestration can introduce subtle behavioural regressions that deterministic testing does not reliably detect. The thesis therefore addresses two linked questions: how to design a reusable evaluation architecture for enterprise GenAI supervision, and how to derive minimum viable tests for one concrete high-value use case.

The architectural contribution answers RQ1 through a reusable supervision workflow that separates source adapters, a canonical evaluation schema, versioned benchmark configuration, repeatable execution paths, and audit-ready result export. In the current implementation, this reusable core is demonstrated across both code-based and interactive workflow paths for the same enterprise summary-evaluation use case, showing how the same supervision pattern can support different application surfaces. RQ2 then uses one enterprise summary-evaluation case to derive a bounded screening process and to examine how platform-visible proxy metrics behave across benchmark settings. The results show that proxy relevance is strongly benchmark-dependent and that the final human-reviewed gate works mainly as a conservative review-routing mechanism rather than as a sharp automatic pass/fail separator. In addition to the calibrated gate, the work produces a reusable minimum-benchmark process note, a reusable template, and a privacy-safe audit bundle. The thesis therefore contributes both a reusable supervision architecture and a bounded, use-case-specific control process rather than a universal benchmark standard or a guarantee of future behavioural consistency.

Keywords: GenAI applications, supervision pipeline, evaluation, safety, reliability.

Declaration of Generative AI Use

The author made limited and declared use of generative AI tools during the preparation of this thesis.

The following models were used during the work:

- OpenAI GPT-5.4 and GPT-5.5, used as the core models for the LLM-as-a-judge functionality in the experimental work
- Google Gemini 3.1 High
- Anthropic Claude Sonnet 4.6
- Anthropic Claude Opus 4.6

These tools were used for three purposes: to support the experiments reported in this thesis, to assist with parts of the program development, and to refine limited parts of the thesis text.

All research design decisions, implementation choices, experimental setup, analysis, interpretation, and the final submitted content remain the responsibility of the author.

— Eneko Retolaza Ardanaz, 16th July 2026

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context and Motivation

Large Language Models are moving from experimental prototypes to business-critical enterprise infrastructure. Unlike deterministic software, GenAI outputs are stochastic. Small configuration changes, such as prompt edits or model replacement, can cause drops in factual accuracy, instruction following, or response style. In enterprise settings, this creates a trust gap: teams need a reliable way to observe whether such changes materially alter benchmark behaviour.

1.2 Problem Statement

Traditional regression testing assumes deterministic behaviour and exact expected outputs. This assumption does not hold for LLM applications where multiple outputs can be semantically valid. Existing evaluation practice often relies on manual reviews or isolated benchmark runs, which are too slow and too fragmented for systematic comparison across iterations. The practical problem addressed in this thesis is the absence of a reliable, automated, and repeatable evaluation pipeline for non-deterministic GenAI applications that makes behavioural differences visible under model, prompt, and system changes.

For companies, scalable oversight is useful because it turns model and prompt changes into inspectable benchmark evidence. Instead of relying on slow manual spot checks, teams can repeatedly test representative scenarios, compare run behaviour across versions, focus expert review on ambiguous cases, and retain evidence for later analysis. This matters especially in enterprise settings where GenAI applications may affect customer communication, operational workflows, or compliance-sensitive processes.

1.3 Thesis Scope

This thesis focuses on the design and validation of a supervision pipeline for enterprise GenAI applications in the KPN context, where KPN is a Dutch telecommunications provider. The architecture is designed to remain as case agnostic as possible even though the strongest completed empirical study concerns one call-summary use case. In the current project, the workflow is instantiated with OpenLayer, an LLM evaluation platform used here as the benchmark engine, and company-specific platform handoffs such as Dataiku, a managed workflow platform widely used in the current KPN environment. The broader claim is architectural rather than universal: the same supervision pattern is intended to support both code-based and interactive applications as long as they can export records into a shared evaluation schema and consume the resulting evaluation artefacts. For the benchmark and minimum needs evaluation, the intended reusable element is not only the minimum-benchmark process and artefact shape, but also the method for separating benchmark-derived semantic reference signals from platform-visible proxy signals. The work therefore targets model-agnostic components so that judge models, application models, and serving platforms can evolve independently.

1.4 Contributions

- A supervision architecture for non-deterministic GenAI applications, designed to be case agnostic and centred on source adapters, a canonical evaluation schema, versioned benchmark profiles, execution, and audit-ready outputs.
- An implemented OpenLayer-centred template repository for code-based projects together with a documented Dataiku workflow for interactive enterprise evaluation.[1]
- A benchmark-based calibration of minimum viable tests for a concrete call-summary use case, reported through privacy-safe derived evidence from sensitive internal company data.
- A comparative alignment study that tests which OpenLayer built-in metrics and metric combinations best track semantic acceptability signals in the local 45-case benchmark and in a separate 100-case public sample.
- A reusable minimum-benchmark process package for summarisation use cases, including a standard process note, a reusable template, and a privacy-safe audit bundle.
- A governance-aware rationale that maps architecture components to regulatory and audit requirements relevant for enterprise evaluation practice.

1.5 Thesis Outline

Chapter 2 presents the state of the art and related work, Chapter 3 defines research objectives and questions, Chapters 4–6 describe the pipeline design and methodology, Chapters 7–8 present and discuss results, and Chapter 9 concludes. Detailed rubrics, reporting templates, disagreement categories, and governance mappings are kept in the appendix so the main text remains focused on the research argument.

[1] The Dataiku adapter implementation is available as a standalone repository at https://github.com/enekoreto/scalable-oversight-dataiku-adapter.

Chapter 2

State of the Art and Related Work

2.1 Review Scope and Selection Strategy

This chapter follows a targeted narrative review strategy focused on evaluation and supervision of enterprise GenAI systems. The selection includes foundational metric papers (BLEU and ROUGE), recent LLM-era evaluation work, LLM-as-a-Judge reliability studies, safety and hallucination assessment papers, and CI/CD-oriented evaluation pipeline papers (Chang et al. 2023; Criscione and Lekies 2024; Gu et al. 2025; Lin 2004; Papineni et al. 2001; Sohapy 2024; Zhang, Cui et al. 2025). Priority was given to sources that provide reproducible methods, quantitative evidence, or deployment-relevant guidance. Given the pace of the field, both peer-reviewed publications and high-impact preprints were included. Works focused only on model pre-training efficiency, without direct implications for evaluation or supervision, were excluded.

2.2 Enterprise LLM Deployment and Evaluation Pressure

Enterprise evaluation pressure is not mainly a story of higher benchmark scores; it is a consequence of faster iteration on systems whose behaviour can shift after changes to the model, prompt, retrieval stack, or orchestration logic. In non-deterministic GenAI applications, those changes are difficult to validate with static benchmark snapshots alone. As a result, evaluation becomes an operational bottleneck: organisations need recurring evidence that updates preserve quality, safety, and task-specific requirements before release (Criscione and Lekies 2024; Sohapy 2024).

This creates a shift from occasional benchmarking to continuous supervision. Work on continuous GenAI evaluation argues for combining offline tests, online monitoring, automated quality assessment, and human review in one recurring process aligned with service objectives (Sohapy 2024). Related regression-testing disclosures propose reusable test suites, unified model interfaces, and CI/CD integration to detect safety or quality regressions before release (Criscione and Lekies 2024). In parallel, DSPy shows how LLM workflows can be compiled and optimised as modular pipelines, reinforcing the need for equally modular evaluation stacks (Khattab et al. 2023).

2.3 Regulatory and Audit Drivers

Beyond model quality, enterprise evaluation architecture is increasingly shaped by governance and assurance requirements. In the EU context, the AI Act introduces a risk-based regulatory model and, for high-risk systems, emphasises obligations such as risk management, activity logging for traceability, technical documentation, human oversight, and robustness/cybersecurity controls (Commission 2026a; Parliament and European Union 2024). The same policy framework states that the AI Act entered into force on 1 August 2024 and became broadly applicable from 2 August 2026, with staged obligations by risk category (Commission 2026a).

Data governance obligations add a second design pressure. The GDPR accountability principle requires organisations not only to comply, but to demonstrate compliance through appropriate governance mechanisms (Commission 2026b; Parliament and European Union 2016). In practice, this favours systems that preserve decision traceability, clear responsibility allocation, and reproducible evidence of how outputs were assessed.

Operational resilience obligations add a third pressure for enterprise environments. NIS2 highlights cybersecurity risk-management measures, incident reporting requirements, and management accountability for non-compliance in covered sectors (Commission 2026c; Parliament and European Union 2022). Together, these drivers strengthen the rationale for evaluation pipelines that generate auditable artefacts and clear escalation paths, not only scalar quality scores.

2.4 Metric Evolution: From Lexical Overlap to Semantic Assessment

Classical metrics such as BLEU and ROUGE were foundational for automatic text evaluation and were designed to provide fast, low-cost proxies for human judgement through overlap with reference outputs (Lin 2004; Papineni et al. 2001). Their value remains clear for comparability and automation. However, open-ended LLM tasks often admit multiple semantically valid outputs, which limits the usefulness of strict lexical matching as a sole release signal.

Recent evaluation literature therefore moves towards richer and more task-sensitive measurement. A broad survey of LLM evaluation highlights the need to assess not only task success but also reasoning quality, robustness, and social risk dimensions (Chang et al. 2023). In this context, reference-free and rubric-based evaluators have gained traction. G-Eval reports stronger human alignment than prior automatic approaches on summarisation and dialogue tasks (Y. Liu et al. 2023). Prometheus focuses on fine-grained rubric-conditioned scoring and reports high correlation with human and strong-model judgements under custom criteria (Kim, Shin et al. 2024). For retrieval-augmented generation, Ragas decomposes evaluation into retrieval and generation dimensions without requiring gold answers, which is especially useful in enterprise domains where labelled data are expensive (Es et al. 2025). For summarisation-specific factual consistency, QAFactEval applies QA-based verification between source content and generated summaries to improve faithfulness assessment (Fabbri et al. 2022).

QAFactEval is particularly relevant because it evaluates factual faithfulness through a structured question-answering pipeline rather than only lexical overlap. The method generates questions intended to cover summary claims, answers those questions using the source document and the candidate summary, and then scores agreement between the two answer sets. Compared with earlier QA-based factuality metrics, the paper reports improvements by combining stronger component choices, including question-consistency filtering and improved answer-overlap scoring, which increases robustness of the final factuality signal (Fabbri et al. 2022).

For enterprise call-summary assessment, this design is useful because it directly tests whether concrete statements in the summary are supported by the call transcript. In practical terms, it can serve as a dedicated factual-consistency axis alongside rubric-based or judge-based quality scoring. At the same time, QAFactEval should not be treated as a complete summary-quality metric by itself, since it primarily targets factual faithfulness and does not fully capture dimensions such as prioritisation, action-item usefulness, coverage balance, or writing quality (Fabbri et al. 2022).

2.5 State of the Art for Summary Evaluation Use Cases

Recent work on summary evaluation extends the QA-based line in several directions. In domain-specific settings, PlainQAFact adapts QA-style factuality evaluation to biomedical plain-language summaries and addresses elaborative explanation, where useful contextual information may be added beyond strict source compression (You and Guo 2026). In dialogue and conversation settings, TofuEval provides a focused benchmark for hallucination assessment in topic-oriented dialogue summaries (Tang et al. 2024), while AUTOSUMM proposes an end-to-end conversation summarisation framework with evaluation-aware design choices for enterprise-like workflows (Gupta et al. 2025). In the current thesis, TofuEval mainly serves as dialogue-summary literature context in the main narrative, whereas AUTOSUMM remains literature context rather than an executed benchmark result.

Another stream targets stronger factual inconsistency detection and long-form verification quality. UIEFID combines universal information extraction with LLM-based reasoning to detect summary inconsistencies at a structured information level (Li and Yu 2025). For long-form generations, VeriFact and FaStFact refine claim extraction and evidence verification pipelines to improve factuality assessment quality and efficiency (X. Liu et al. 2025; Wan et al. 2025). In parallel,

Prometheus 2 strengthens rubric-based LLM-as-a-judge evaluation, which is useful for summary dimensions that go beyond factuality, such as coherence, relevance, and instruction compliance (Kim, Suk et al. 2024).

Taken together, the state of the art for summaries now combines specialised factuality metrics, extraction- and evidence-based verification, and rubric-guided judge models. However, these components are usually evaluated independently. A key project use case in this thesis is summary supervision (including call summaries), so the practical objective is to combine these components into a single, auditable evaluation workflow rather than selecting only one metric family.

2.6 LLM-as-a-Judge: Scalability, Bias, and Calibration

LLM-as-a-Judge is now central to scalable evaluation. Foundational work using MT-Bench and Chatbot Arena reports that strong judges can reach agreement with human preferences at levels comparable to inter-human agreement in those settings (L. Zheng et al. 2023). Subsequent survey work consolidates this direction and emphasises reliability as the core unresolved issue (Gu et al. 2025).

Recent papers address this reliability gap from different angles. Panel-based judging replaces a single evaluator with a diverse model jury and reports lower intra-model bias with substantial cost reduction (Verga et al. 2024). Bias-focused analysis shows that judges may over-accept outputs, with high true-positive rates but weak rejection of invalid outputs, and proposes veto- and regression-based corrections (Jain et al. 2025). Causal Judge Evaluation introduces calibration against a smaller oracle-labelled subset and reports large ranking gains with quantified uncertainty at much lower labelling cost (Landesberg and Narayan 2026). In software engineering contexts, REFINE operationalises evaluator validation by generating controlled quality degradations and testing ranking fidelity, reporting substantial alignment improvements for selected judge configurations (Fandina et al. 2025). Lessons from process reward model development are also relevant here: annotation strategy and evaluation protocol can introduce hidden bias, so process-level quality controls matter as much as final-score optimisation (Zhang, C. Zheng et al. 2025).

2.7 Safety, Hallucination, and Truthfulness-Oriented Evaluation

Quality metrics alone are insufficient for enterprise risk management. Agent-SafetyBench broadens evaluation to agentic environments and reports that evaluated agents still show substantial safety weaknesses, indicating a persistent gap between capability and safe behaviour (Zhang, Cui et al. 2025). For hallucination and truthfulness, two complementary directions emerge. One line uses internal states: probing hidden activations can separate true and false statements with meaningful accuracy (Azaria and Mitchell 2023). Another line uses attention-map structure, where spectral features of attention graphs improve hallucination detection among attention-based methods (Binkowski et al. 2025). Together, these works support integrating dedicated safety and truthfulness checks into release workflows rather than treating them as optional post-hoc analyses.

2.8 State of the Art Limitations

Despite rapid progress, the current evidence base has practical limitations that matter directly for enterprise adoption. A large share of the most relevant work is very recent, and much of it still circulates first as preprints. This does not make the work unusable, but it does mean that independent replication across domains, organisations, and operational settings is still limited. As a result, reported gains in judge alignment, factuality detection, or hallucination screening cannot automatically be treated as stable design truths for production environments.

Transfer is a second limitation. Many methods are validated on benchmark suites that are clean, public, and task-specific, whereas enterprise applications often involve proprietary data, domain-specific terminology, evolving business rules, and mixed quality requirements that are harder to capture in a standard benchmark. A method that performs well on a research dataset may therefore still require recalibration before it becomes trustworthy in a concrete company setting.

The literature is also fragmented at the level of system design. Many papers optimise one layer of the problem, such as metric construction, judge debiasing, hallucination detection, or

evidence verification, without showing how these components should be combined into a complete deployment-ready supervision architecture. For this thesis, that limitation is decisive: the research problem is not only which individual evaluation techniques look promising, but how they can be integrated into a repeatable, auditable, and operationally usable pipeline for enterprise release decisions.

2.9 Synthesis and Research Gap

The literature establishes four robust design requirements for this thesis. First, lexical overlap metrics should be treated as supporting signals, while semantic, rubric-driven, and reference-free methods should carry most evaluative weight in open-ended tasks (Chang et al. 2023; Es et al. 2025; Kim, Shin et al. 2024; Lin 2004; Y. Liu et al. 2023; Papineni et al. 2001). Second, judge-based evaluation must be validated and calibrated, with explicit bias controls and uncertainty handling before it is used for release decisions (Fandina et al. 2025; Gu et al. 2025; Jain et al. 2025; Landesberg and Narayan 2026; Verga et al. 2024; L. Zheng et al. 2023). Third, safety and hallucination checks should be first-class components of supervision (Azaria and Mitchell 2023; Binkowski et al. 2025; Zhang, Cui et al. 2025). Fourth, evaluation should run continuously in a CI/CD-compatible loop rather than as an isolated experiment (Criscione and Lekies 2024; Khattab et al. 2023; Sohapy 2024).

Accordingly, the central gap is integration: the field provides strong individual components, but not a standardised, model-agnostic pipeline that combines these components into auditable release gating for enterprise GenAI systems. Closing this integration gap is the primary contribution target of this thesis. This integration gap is especially relevant for summary evaluation, which is one of the concrete use cases of the project.

2.10 Gap-to-Question Mapping

The identified gaps map directly to the two research questions in Chapter 3.

- **Gap A (need for robust semantic regression-detection components) → RQ1.** The literature shows the limits of lexical overlap and motivates reference-free semantic evaluation and rubric-based judging as architectural building blocks rather than optional add-ons (Es et al. 2025; Kim, Shin et al. 2024; Lin 2004; Y. Liu et al. 2023; Papineni et al. 2001).
- **Gap B (judge reliability, bias, and calibration) → RQ1 and RQ2.** Judge methods are scalable but require debiasing, evaluator validation, and uncertainty-aware calibration before they are suitable for operational gating or minimum-requirements decisions (Fandina et al. 2025; Gu et al. 2025; Jain et al. 2025; Landesberg and Narayan 2026; Verga et al. 2024; L. Zheng et al. 2023).
- **Gap C (lack of integrated, model-agnostic enterprise architecture) → RQ1.** Existing work offers components, but fewer complete designs that integrate orchestration, scoring, governance, and release decisions in one deployable pipeline (Criscione and Lekies 2024; Khattab et al. 2023; Sohapy 2024).
- **Gap D (insufficiently standardised minimum safety and consistency checks) → RQ2.** Safety and hallucination studies indicate what should be tested, but do not yet define a broadly accepted minimal operational test set for enterprise release gating (Azaria and Mitchell 2023; Binkowski et al. 2025; Zhang, Cui et al. 2025).

Chapter 3

Research Objective and Questions

3.1 Research Objective

The objective is to design and evaluate a supervision pipeline intended to be scalable, case agnostic, and model agnostic, and able to make material behavioural changes in enterprise GenAI applications visible on fixed benchmarks across model, prompt, or system updates.

3.2 Main Research Question

How can an automated supervision pipeline for non-deterministic GenAI applications be designed and evaluated to provide structured benchmark-based comparison of behavioural change in enterprise settings?

3.3 Sub-Questions

- RQ1: What architectural components are necessary for a scalable, model-agnostic, and case-agnostic evaluation pipeline that can support both code-based and interactive GenAI applications in cloud-native environments?
- RQ2: What minimum viable tests are needed to make material behavioural inconsistency visible in a concrete enterprise GenAI use case across model, prompt, or system updates?

To address RQ1, the thesis proposes a modular supervision architecture consisting of source-system adapters, a canonical evaluation schema, and audit-ready export mechanisms. While the architecture is designed to be conceptually model- and case-agnostic, it is instantiated and tested within the KPN corporate environment using OpenLayer as the benchmark engine and Dataiku as the workflow integration layer.

To address RQ2, the research relies on an empirical threshold-calibration study applied to a sensitive corporate call-summary dataset. By comparing the performance of academic semantic metrics against platform-visible proxies across both local enterprise data and a public sample, the study defines a procedure for establishing minimum viable evaluation tests. Crucially, this approach focuses on providing a reusable, privacy-safe calibration methodology rather than asserting that any specific numeric threshold constitutes a universal standard.

Chapter 4

Supervision Pipeline Design

4.1 Design Principles

The pipeline is designed around modularity, reproducibility, auditability, and bounded automation. Modularity ensures that source-specific extraction logic, benchmark configuration, scoring logic, and governance outputs can evolve independently rather than collapsing into one tool-specific workflow. Reproducibility matters because score changes are only interpretable if the evaluated scenarios, evaluator settings, and runtime metadata can be reconstructed later. Auditability matters because the intended output is not merely a metric dashboard, but a benchmark evidence package that can be inspected after review, comparison, or incident analysis. Bounded automation means routine evidence generation should be automated while borderline or high-impact cases remain reviewable rather than silently auto-resolved. Components should therefore be replaceable (for example, swapping judge models) without rewriting orchestration logic, and design choices must support enterprise governance requirements and traceable evaluation outcomes.

4.2 Architecture Overview

The architecture follows a reusable supervision workflow integrated with surrounding engineering systems. Its purpose is not to prescribe one platform, but to standardise how heterogeneous GenAI applications hand off data for evaluation, how benchmark settings are versioned, and how results are returned as auditable benchmark evidence. In the company setting, OpenLayer is currently the benchmark engine used to execute this workflow, but the surrounding architecture is broader than any single tool.

Figure 4.1 summarises the implemented workflow abstraction used in the thesis. The diagram focuses on the key workflow points that already exist in the project: source adapters, normalisation to a canonical evaluation schema, versioned benchmark configuration, execution, and outputs. This framing is important for RQ1 because it keeps platform-specific integrations separate from the reusable supervision pattern.

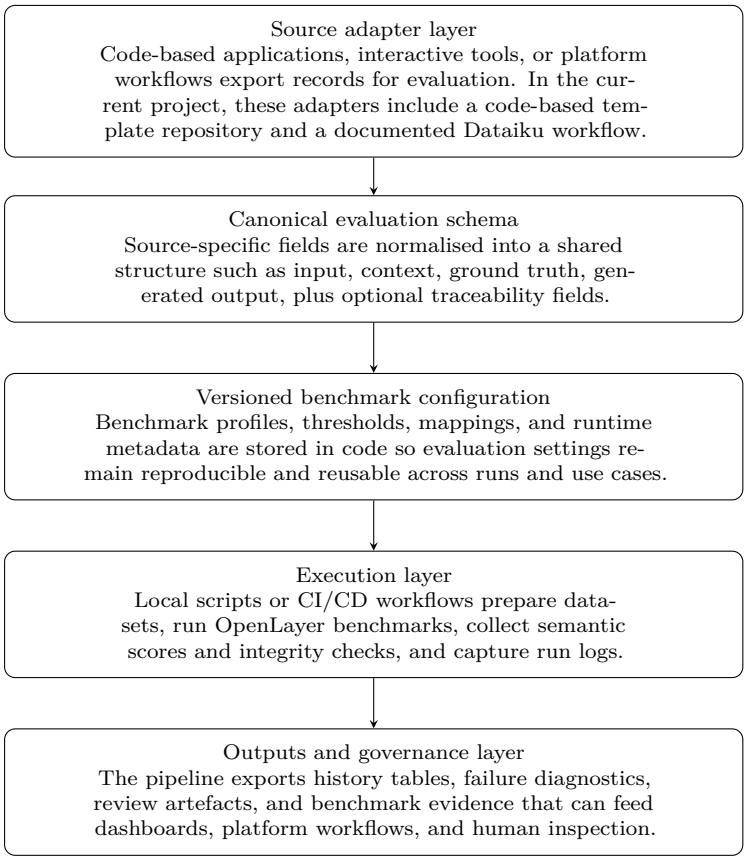

Figure 4.1: Thesis-authored abstraction of the implemented supervision workflow. The reusable architecture is organised around adapters, a canonical evaluation schema, versioned benchmark configuration, execution, and audit-ready outputs rather than around one specific platform.

4.3 Why These Components Are Necessary

RQ1 asks which architectural components are necessary for scalable supervision, not merely which components are convenient in one implementation. The five-layer design is treated as a minimal viable supervision architecture because each layer addresses a distinct enterprise evaluation need and removing any one of them weakens reuse, reproducibility, or governance.

Source adapters are necessary because application surfaces differ materially in how they structure evaluation data. Without adapters, each new system would require a unique evaluation workflow and the architecture would not scale beyond one integration path. The canonical evaluation schema is necessary because score interpretation cannot be reused if every source defines its own field semantics. Without schema normalisation, thresholds, benchmark profiles, and reports would remain tied to one application format rather than to the evaluation logic itself.

Versioned benchmark configuration is necessary because non-deterministic evaluation is only credible when scenario sets, mappings, thresholds, and evaluator settings can be reconstructed for later comparison. Without configuration versioning, a changed score could reflect benchmark drift rather than a genuine behavioural shift. The execution layer is necessary because an architecture that cannot be exercised through local runs, CI/CD workflows, or structured platform flows remains a documentation pattern rather than an operational supervision system.

Finally, outputs and governance artefacts are necessary because enterprise supervision is not complete when a score is computed. Meaningful comparison requires interpretable evidence, failure diagnostics, and records of what was evaluated under which settings. Without this output layer, the pipeline would produce analytics, but not a governable evaluation process. For this reason, the architecture is intentionally defined as a chain from ingestion to auditable evidence rather than as a benchmark runner alone.

4.4 Software Quality Attributes

As a software engineering artefact, the pipeline should be judged not only by whether it runs, but also by which architectural quality attributes it preserves when the environment changes. In this thesis, the most important attributes are scalability, adaptability, reusability, maintainability, portability, and traceability. These terms are used in an architectural sense rather than as blanket claims about organisation-wide scale or universal benchmark transfer.

Scalability means that additional application surfaces can be integrated without redesigning the supervision core. The architecture supports this by isolating source-specific extraction in adapters and by normalising records into a canonical evaluation schema. As a result, growth in the number of use cases should primarily require new mappings at the boundary rather than a new evaluation workflow for each system.

Adaptability and portability mean that local execution details may change while the core interpretation of evaluation remains stable. In the current project, this applies to changes in benchmark engines, platform environments, and orchestration surfaces such as repository workflows or Dataiku flows. The design goal is therefore not to freeze one toolchain, but to keep the architectural invariants stable: the schema, benchmark configuration, interpretation logic, and audit-ready outputs.

Reusability and maintainability follow from modular separation. Benchmark profiles, thresholds, adapters, execution logic, and reporting artefacts are not collapsed into one script or one platform configuration. This separation reduces the cost of changing one concern at a time, such as replacing a judge model, updating a dataset mapping, or tightening a comparison rule, without forcing a full redesign of the rest of the workflow.

Traceability is treated as a first-class quality attribute because enterprise supervision is only useful if results can be linked back to the evaluated input, benchmark version, execution context, and final evaluation outcome. For that reason, the architecture stores canonical fields, versioned settings, and exported evidence together rather than treating them as optional metadata. In practice, this allows the pipeline to support both engineering reuse and governance review.

4.5 Reusable Interfaces and Platform Adapters

The key RQ1 design decision is to separate the reusable supervision core from platform-specific adapters. A code-based application can create evaluation rows from a repository workflow, while an interactive application can hand off the same information through a platform flow such as Dataiku, which is widely used in the current KPN workflow environment. In this context, a platform is a managed workflow environment that organises datasets, recipes, execution steps, and user-triggered runs behind a higher-level interface rather than exposing only source code or service endpoints (Dataiku 2025). In both cases, the adapter layer maps source-specific fields into the canonical evaluation schema before benchmark execution. This allows the same benchmark profiles, score interpretation rules, and reporting logic to be reused without rewriting the full pipeline for each application type.

Figure 4.2 shows one concrete Dataiku instantiation of this adapter pattern in the current project. Separate model-version datasets are first prepared inside the platform and then merged through a Python step into a shared benchmark output that can be synchronised downstream. The important architectural point is not the specific recipe icons or dataset names, but the preservation of the adapter boundary: Dataiku performs source-local preparation while the exported output still acts as a canonical handoff surface for shared benchmark logic.

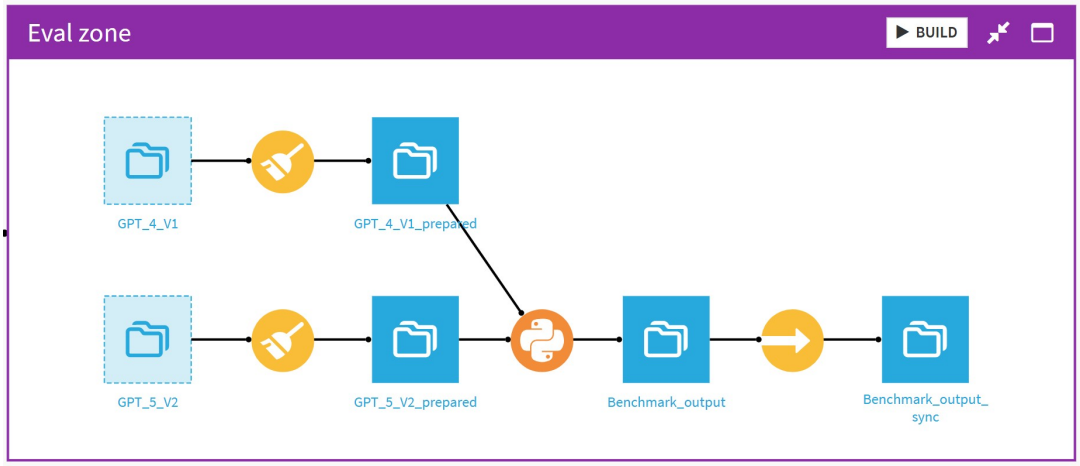

Figure 4.2: Illustrative Dataiku evaluation flow used as an interactive-platform adapter in the current project. Separate model-version datasets are prepared inside the platform and merged into a shared benchmark output before downstream synchronisation.

In a Dataiku-style flow, that also means users can inspect how results shift across prompt versions, prompt parameters, model versions, and dataset versions instead of treating each run as an isolated score snapshot. Figure 4.3 illustrates such a comparison view for two prepared model-version runs, where benchmark-visible metrics are grouped by dataset version inside the platform. This kind of view is practically useful because it gives non-developer users a quick way to inspect whether a new run changes proxy quality signals before those outputs are handed back into the wider supervision process. The screenshot is therefore an illustrative workflow example rather than the ranked evidence used later to identify the strongest local and public alignment signals. It does not replace the offline semantic reference layer used elsewhere in the thesis, but it does provide a practical comparison surface inside the interactive workflow.

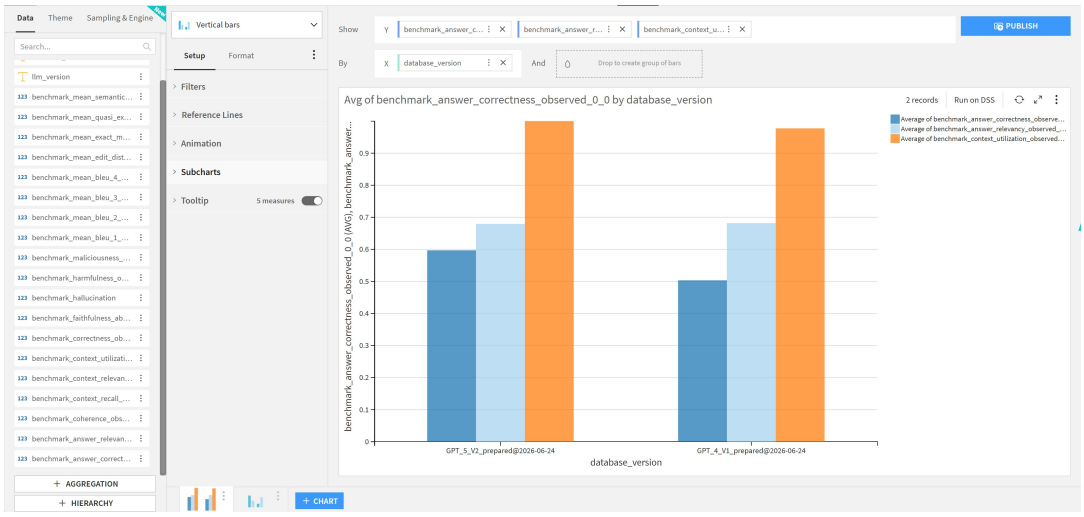

Figure 4.3: Illustrative Dataiku comparison view for two prepared benchmark runs. The platform groups benchmark-visible metrics by dataset version so users can inspect how model-version changes affect observed signals without leaving the workflow environment.

Enterprise GenAI systems are not all operated in the same way. Some are code-first applications that can export evaluation rows directly from a repository, script, or service pipeline. Others are embedded in interactive data or ML platforms where users prepare datasets, run recipes, and publish outputs through a managed flow. The thesis treats these operational differences as adapter concerns rather than as reasons to redesign the evaluation core for each use case.

The adapter boundary is intentionally strict. Adapters may change how raw source fields are extracted, renamed, enriched with traceability metadata, or batched for evaluation, but they do not redefine benchmark semantics, threshold policy, or reporting structure. Those remain in the shared configuration and interpretation layers. This boundary is what allows a Dataiku workflow and a code-based template repository to reuse one evaluation core instead of merely sending data to the same reporting destination.

This boundary also constrains how benchmark meaning is represented. In the current call-summary use case, the authoritative 45-case benchmark is not just a table of transcripts and summaries; it also contains structured reference fields about key business questions and outcomes. Those fields contribute to the offline calibration target, while the later public comparison remains analytically separate from the local comparison configuration. An adapter is therefore semantically faithful only if it preserves that structure in the schema, for example through explicit reference columns or a versioned reference bundle consumed by custom metrics. Collapsing the benchmark to a single free-text ground-truth summary may be operationally convenient, but it changes the evaluated object and weakens direct comparability with the calibrated offline reference configuration.

The current implementation already demonstrates two adapter styles for the same KPN call-summary use case. One is a code-based template repository around OpenLayer, including local execution, CI/CD execution, and exported result tables for code-based projects. The other is a documented Dataiku workflow that prepares the same canonical fields before running benchmark logic for interactive projects. In practical KPN terms, the current deliverable is therefore not only architectural text, but two concrete summary-evaluation deliverables built on the same supervision core: a code-based template repository and a documented Dataiku workflow. The planned broader template repository should therefore be understood as future generalisation of an architecture that is already visible in the working project, rather than as the first moment at which the architecture exists.

The packaging story is therefore partial rather than absent. A fully generic template repository is still future work, but the current project already exposes reusable packaging surfaces in practice: a standard minimum-benchmark process note, a reusable benchmark template, and a privacy-safe RQ2 bundle that captures the current calibration logic and audit artefacts without exposing sensitive benchmark content.

One concrete use case considered in this project is summary evaluation, in particular enterprise call summaries. Therefore, scenario design and evaluation criteria include summary-specific checks such as factual faithfulness to source transcripts, coverage of key decisions and action items, and concise but accurate phrasing. The point of the architecture, however, is that these task-specific checks sit on top of a reusable pipeline rather than defining a specific pipeline for summaries only.

4.6 Metric Rationale: Beyond Lexical Overlap

Traditional metrics such as BLEU and ROUGE estimate quality by lexical overlap with reference texts (Lin 2004; Papineni et al. 2001). This is useful for constrained generation tasks, but it is insufficient as the main comparison basis for open-ended LLM applications.

In non-deterministic tasks, several outputs can be equally valid while using different wording. A strict overlap metric can therefore under-score correct paraphrases. The inverse failure mode also exists: an output may share many words with a reference yet still be factually wrong, unsafe, or misaligned with instructions. For enterprise supervision, this creates unacceptable false negatives and false positives if lexical matching is used alone (Chang et al. 2023).

Semantic and source-grounded evaluation address this limitation by scoring meaning-level quality dimensions, such as faithfulness, instruction following, usefulness, and reasoning quality, without relying on exact lexical match to one gold phrasing (Kim, Shin et al. 2024; Y. Liu et al. 2023). For retrieval-augmented or summary workflows, specialised checks can further separate retrieval quality, answer grounding, and factual consistency (Es et al. 2025; Fabbri et al. 2022). Accordingly, this thesis treats lexical overlap metrics as optional supporting diagnostics and uses semantic, judge-based, and rule-based signals as the primary basis for behavioural comparison and benchmark interpretation.

Architecturally, these metric choices are attached to benchmark profiles rather than hardwired into adapters or orchestration. This matters for RQ1 because it keeps the supervision core metric-

flexible: the summary use case can emphasise factual consistency today, while future use cases can introduce different semantic or rule-based checks without redesigning ingestion, execution, or reporting.

4.7 Governance Integration and Audit Rationale

The architecture is designed not only to compute quality metrics, but also to create evidence for governance review. This is the rationale for design choices such as scenario versioning, stored prompts and outputs, score traces, human escalation, and evaluation evidence packages. These design choices support traceability, human oversight, demonstrable accountability, and operational risk management in line with EU AI Act, GDPR, and NIS2 expectations (Commission 2026a,b,c; Parliament and European Union 2016, 2022, 2024). The detailed requirement-to-control mapping is placed in Appendix A.6 to keep the core architecture readable.

Each architectural component therefore carries a governance function in addition to a technical one. Adapters and the canonical schema support traceability from the source system to the evaluated record. Versioned benchmark configuration supports accountability by freezing the exact test logic behind an evaluation conclusion. Execution logs and exported diagnostics support auditability and post-incident analysis. Review routing supports human oversight by ensuring uncertain cases remain contestable rather than silently auto-resolved. Governance is therefore embedded in the workflow itself rather than appended after scoring.

4.8 Operationalisation in the Target Environment

In this thesis, the pipeline is operationalised in the KPN environment with OpenLayer as the current benchmark engine and with a code-based template repository plus documented Dataiku-based interactive workflows (Dataiku 2025). These platforms are examples of the current company setting, not requirements of the research design. More generally, the architecture is intended to remain usable wherever an organisation can export evaluation records, normalise them into the canonical schema, execute benchmark logic, and feed the resulting artefacts back into evaluation or review processes. The same architecture can therefore support both code-based and interactive applications as long as records can be mapped into the canonical evaluation schema and the resulting evaluation artefacts can be consumed by the surrounding workflow environment. The implementation emphasises compatibility with CI/CD practices, containerised execution, and later packaging into a reusable template repository.

For that reason, the current evidence should be read in two layers. The implemented workflow already supports repeatable execution, traceable exports, and privacy-safe packaging of the current RQ2 benchmark logic. What remains future work is broader generic packaging for faster onboarding of additional use cases.

The architecture remains portable because only some elements are allowed to vary locally. Source extraction, environment-specific runners, and downstream workflow integration may change from one organisation or platform to another. The architectural invariants are the canonical evaluation schema, the versioned benchmark configuration, the interpretation logic for benchmark outputs, and the audit-ready export structure. In other words, changing the operating environment should change adapters and invocation details, not the meaning of the evaluation.

This distinction is also what keeps the thesis case agnostic and service provider agnostic. If OpenLayer were replaced by another benchmark engine, or if Dataiku were replaced by another managed workflow platform, the research claim would still stand as long as the replacement could consume the shared schema, execute the intended benchmark logic, and return equivalent evaluation artefacts. The architecture is therefore defined at the level of interfaces, workflow rules, and evidence packages rather than at the level of vendor features.

Chapter 5

Methodology

5.1 Research Design

This thesis uses Technical Action Research, combining artefact construction with practical intervention in an enterprise context. The method follows an iterative regulatory cycle: diagnose, design, implement, and evaluate.

This approach is appropriate because the research problem is not only whether a metric performs well on a static benchmark, but whether a supervision pipeline can be made useful for repeatable benchmark-based inspection of behavioural change. Technical Action Research supports this by linking design choices to observed organisational needs: diagnose the current evaluation gap, design a pipeline response, implement a prototype, and evaluate whether the artefact makes model, prompt, or system differences inspectable on a fixed benchmark. The cycle also fits the KPN context, where scenario selection, governance expectations, and workflow constraints are likely to be refined as the prototype is tested.

Because the two research questions are methodologically different, the evaluation is split into two explicit tracks. RQ1 is treated as a development and architecture-validation question, while RQ2 is treated as a benchmark-calibration and benchmark-comparison question for one concrete use case. Keeping these two tracks separate is necessary to avoid presenting architectural design evidence, threshold evidence, and benchmark comparison as if they were the same kind of result.

5.2 Methodology for RQ1

RQ1 asks which architectural components are necessary for a scalable, case-agnostic supervision pipeline. This question is not answered by optimising one benchmark score on one dataset. Instead, it is answered through artefact-oriented design validation: the architecture is considered adequate if it isolates the reusable evaluation core from platform-specific handoff logic and if the same supervision pattern can be instantiated across different application surfaces without changing the result.

For that reason, the unit of analysis for RQ1 is the workflow architecture itself. The main evidence is implementation and configuration evidence rather than benchmark performance. The evaluation examines whether the current project provides a stable separation between source adapters, a canonical evaluation schema, versioned benchmark configuration, execution paths, and exported artefacts. These components were not just defined; they were traced in the implemented template-repository structure, repo-managed benchmark profiles, local runners, CI/CD flow, exported result tables, and the documented Dataiku workflow path.

The RQ1 procedure had four steps. First, architectural evaluation criteria were derived from the literature and enterprise evaluation constraints: modularity, reproducibility, auditability, model agnosticism, and support for both code-based and interactive application workflows. Second, these criteria were translated into explicit interface boundaries, especially the separation between source adapters and the evaluation schema. Third, the implemented project was examined to determine whether the same benchmark logic could be exercised through more than one operational path, namely a code-based template repository and a Dataiku-oriented interactive workflow. Fourth, the architecture was assessed at the artefact level by checking whether benchmark configuration,

execution, and exports remained stable across these entry points. As a limited practical validation of the interactive-platform path, one researcher also used the Dataiku workflow for a summary-evaluation use case to compare and contrast run results, providing direct evidence that the user-facing comparison flow could be operated in the intended setting.

The resulting RQ1 claim is intentionally bounded. The thesis does not claim that every future enterprise use case has already been onboarded, nor that a broader cross-use-case template repository is already complete. Instead, it claims that the current implementation already exhibits a reusable supervision pattern that can be packaged further without changing its architectural core.

5.3 Methodology for RQ2

RQ2 asks which minimum viable tests are needed to screen for material behavioural inconsistency in one concrete enterprise GenAI use case. Unlike RQ1, this question requires use-case-specific empirical calibration and an explicit check of how platform-visible benchmark signals relate to the same acceptability pattern. The methodological objective is therefore twofold: calculate local results with the current academic industry standard processes, and examine which OpenLayer-visible metrics best approximate semantic acceptability under both the local enterprise benchmark and a separate public comparison surface.

The threshold-setting source of truth for RQ2 is a 45-case benchmark for the KPN call-summary use case stored in an internal benchmark workbook maintained in the project artefacts. Each case represents one summary evaluation built around a source call transcript, a candidate generated summary, and structured reference information used to assess whether key facts, actions, and claims were preserved faithfully. In that sense, the dataset is not just a collection of summaries, but a fixed calibration table that allows the same enterprise cases to be scored under different metrics, thresholds, and decision rules.

More specifically, the benchmark is not based on an unconstrained judgement of whether a summary sounds generally good. Each case is anchored in predefined business information questions encoded through structured reference fields, such as the call reason, the follow-up action, the outcome, and related issue-specific fields. The evaluation task is therefore whether a generated summary preserves the answers to those concrete questions faithfully enough for benchmark-based comparison across system changes.

Because the source material contains sensitive internal company information, the benchmark remains inside the company environment and the thesis reports only privacy-safe derived outputs. This makes the study deliberately bounded, but still methodologically defensible. The calibration is anchored in UIEFID and QAFactEval, treated here as the best-available academic and industry-relevant reference standards for factual summary assessment in the present thesis context (Fabbri et al. 2022; Li and Yu 2025). The research contribution therefore lies in the calibration procedure, the resulting minimum-requirements benchmark configuration, and the benchmark-to-platform comparison logic rather than in publication of raw customer examples.

In addition to the local 45-case threshold-setting source, RQ2 includes a second comparison surface: a sampled 100-case public dataset branch derived from a 34-row QAConv base sample (Wu et al. 2022). This branch is used to study how OpenLayer built-in metrics behave under a different benchmark structure. In that branch, the same semantic metric families, UIEFID and QAFactEval, are used as reference signals for acceptable versus not-acceptable comparisons, and OpenLayer single metrics, pairwise combinations, and three and four metric combinations are ranked with Cohen's kappa as the primary comparison criterion, while AUC and balanced accuracy are retained as supporting diagnostics. In this thesis, kappa is prioritised because the practical objective is agreement with benchmark acceptability labels rather than score separation alone, and kappa adjusts that agreement for chance on a modest binary subset with some label imbalance. Plain correlation would capture continuous score co-movement, but it would not express the thresholded acceptable-versus-not-acceptable agreement that matters for this comparison task. The OpenLayer metric names referenced in these comparisons follow the platform's official tests overview and aggregate-metrics documentation (OpenLayer 2026a,b). This public comparison branch is not used to tune the KPN thresholds; its role is to test whether the OpenLayer signals that appear useful on one benchmark surface remain useful on another.

The core local-calibration procedure had seven steps. First, the fixed 45-case benchmark was scored with the two selected semantic factuality metric families, UIEFID and QAFactEval. Second, after manual review of the benchmark cases, the final human labels were consolidated into pass, fail, and review outcomes. The 32 cases with explicit binary pass/fail decisions were used for threshold fitting, while the 13 cases that remained `review` were retained as deliberately ambiguous cases and excluded from the binary calibration subset. Third, candidate thresholds were swept across the observed score distribution for each metric, reusing the existing per-case metric scores so the recalibration could be refreshed without rerunning the expensive semantic backends. Fourth, thresholds were selected using an explicit policy: prefer the lowest threshold satisfying the recall-on-bad and good-case-fail-rate constraints, and otherwise fall back to the threshold with the best Cohen's kappa, with balanced accuracy retained as a secondary diagnostic. The targets were manually chosen for this use case to catch most bad summaries while limiting unnecessary escalation of good ones, rather than taken from the metric papers themselves. The broader calibration-and-validation logic follows recent work on calibrated surrogate metrics, judge-bias control, and evaluator validation (Fandina et al. 2025; Jain et al. 2025; Landesberg and Narayan 2026). Fifth, bootstrap resampling was used to test whether the selected thresholds were stable. Sixth, the thresholds were turned into a review-oriented screening configuration by adding integrity checks and a review margin. In other words, the final reported threshold results come from semantic scores calibrated against explicit human pass/fail labels on the binary subset, while the `review` cases remain a separate ambiguity category rather than being forced into the binary fit. Seventh, OpenLayer was used to see which parts of this configuration could be reflected reliably on the platform surface.

Two additional comparison procedures were then run. First, the same 45-case benchmark was analysed at the OpenLayer-metric level by ranking built-in single metrics, pairwise combinations, and three and four metric combinations against the local acceptable versus not-acceptable pattern on the 32 binary human-labelled cases, again with Cohen's kappa treated as the primary ranking criterion. Second, a sampled 100-case public branch was prepared, scored with the same metric families, and subjected to the same OpenLayer ranking analysis. The resulting local-versus-public comparison tables then measured how far metric relevance rankings shifted across the KPN enterprise benchmark and the public comparison surface.

This design produces four linked evidence layers. The first is the authoritative KPN calibration result from the fixed 45-case benchmark. The second is the local benchmark-to-OpenLayer alignment analysis that measures which built-in signals best approximate the local semantic acceptability pattern. The third is the 100 case public benchmark-to-OpenLayer comparison branch, which tests how strongly those metric rankings transfer to a different environment. The fourth is a reusable method package, consisting of a standard minimum-benchmark process note, a benchmark template, and a privacy-safe audit bundle that captures the current calibration artefacts without exposing raw company data.

The review-packet workflow remained methodologically important because it provided the path by which borderline cases were inspected and merged back into a consolidated evaluation workbook with stable identifiers and provenance fields. In the final rerun, those merged human labels became the authoritative source for calibration. At the same time, the completed review did not collapse every case into a binary label: 13 cases remained `review`. Those cases therefore stay inside the final benchmark-level comparison analysis, but they are excluded from the binary threshold fit itself.

The reuse claim is therefore methodological rather than threshold based. The workflow shape can be reused for future summarisation use cases, but the selected semantic metrics, numeric cutoffs, review margin, and platform-safe fallback checks must be recalibrated against each new benchmark rather than copied forward unchanged.

5.4 Validity and Ethics

Threats to validity differ across the two questions. For RQ1, the main threats are overgeneralisation from one company implementation, confusion between platform-specific convenience and general architectural necessity, and premature reliance on future packaging work as if it were already completed evidence. These risks are mitigated by keeping the RQ1 claim at the level of reusable architecture rather than universal benchmark performance, by separating the core pipeline from

its current adapters, and by treating the planned template repository as future work rather than current proof.

For RQ2, the main threats are judge bias (for example verbosity, position, or agreeableness bias), limited scenario coverage, residual ambiguity in the 13 cases that remained `review`, domain transfer effects, and comparison-surface mismatch (Jain et al. 2025). Mitigations include transparent scoring criteria, explicit reporting of the reviewed-label calibration subset, bootstrap stability checks, kappa-first threshold selection, and a recorded review-packet workflow with merge-back provenance for the borderline cases (Landesberg and Narayan 2026). The benchmark-to-platform comparison work is additionally constrained by implementation-surface differences between the reviewed local 45-case branch and the 100-case public branch. That means the public-versus-local metric contrast should be read as evidence of transfer instability, not as a perfect ceteris paribus experiment. Enterprise privacy and governance constraints are treated as first-class design requirements, which is why the RQ2 calibration uses sensitive internal company data only inside the company environment and the exported thesis evidence is privacy-safe and avoids raw customer text.

Chapter 6

Experimental Setup

6.1 System and Environment

The experimental environment targets enterprise GenAI applications in the KPN setting. The prototype is implemented in Python and executed through containerised workflows so it can integrate with CI/CD pipelines, code-based automation through the template repository, and interactive platforms. For RQ1, the relevant system evidence is the implemented OpenLayer-centred template repository plus its documented adapter paths. For RQ2, the empirical work was operationalised in two main comparison cases rather than as a single benchmark run. A second comparative case is a 100-case public sample derived from a 34-row QAConv base sample (Wu et al. 2022), used to study how OpenLayer built-in metrics align with the same semantic metric families under a different benchmark surface. Because the KPN workbook contains sensitive internal company information, the thesis reports only privacy-safe derived outputs.

6.2 RQ1 Implementation Surface

RQ1 was evaluated on the implemented architecture rather than on one benchmark score. The current project exposes a reusable supervision pattern with five concrete layers: source adapters, a canonical evaluation schema, versioned benchmark profiles, execution paths, and exported result artefacts. Two adapter styles are already documented. The first is a code-based template repository that prepares datasets locally or in CI/CD and runs OpenLayer benchmarks through versioned configuration. The second is a documented Dataiku workflow that maps source columns to the same canonical evaluation schema before benchmark execution. This means the research claim is platform-agnostic even though the current company setting uses specific tools.

The broader cross-use-case template repository is not yet complete, so the current RQ1 evidence comes from these working delivery surfaces rather than from a fully generic reusable scaffold. The thesis therefore evaluates the architectural pattern that is already implemented and documented, while treating later packaging work as future engineering consolidation.

6.3 RQ2 Development Logic

RQ2 was developed as the point where the general supervision architecture had to be translated into a concrete release-control rule. The underlying problem was practical: a reusable pipeline is not sufficient for deployment governance unless it can be reduced to a bounded set of checks that distinguishes clearly acceptable outputs, clearly unacceptable outputs, and ambiguous outputs that should be escalated. For that reason, RQ2 was framed not as a search for a universal benchmark, but as a calibration problem for one concrete enterprise use case together with a comparison problem about which platform-visible benchmark signals remain relevant across different benchmark surfaces.

The development of RQ2 followed six deliberate design choices. First, the question was narrowed to the KPN call-summary use case, because this task combines high operational relevance with a realistic failure mode: summaries can look fluent while still skipping key actions or introducing unsupported claims. Second, the evaluation was anchored in semantic factuality rather than lexical

overlap, because the use case allows multiple valid phrasings but does not accept unsupported content. Third, the study used a fixed local benchmark rather than ad hoc manual examples, so the resulting gate could be justified by repeatable evidence rather than retrospective judgement. Fourth, the design separated semantic calibration from platform enforcement, because the benchmark should determine what quality means before the serving platform is allowed to constrain what can be checked operationally. Fifth, the result was intentionally framed as a candidate minimum-requirements gate, since the available evidence was strong enough for bounded calibration but not yet strong enough for a universal or sharply automated standard. Sixth, the study did not stop at threshold setting: it also analysed which OpenLayer-visible metrics and multi-metric combinations best approximate semantic acceptability in both the 45-case local benchmark and a separate 100-case public comparison sample.

In practical terms, this means the benchmark does not treat the summary as a free-form artefact judged only holistically. The use case is already structured around predefined business information questions, and a summary is acceptable only if it preserves the answers to those questions well enough for operational use, for example what the call was about, what follow-up was required, and what outcome was reached.

This development logic matters for interpretation. The thesis is not claiming that RQ2 discovered a final generic test suite for all enterprise LLM systems. It is claiming that minimum viable tests can be derived in a disciplined way: start from a real use case, choose metric families for defensible reasons, calibrate on a fixed benchmark, add explicit review routing, package the resulting artefacts in a reusable minimum-benchmark format, and then test both what remains operationally enforceable in-platform and which OpenLayer-visible signals are actually relevant on local and public benchmark surfaces.

Accordingly, RQ2 is not presented as a threshold sweep alone. It yields a local screening gate, a benchmark-to-OpenLayer alignment study on the KPN benchmark, a matched public comparison branch, and a privacy-safe audit bundle. A standard process note and reusable template keep the workflow reusable for future use cases, while leaving the actual thresholds, review rules, and proxy-signal rankings benchmark specific.

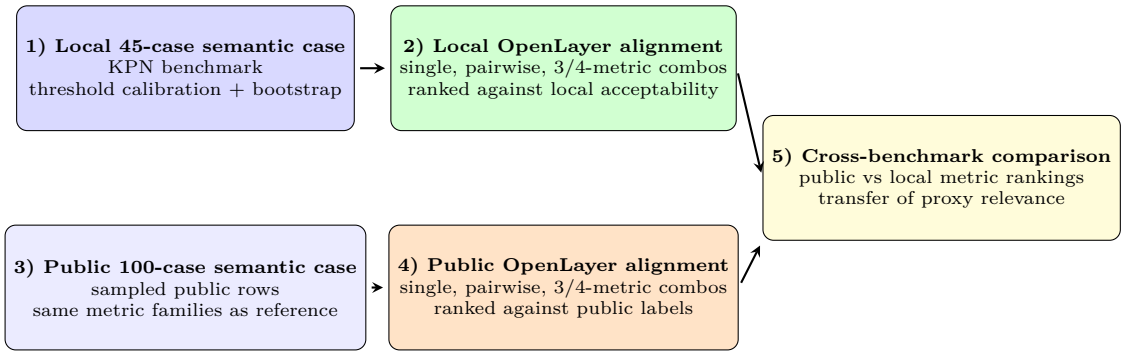

Figure 6.1: RQ2 two-case design: local threshold setting, local and public OpenLayer alignment studies, and cross-benchmark comparison.

6.4 Metrics

The RQ2 calibration focused on two semantic factuality metrics: UIEFID and QAFactEval. For each metric, the calibration workflow computed candidate thresholds over the observed score distribution and recorded recall on bad cases, good-case fail rate, balanced accuracy, Cohen's kappa, and pass rate. The explicit selection policy was: choose the lowest threshold satisfying $\text{recall}_{\text{bad}} \geq 0.8$ and good_fail_rate ≤ 0.25; if no threshold satisfies both constraints, fall back to the threshold with the best Cohen's kappa. These values were manually chosen as pragmatic operating targets for this use case rather than taken from the metric literature or platform defaults: the 0.8 recall target prioritises catching most materially bad summaries, while the 0.25 good-case fail-rate cap limits how many acceptable summaries are unnecessarily escalated to review. They are therefore study-specific operating targets for a conservative screening gate on the reviewed binary

subset, not literature-derived constants. The broader use of explicit calibration targets, bias-aware evaluator control, and post-selection stability checking is consistent with recent work on calibrated surrogate metrics and evaluator validation (Fandina et al. 2025; Jain et al. 2025; Landesberg and Narayan 2026). Threshold stability was then checked with 300 bootstrap samples.

The two metrics were chosen because they capture complementary parts of the failure surface. UIEFID operates at a structured inconsistency level and is useful when a summary preserves the general topic but distorts specific information. QAFactEval approaches the same problem from a question-answering perspective and is useful when the key issue is whether concrete claims in the generated summary can be supported by the source transcript. Using both avoids treating one metric family as a complete theory of summary quality while still keeping the calibration narrow enough to be operational.

In RQ2 these metric families play two different roles. In the 45-case local case, the existing per-case semantic scores were reused in a fast recalibration path after manual review so thresholds and local ranking analyses could be refreshed without rerunning the expensive semantic backends. In the 100-case public comparison case, the same metric families were used as the semantic reference surface for the sampled public rows. The resulting public-versus-local comparison should therefore be interpreted at the level of metric families and benchmark surfaces rather than as a strict backend-controlled ablation.

For reproducibility, the calibration logic can be written in compact form. Let $i \in \{1, \dots, N\}$ index benchmark cases and let $z_i \in \{0, 1\}$ denote the calibration target label, where $z_i = 1$ indicates a bad case and $z_i = 0$ a good case. For metric m with score $s_{i,m}$ and candidate threshold t, define a fail prediction

$$\hat{z}_i^{(m,t)} = \mathbf{1}[s_{i,m} < t].$$

The threshold sweep evaluates at least three quantities:

$$\mathrm{Recall}_{\mathrm{bad}}(m,t) = \frac{\sum_i \mathbf{1}[z_i = 1 \wedge \hat{z}_i^{(m,t)} = 1]}{\sum_i \mathbf{1}[z_i = 1]},$$

$$\mathrm{GoodFailRate}(m,t) = \frac{\sum_i \mathbf{1}[z_i = 0 \wedge \hat{z}_i^{(m,t)} = 1]}{\sum_i \mathbf{1}[z_i = 0]},$$

$$\mathrm{BA}(m,t) = \frac{\mathrm{TPR}(m,t) + \mathrm{TNR}(m,t)}{2}.$$

In the reviewed rerun, Cohen's kappa $\kappa(m,t)$ from the same confusion matrix was treated as the primary fallback selection score, while balanced accuracy was retained as a secondary diagnostic. The selection policy is then

$$t_m^* = \min\{t : \mathrm{Recall}_{\mathrm{bad}}(m,t) \geq 0.8 \ \wedge \ \mathrm{GoodFailRate}(m,t) \leq 0.25\}$$

when the feasible set is non-empty; otherwise

$$t_m^* = \arg\max_t \kappa(m,t).$$

If multiple thresholds attain the same maximum κ, the lowest threshold is retained. After selecting $(t^*_{\mathrm{UIEFID}}, t^*_{\mathrm{QAFactEval}})$, the review margin uses $\delta = 0.02$ with absolute-distance routing: review if $\exists m : |s_{i,m} - t_m^*| \leq \delta$ or if exactly one semantic metric passes.

These evaluator-side questions should not be confused with the business structure of the benchmark itself. In the current use case, the summary is already assessed against predefined information needs captured in the reference fields; QAFactEval adds a QA-based factuality check on top of that structure rather than defining the use case on its own.

Table 6.1 gives the simplified operational scoring sketches used in this thesis, where x denotes the source transcript and y the candidate summary. These are included for interpretive clarity rather than as full reimplementations of the original papers; Appendix A.2 provides the fuller breakdown and benchmark distinction.

Table 6.1: Operational definitions of the RQ2 semantic metrics.

Metric	Operational scoring sketch	Role in this thesis
UIEFID	$\text{UIEFID}(x, y) \approx \frac{1}{\lvert U(y)\rvert} \sum_{u \in U(y)} \text{Consistent}(u, x)$ where $U(y)$ are extracted information units from the summary.	Structured factual-consistency signal for distorted, unsupported, or missing details.
QAFactEval	$\text{QAFactEval}(x, y) \approx \frac{1}{\lvert Q(y)\rvert} \sum_{q \in Q(y)} \text{Agree}(a_x(q), a_y(q))$ where $Q(y)$ is a question set generated from summary claims.	QA-based factuality signal for testing whether concrete claims in the summary are supported by the transcript.

6.5 Baselines and Variants

The semantic calibration compared the recommended thresholds against the previous OpenLayer defaults of UIEFID = 0.55 and QAFactEval = 0.45. The final decision rule combined the calibrated semantic thresholds with hard integrity checks (null rows, duplicate rows, and maximum summary length) and a review band around borderline cases. A separate alignment study then compared OpenLayer built-in single metrics, pairwise combinations, and three- and four-metric combinations against semantic comparison labels in both the local 45-case benchmark and the 100-case public sample. OpenLayer was therefore evaluated in two ways: as an operational safeguard layer for the fixed gate, and as a comparative proxy-signal surface whose relevance had to be measured rather than assumed.

This baseline comparison served two purposes. First, it tested whether the inherited defaults were already adequate for the use case or whether they needed recalibration. Second, it prevented the final thresholds from appearing arbitrary: the recommended values were not only selected from the observed benchmark distribution, but also compared against the pre-existing operational defaults that a team might otherwise have accepted without validation.

6.6 Calibration Target

Final human review produced 45 adjudicated benchmark rows: 12 were labelled pass, 20 were labelled fail, and 13 remained `review`. For binary threshold fitting, the 32 explicit pass/fail cases were used as the calibration subset, while the 13 retained `review` cases were excluded from the binary fit and carried forward as an explicit ambiguity category in the operational analysis.

Earlier proxy labels remained useful as reviewer-prioritisation signals and historical context, but they are no longer the source of the reported thresholds. The final calibration is therefore more credible than the provisional rerun because it is grounded in explicit human labels, yet it is also more conservative because the reviewer did not force all ambiguous cases into binary closure.

6.7 Procedure

The RQ2 procedure had three linked phases. First, the fixed 45-case benchmark was scored with UIEFID and QAFactEval, and those stored per-case scores were then reused during the reviewed rerun. Manual review supplied 12 pass, 20 fail, and 13 review outcomes. The script swept candidate thresholds on the 32 binary cases, selected recommended cutoffs using the policy above, and combined those cutoffs with a review margin of 0.02 and hard integrity checks to produce a pass-review-fail triage rule on the full 45-case table. Second, the same 45-case benchmark was analysed at the OpenLayer-metric level by ranking built-in single metrics, pairwise combinations, and three- and four-metric combinations against the local acceptable versus not-acceptable pattern on the 32 binary reviewed cases. Third, a sampled 100-case public comparison set was scored with the same metric families, the same OpenLayer ranking analysis was repeated there, and local-versus-public comparison tables were generated to test transfer of metric relevance across benchmark surfaces.

The manual review and merge-back workflow is therefore not a downstream add-on in the final reported version of RQ2; it is the source of the reported calibration labels. What remains separate is the decision to preserve 13 cases as `review` rather than forcing them into pass or fail.

The ordering of these steps was intentional. The local benchmark had to be scored before manual review and threshold recalibration could be linked to stable case identifiers; reviewed labels had to be consolidated before thresholds could be recalibrated; and the decision rule had to exist before either the OpenLayer cross-check or the benchmark-comparison analysis could be interpreted correctly. Reversing this order would have allowed the current platform implementation or public benchmark behaviour to dictate the semantic definition of quality, which would have weakened the academic claim. By keeping the order fixed, the thesis ensures that platform limitations and public-transfer effects are treated as findings about proxy relevance and enforcement, not as definitions of quality.

The resulting design deliberately separates three layers. The offline semantic gate is the authoritative quality gate for RQ2. The OpenLayer alignment analyses study which built-in signals are good or bad proxies for that gate under different benchmark surfaces. OpenLayer itself provides only the platform-side operational checks that proved reliable enough in the benchmark run: null-row count, duplicate-row count, maximum summary length, and a row-count smoke-test floor. This separation became necessary because platform row-count behaviour remained unreliable across the available logs, and because metrics that look highly discriminative on the public comparison surface do not necessarily behave the same way on the local enterprise benchmark.

The more important limitation, however, is representational rather than purely operational. The offline 45-case calibration table contains structured reference information about the business outcomes that the summary must preserve, whereas the current OpenLayer export reduces each case to a transcript, one reference summary, and one generated summary. Because of that schema collapse, OpenLayer can currently reproduce only a narrower summary-to-summary check and not the full structured calibration target that underlies the offline pass-review-fail rule. A direct repair path is to change the use-case adapter instead of weakening the gate: preserve the structured reference fields in the canonical export, either as dedicated columns or as a versioned reference bundle that custom metrics can parse, and then point the OpenLayer profile at that richer target. Under that design, the same 45 case identifiers would be evaluated against semantically comparable evidence on both sides, while public datasets would remain comparison surfaces rather than substitutes for the local benchmark.

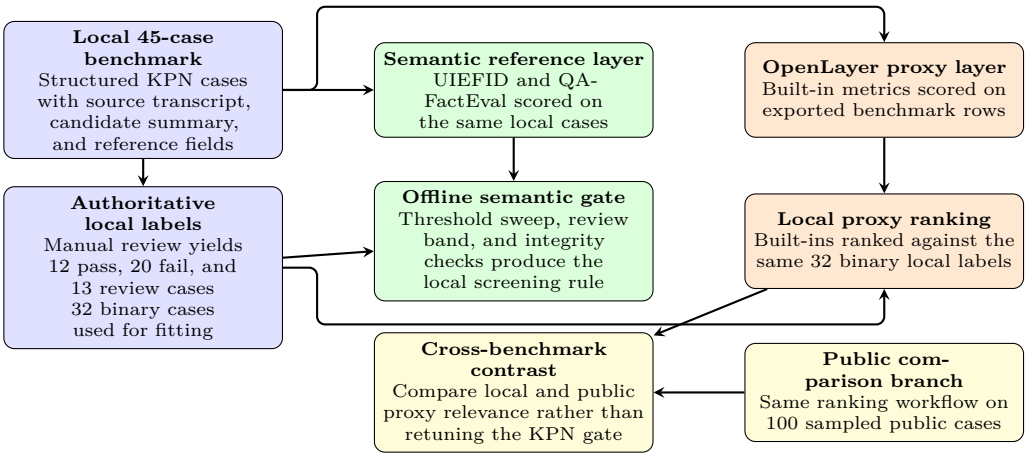

Figure 6.2: RQ2 procedure role separation: the reviewed local benchmark provides the authoritative labels, UIEFID and QAFactEval define the offline semantic gate, and OpenLayer built-ins are evaluated separately as proxy signals on local and public benchmark surfaces.

Chapter 7

Results

7.1 Primary Outcomes

Results are organised around the two research questions. RQ1 is answered by the implemented supervision architecture and its reusable workflow components. RQ2 is answered through two linked empirical outputs: a reviewed-label recalibration on the 45-case KPN call-summary benchmark, in which 32 binary human labels are used for threshold fitting while 13 retained `review` labels remain outside the binary fit, and a comparative OpenLayer-alignment study that ranks which built-in single metrics and metric combinations best approximate semantic acceptability on both the local benchmark and a separate 100-case public sample. Because the KPN benchmark remains the threshold-setting source of truth, the section first summarises the final reviewed calibration setup, then reports the local and public alignment analyses, and finally relates the reusable process outputs and operational cross-check back to the broader architectural contribution.

Table 7.1: RQ2 local benchmark and calibration setup.

Item	Result
Benchmark cases	45
Final human-reviewed cases	45
Binary calibration cases	32
Retained review cases	13
Binary human pass labels	12
Binary human fail labels	20
Semantic metrics calibrated	UIEFID, QAFactEval

Table 7.1 shows that the final reported calibration was performed on a fixed dataset and with explicit human-reviewed labels rather than with hidden manual tuning. This matters because the earlier proxy-based rerun is now only a provisional predecessor. The reported cutoffs come from the reviewed rerun, while the separate public comparison branch remains transfer evidence rather than a source of threshold fitting.

7.2 RQ1 Architecture Outcome

RQ1 is answered by the implemented supervision workflow described across Chapters 4–6. The key result is not one benchmark score, but the successful decomposition of supervision into reusable layers that can be applied across different application surfaces. Table 7.2 summarises the current architecture components and the concrete project evidence supporting each one.

Table 7.2: Implemented RQ1 architecture components and current evidence.

Component	Case-agnostic role	Current project evidence
Source adapters	Accept evaluation records from heterogeneous application surfaces	Code-based template repository and documented Dataiku summary-evaluation workflow
Canonical evaluation schema	Keep benchmark execution independent of the originating system	Shared fields such as input, context, ground truth, generated output, and traceability metadata
Versioned benchmark configuration	Make tests, mappings, thresholds, and runtime metadata reproducible	Repo-managed OpenLayer profiles, rendered configs, and versioned run metadata
Execution paths	Support repeatable local and CI/CD benchmarking	Local scripts, OpenLayer CLI flow, and GitHub Actions workflow
Result exports	Return audit-ready evidence to the surrounding delivery process	History CSV tables, logs, benchmark outputs, failure diagnostics, and review artefacts
Packaging path	Support future onboarding without changing the architecture	Standard minimum-benchmark process note, reusable template, and privacy-safe bundle; broader generic packaging still future work

The answer to RQ1 is that a scalable supervision pipeline with case-agnostic design goals requires five core architectural components:

1. source adapters that accept evaluation records from heterogeneous application surfaces;
2. a canonical evaluation schema that normalises those records into shared benchmark fields;
3. versioned benchmark configuration that freezes mappings, thresholds, and evaluator settings;
4. repeatable execution paths that run the same benchmark logic locally, in CI/CD, or through controlled platform flows; and
5. audit-ready exports that return scores, diagnostics, logs, and review artefacts to the surrounding delivery process.

In the current implementation, these five components are instantiated through a code-based OpenLayer template repository and a documented Dataiku workflow. For the KPN call-summary use case, this means the deliverable already exists in two concrete forms: a template repository for code-based projects and a documented Dataiku summary-evaluation workflow for interactive projects. A broader packaging surface already exists through the standard minimum-benchmark process note, the reusable template, and the privacy-safe bundle, but a fully generic cross-use-case template repository remains future work.

These results show that the reusable core is not OpenLayer or Dataiku themselves, but the interface pattern that allows different application surfaces to flow through the same benchmark and reporting logic. In the company context, OpenLayer serves as the benchmark engine and Dataiku as one interactive-platform adapter. The same architecture can also supervise code-based applications that generate evaluation records directly from a repository or service workflow.

The interactive-platform path also received a limited user-oriented validation. One researcher used the Dataiku workflow for a summary use case to compare and contrast run outputs across prepared benchmark runs. The screenshots shown earlier in Chapter 4 were captured from that use and indicate that the comparison surface was usable for inspecting model-version differences within the intended workflow context.

7.3 RQ2 Threshold Calibration Outcomes

The RQ2 findings should be read as the outcome of a staged development process rather than as isolated threshold numbers. Literature justified the metric families, the local benchmark supplied the calibration target, an earlier proxy-based rerun made the first gate explicit, and the final reviewed rerun then reused the stored per-case semantic scores and recalibrated thresholds on the 32 binary human-labelled cases while preserving 13 cases as `review`. The quantitative results below therefore describe the final human-reviewed outcome rather than the earlier provisional summary.

Figure 7.1 summarises the main internal path from the fixed benchmark to the final reviewed rerun. It is intended to make the dependency structure explicit: the final thresholds come from the manually reviewed label set, the retained `review` cases remain visible as an explicit ambiguity category, and the calibrated outputs are surfaced back as benchmark evidence for comparison and human inspection.

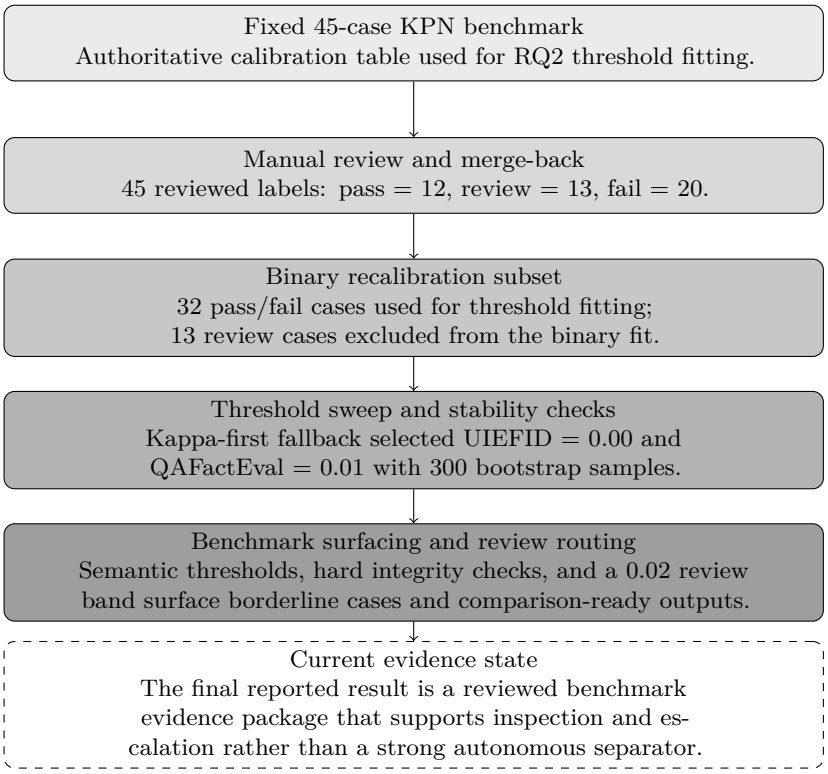

Figure 7.1: RQ2 evidence flow from manual review to reviewed benchmark surfacing.

For RQ2, the central quantitative finding is no longer a sharp automatic separator but a conservative reviewed-label screening configuration that mainly surfaces ambiguous benchmark behaviour for inspection. Table 7.3 reports the selected thresholds together with the binary-fit diagnostics on the 32 reviewed pass/fail cases. No threshold met the strict recall and good-case fail constraints simultaneously, so both recommendations came from the documented kappa-first fallback rule rather than from hidden manual selection.

Table 7.3: Reviewed-label RQ2 semantic thresholds and binary-fit diagnostics.

Metric	Recommended	Selection status	Recall_{bad}	Good fail rate	BA	κ
UIEFID	0.00	fallback best κ	0.0000	0.0000	0.5000	0.0000
QAFactEval	0.01	fallback best κ	0.8000	0.5833	0.6083	0.2281

The reviewed-label threshold sweep reveals why the final configuration becomes conservative. For UIEFID, the kappa-maximising fallback is already at 0.00, with recall on bad cases of 0.0 and balanced accuracy of 0.5, which means that the metric provides no useful binary separation on the reviewed subset. This should not be read as evidence that 0.00 is an inherently meaningful quality boundary; it means that the reviewed rerun did not justify any stricter standalone UIEFID cutoff on the binary subset. For QAFactEval, threshold 0.01 reaches recall on bad cases of 0.8, but only with a good-case fail rate of 0.5833. In other words, QAFactEval still retains some binary separation because it distinguishes a substantial share of bad cases from good ones, but not cleanly enough to support a low-escalation gate. The reviewed rerun therefore does not identify a sharp binary boundary; instead, it shows that once human reviewers preserve a substantial `review` category, the

calibrated benchmark surface is better suited to exposing borderline behaviour for inspection than to automatically separating clean pass and fail cases.

7.4 RQ2 Stability and Review-Oriented Screening

Bootstrap resampling no longer supports a strong claim of tight threshold stability. UIEFID is centred at 0.00 with a wide upper tail to 0.84, while QAFactEval has a median of 0.01 but a very wide upper tail to 4.6384. These distributions indicate that threshold selection is unstable on the small binary reviewed subset and reinforce the conservative interpretation of the final gate.

Table 7.4: Bootstrap stability summary for the selected RQ2 thresholds.

Metric	Recommended	Mean	Median	$p05$	$p95$
UIEFID	0.00	0.169935	0.00	0.00	0.84
QAFactEval	0.01	0.686123	0.01	0.01	4.6384

Applied to all 45 cases, the calibrated configuration combined the semantic thresholds, a review margin of 0.02, and hard integrity checks into a heavily review-oriented benchmark screen with only a small number of clean automatic passes and no automatic fail outcome. Most cases remained threshold-adjacent, and many passed exactly one semantic metric, which is the clearest sign that the reviewed rerun materially changed the earlier story: the final benchmark surface is informative as an escalation and comparison mechanism, but not as a strong autonomous pass/fail separator.

7.5 RQ2 Local OpenLayer Alignment Outcomes

Threshold fitting is only one part of the RQ2 result. The current repository also evaluates which OpenLayer-visible signals best track the same local acceptable-versus-not-acceptable pattern in the 45-case benchmark. This matters because OpenLayer is not only the current benchmark engine; it is also the main candidate operational surface where enterprise teams would like proxy signals to be meaningful without re-implementing the full offline semantic workflow. The built-in metric names reported below follow the official OpenLayer tests and aggregate-metrics documentation (OpenLayer 2026a,b).

Table 7.5 compares the strongest local and public alignment signals at a glance. Table 7.6 isolates the locally strongest built-in OpenLayer signals for the KPN call-summary use case. Appendix A summarises the final reviewed calibration outputs, and Appendix B reproduces the full local and public ranking tables.

Table 7.5: Compact cross-benchmark comparison of the strongest RQ2 alignment signals.

Surface	Best built-in single	Best built-in pair	Best built-in 3-metric combo	Reference-family example
Local 45-case	answerRelevancy / answerCorrectness (tie) $\kappa = 0.2941$, BA = 0.6667	answerRelevancy + faithfulness (tie with answerCorrectness + faithfulness) $\kappa = 0.4098$, BA = 0.7083	answerRelevancy + answerCorrectness + contextRecall $\kappa = 0.3231$, BA = 0.6750	proxy QAFactEval $\kappa = 0.9344$, BA = 0.9750
Public 100-case	answerCorrectness / meanSemanticSimilarity (tie) $\kappa = 1.0000$, BA = 1.0000	answerCorrectness + meanBleu2 (one of several ties) $\kappa = 1.0000$, BA = 1.0000	meanSemanticSimilarity + answerCorrectness + meanEditDistance $\kappa = 0.9356$, BA = 0.9688	paper UIEFID $\kappa = 0.5846$, BA = 0.7941

The reviewed-label local alignment study shows two things at once. First, the semantic reference-family scores remain the defining signals of the local benchmark logic: the local QAFactEval reference score still tracks the reviewed labels far more strongly than any OpenLayer built-in. Second, even after refreshing the rankings against human labels, no OpenLayer built-in signal becomes a decisive local proxy. The best built-in single metrics, answerRelevancy and answerCorrectness, reach only $\kappa = 0.2941$; the best built-in pair reaches $\kappa = 0.4098$; and the best built-in three-metric combination reaches $\kappa = 0.3231$. In practical terms, OpenLayer built-ins can assist interpretation, but they do not replace the local semantic layer.

Table 7.6: Most relevant built-in OpenLayer signals for the local 45-case KPN call-summary benchmark.

Signal class	Local OpenLayer highlight	κ	Bal. acc.
Single metric	answerRelevancy / answerCorrectness (tie)	0.2941	0.6667
Pairwise combination	answerRelevancy + faithfulness (tie with answerCorrectness + faithfulness)	0.4098	0.7083
Three-metric combination	answerRelevancy + answerCorrectness + contextRecall	0.3231	0.6750

This built-in-only local profile is also substantively plausible for the KPN call-summary use case. AnswerRelevancy and answerCorrectness are the strongest single built-ins because acceptable summaries must stay focused on the customer interaction and reflect the source content without materially misstating what happened. The strongest local pairs add faithfulness, which is consistent with a benchmark in which unsupported details, distorted claims, and source-summary mismatches matter more than stylistic fluency alone. This explanation should still be read cautiously: the built-ins capture some task-relevant aspects of local acceptability, but they remain only partial proxies for the fuller benchmark-specific judgement preserved by the local semantic reference layer and human review.

7.6 RQ2 Comparative Public OpenLayer Alignment Case

The second empirical RQ2 case repeats the same benchmark-to-platform alignment question on a public comparison surface. In this branch, a 34-row QAConv base sample (Wu et al. 2022) was expanded into a 100-case comparison surface, scored with the same semantic metric families using the project's paper-style real backends, and then analysed through the same OpenLayer single-metric, pairwise, and multi-metric ranking workflow.

This public case behaved very differently from the local 45-case benchmark, as Table 7.5 already suggests. Several OpenLayer built-ins, especially meanSemanticSimilarity and answerCorrectness, become extremely strong separators on the public case, and even the best built-in pairwise results reach perfect separation on the sampled surface. The best three-metric public combination also remains highly discriminative.

The public branch therefore does not weaken the local result; it clarifies it. On the sampled public surface, several OpenLayer built-ins look excellent. On the local enterprise benchmark, the same platform-visible signals are far weaker. This is exactly the kind of transfer gap that a deployment-oriented supervision workflow has to make visible rather than hide behind one global metric choice.

7.7 RQ2 Cross-Benchmark Metric-Relevance Contrast

The repository now contains direct local-versus-public comparison tables that quantify how strongly metric relevance shifts across benchmark surfaces. Table 7.7 shows the four highest-local-kappa single-metric contrasts from the local ranking, including the two reference-family signals and the two strongest OpenLayer built-ins. This keeps the subsection anchored on the local benchmark while still retaining the public values needed for cross-benchmark contrast.

Table 7.7: Top four local single-metric local-versus-public contrasts by local Cohen's kappa.

Metric	Local AUC	Pub. AUC	Local kappa	Pub. kappa
QAFactEval family (proxy local / paper public)	0.9500	0.6618	0.9344	0.3202
UIEFID family (proxy local / paper public)	0.9438	0.7941	0.8667	0.5846
answerRelevancy	0.6563	0.5207	0.2941	0.1073
answerCorrectness	0.6458	1.0000	0.2941	1.0000

Table 7.7 makes two points visible immediately. First, the highest local single-metric kappas belong to the semantic reference-family scores rather than to the OpenLayer built-ins. Second, the public values do not preserve that same ordering cleanly: answerCorrectness is only a modest local proxy but becomes a perfect public separator, whereas the QAFactEval family is the strongest local signal but much weaker on the public branch.

Figure 7.2 mirrors the same four metrics as direct local-to-public links at the level of Cohen's kappa. Read together, the table and figure show that the two reference-family metrics dominate the local ranking, while the public branch especially rewards answerCorrectness and, to a lesser extent, UIEFID.

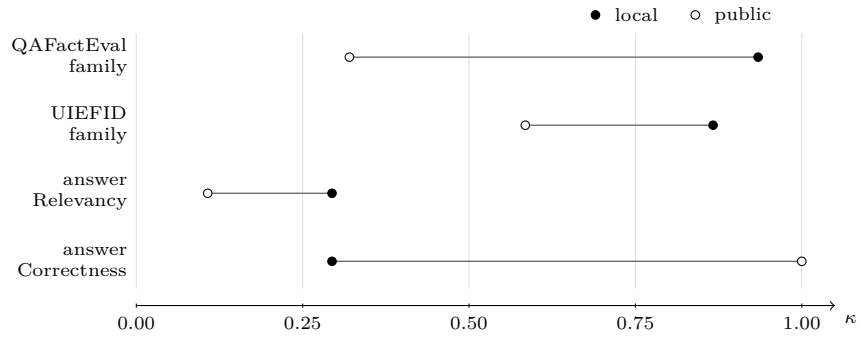

Figure 7.2: Local-versus-public Cohen's kappa gaps for the four highest-local-kappa single-metric signals. Each line connects the local and public kappa values for the same signal family or OpenLayer built-in metric.

Taken together, the table and figure show that the metric contrast is not just a matter of small numeric drift. On the reviewed 45-case enterprise benchmark, the strongest local single metrics overall are the semantic reference families, while the strongest local OpenLayer built-ins are answerRelevancy and answerCorrectness at only 0.2941 kappa. By contrast, answerCorrectness becomes a perfect public separator, whereas the QAFactEval and UIEFID families lose substantial relative strength on the public branch. The same instability appears in the pairwise and higher-order comparison outputs: the strongest local built-in pairs reach only roughly the 0.30–0.41 kappa range, whereas corresponding public pairs rise to values near 1.0, and the best local built-in three-metric combination reaches only 0.3231 versus 0.9356 on the public branch. The OpenLayer metric labels referenced in the built-in rows follow the platform's official aggregate-metrics documentation (OpenLayer 2026a).

This gap is consistent with the two benchmarks asking for different things: the public cases and the KPN 45-case enterprise benchmark encode different quality requirements, critical information fields, and tolerance for omissions, so the same built-in metrics do not retain the same relevance across both settings.

This comparison should be interpreted carefully. The public branch uses paper-style UIEFID and QAFactEval backends, whereas the local 45-case comparison files use calibrated proxy variants for the same metric families. Even with that caveat, the ranking shift is too large to ignore. The safest interpretation is therefore not that one global OpenLayer metric has been found, but that metric relevance is strongly benchmark-dependent and must be established locally.

7.8 RQ2 Operational Cross-check

The OpenLayer cross-check provided an operational validation result rather than a replacement for the semantic gate. Across the available logs, two row-count issues were observed: one earlier completed log reflected misleading row-count metadata caused by multiline CSV handling, while the successful benchmark-profile run returned 21 test items, of which 20 passed and 1 failed. The single failing item in that successful run was still the exact row-count check: OpenLayer reported a subpopulation row count of 10 even though the local benchmark slice contained 45 rows. This confirms that platform-side row-count enforcement is not yet reliable enough to serve as the primary quality gate for this use case.

The resulting minimum-requirements design has two layers that form the current minimum viable test set. The authoritative semantic gate consists of two calibrated semantic factuality tests (UIEFID ≥ 0.00 and QAFactEval ≥ 0.01), three hard integrity checks (null rows, duplicate rows, maximum summary length), and explicit review routing for borderline cases. This set is minimal in the practical sense used by the thesis: each element covers a distinct failure mode that should be controlled in a pre-release screening decision. In the final reviewed rerun, however, this authoritative gate operates mainly as a conservative escalation policy rather than a strong autonomous pass/fail filter. The platform gate serves as a narrower second layer, providing a row-count smoke-test floor. As demonstrated by the subpopulation-row-count failure, the platform result remains valuable as an operational safeguard, but it is not treated as part of the authoritative semantic minimum because current platform behaviour was not reliable enough for full enforcement.

In other words, RQ2 is answered in three linked steps: first by deriving the minimum gate on the fixed KPN benchmark, second by testing which OpenLayer-visible signals best approximate semantic acceptability on both the local benchmark and a separate public comparison surface, and third by showing how that gate is handled operationally through review routing and an OpenLayer cross-check.

Overall, these results answer RQ2 with a bounded but concrete outcome: the thesis now provides a human-reviewed benchmark-derived control rule for KPN call summaries, a local benchmark-to-OpenLayer alignment analysis refreshed against reviewed labels, a 100-case public comparison that exposes strong benchmark dependence in OpenLayer metric relevance, a reusable process package that explains how such a gate should be produced, explicit recalibration logic, stability diagnostics, a documented operational cross-check, and a traceable review-packet workflow. What it still does not provide is a sharp autonomous separator, a pure backend-controlled local-versus-public comparison, or proof that the KPN thresholds transfer unchanged to other use cases, because 13 cases remain in the explicit `review` category and the final rerun is intentionally conservative.

Chapter 8

Discussion

8.1 Interpretation of Findings

The current findings should be interpreted as a linked answer to RQ1 and RQ2. RQ1 is answered at the level of artefact design: the thesis defines and documents a supervision architecture intended to be case agnostic, built around adapters, a canonical evaluation schema, versioned benchmark configuration, execution, and audit-ready outputs. RQ2 provides the strongest completed empirical evidence, where the thesis moves from a generic idea of minimum viable testing to a documented benchmark-based comparison configuration for the KPN call-summary use case, a final reviewed-label recalibration on a 32-case binary subset with 13 retained `review` outcomes, a benchmark-to-platform alignment study on the same 45-case surface, a comparative 100-case public alignment study, and a reusable minimum-benchmark package. The result therefore supports the research objective at the level of bounded evaluation-design evidence, even though only RQ2 is calibrated on one concrete use case and neither question yet constitutes exhaustive validation across all intended application classes.

The interpretation must preserve one central distinction. The literature supports the metric families and the evaluation design: QAFactEval as a dedicated factual-consistency axis, UIEFID as a structured inconsistency detector, and conversation-summary work as justification for task-specific evaluation beyond lexical overlap (Fabbri et al. 2022; Gupta et al. 2025; Li and Yu 2025; Tang et al. 2024; Wan et al. 2025). The exact numeric cutoffs, however, are not literature-derived. They are benchmark-derived thresholds fitted on the local reviewed benchmark, specifically on the 32 binary human-labelled cases while 13 cases were retained as `review`. This is precisely why the resulting configuration should be presented as a human-reviewed, use-case-specific comparison setup rather than as a universal benchmark standard.

A second distinction concerns benchmark transfer. The 45-case enterprise benchmark and the 100-case public comparison surface do not reward the same OpenLayer proxies in the same way. On the public sample, metrics such as meanSemanticSimilarity and answerCorrectness nearly perfectly separate acceptable from not-acceptable cases, whereas on the local enterprise benchmark their kappa values remain low and no built-in combination becomes a decisive local proxy. This shows that public success of a platform metric does not license direct reuse as a local enterprise quality gate.

8.2 Evaluator Fit for the KPN Use Case

The local ranking results are also interpretable in use-case terms rather than only as abstract metric comparisons. The KPN benchmark is a structured enterprise call-summary task in which a summary is acceptable only if it preserves operationally relevant facts such as the call reason, the agreed follow-up, the outcome, and other issue-specific fields. In that setting, the main failure mode is not unusual wording by itself, but omission, distortion, or unsupported invention of business-critical content. A metric family that is more directly tied to source-grounded factual support should therefore be expected to fit the local benchmark better than a generic platform-visible proxy.

This is why the reference-family signals are stronger local evaluators in the present study. QAFactEval is well matched to the benchmark because it tests whether concrete claims expressed

in the summary can be supported by the source transcript through structured question-answer agreement. UIEFID is similarly well matched because it operates at the level of structured information-unit consistency and is sensitive to summaries that preserve the overall topic while still distorting specific facts or actions (Fabbri et al. 2022; Li and Yu 2025). Because the benchmark labels are anchored in predefined business questions and structured reference fields, the reviewed subset is likely closer to question-level factual support than to a more generic inconsistency notion; this helps explain why QAFactEval supports a cleaner standalone cutoff in the final rerun, while UIEFID remains locally relevant yet does not yield a strong standalone threshold. Table 7.7 and Figure 7.2 are consistent with that expectation: the QAFactEval and UIEFID families occupy the top local single-metric positions, whereas the strongest local OpenLayer built-ins, answerRelevancy and answerCorrectness, remain far lower.

This should still be interpreted as local evaluator fit rather than as proof of universal superiority. The same results chapter shows that metric relevance shifts on the public comparison surface, and the final configuration still requires explicit review routing and additional integrity checks. The defensible conclusion is therefore narrower: for this industry-specific call-summary benchmark, source-grounded semantic factuality evaluators provide the most appropriate local reference layer, while OpenLayer built-ins remain secondary operational approximations.

8.3 Practical Implications

For organisations evaluating GenAI systems, the practical contribution of RQ1 is that supervision no longer needs to be redesigned from scratch for each application surface. A code-based service and an interactive platform workflow can share the same evaluation core if both can map their records into the canonical schema and consume the exported result artefacts. In the company context, OpenLayer and Dataiku are therefore examples of integrations, not the full definition of the pipeline.

RQ2 adds a bounded benchmark-comparison pattern instead of only a metric discussion. In the final reviewed rerun, the calibrated semantic thresholds, the review band, and the narrower platform checks form a deliberately conservative configuration: only clearly unambiguous cases pass automatically, while most cases are routed to human review rather than being auto-failed.

In the summary use case, this is especially important because concise and persuasive summaries can still miss crucial action items or introduce unsupported details. The calibrated thresholds and review band respond directly to that risk. The final reviewed rerun shows that, once human reviewers preserve a substantial ambiguity category, conservative review routing is preferable to brittle automatic closure.

The comparative alignment branch adds a second practical implication. Teams should not assume that OpenLayer built-ins which look excellent on public benchmarks will behave the same way on enterprise data. In the present study, answerCorrectness and meanSemanticSimilarity are almost perfect separators on the 100-case public surface but weak local proxies on the 45-case KPN benchmark. For local evaluation practice, that means platform metrics must be validated against the local benchmark rather than promoted from public benchmark success into benchmark-facing conclusions by default.

More generally, benchmark design should follow the operational risk of the use case: tasks with little tolerance for omitted, unsupported, or distorted information need stricter local benchmarks anchored in explicit business-critical reference fields, whereas less critical tasks can justify lighter benchmark structures and wider acceptance margins.

The review packet and merge artefacts matter because they turned human adjudication into a governed calibration input rather than leaving it as an informal afterthought. Borderline cases can be isolated, inspected, and written back into governed artefacts with explicit provenance, and the final label set can preserve `review` as a genuine category instead of forcing uncertain cases into artificial binary closure.

From a governance perspective, the practical value of the pipeline is that it transforms evaluation from an opaque model-quality check into an auditable evaluation process. The calibrated configuration is documented, the selection policy is explicit, and the platform limitations are recorded rather than hidden. This directly supports regulatory expectations around traceability, oversight, and demonstrable accountability in EU enterprise contexts (Parliament and European Union 2016,

2022, 2024). The architecture rationale is therefore not only technical efficiency, but also reduction of compliance and assurance risk during benchmark-based change assessment.

The process packaging also has a practical implication for future onboarding. A standard minimum-benchmark process note, a reusable template, and a privacy-safe bundle lower the documentation and audit burden for the next use case, even though they do not remove the need for fresh calibration on that next benchmark.

8.4 Architecture Trade-offs

The main advantages of the architecture are scalability, repeatability, auditability, and model agnosticism. Here, scalability should be read mainly at the architecture and workflow level: the same supervision core can be reused across adapters, run locally or in CI/CD-style workflows, and extended to new use cases without redesigning the evaluation logic each time. It allows teams to evaluate many scenarios more often than manual review would allow, while still preserving evidence for review and later inspection.

The trade-offs are cost, latency, calibration effort, comparison complexity, and residual dependence on human judgement. On the RQ1 side, the architecture gains reuse by enforcing a canonical schema and versioned configuration, but this also creates onboarding work for each new adapter. On the RQ2 side, the most visible trade-off is between semantic fidelity and operational enforceability. The offline semantic reference layer is stronger and better aligned with the use case, but it currently cannot be delegated completely to the platform. OpenLayer adds valuable operational safeguards, yet its row-count behaviour in the benchmark run and its unstable cross-benchmark proxy relevance show that platform convenience cannot substitute for the benchmark-derived semantic layer. The newer process packaging reduces repetition for future use cases, but it does not remove the need for local benchmark preparation, local calibration effort, or recalibration. Human review therefore remains necessary for borderline or high-risk cases, and the pipeline should be treated as structured review support rather than a complete replacement for expert judgement.

8.5 Limitations

A key limitation on the RQ1 side is that the architectural claim is supported by one working company implementation, namely a code-based template repository plus documented adapter flows, not yet by a completed cross-organisational generalisation of that template repository or by comparative validation across many application categories. The thesis therefore claims reusable design logic, not definitive proof that every future use case will integrate at the same cost.

The user-oriented validation of the interactive Dataiku path is also limited in scope. It consisted of one researcher using the workflow to compare and contrast summary-evaluation runs, which is useful as evidence that the interface can be operated in context, but should not be interpreted as a formal usability study or a broad user-acceptance evaluation.

A key limitation on the RQ2 side is no longer the absence of human review, but the fact that the final human review preserves 13 cases as explicit `review` outcomes. That improves label honesty, yet it leaves only 32 binary cases for threshold fitting and prevents the benchmark from behaving like a clean binary calibration set.

Further limitations follow from sample size, implementation surface, and scope. The 45-case benchmark size is intentionally modest: evaluating sensitive enterprise data requires costly expert annotation and the manual establishment of ground-truth structured reference fields, so the dataset was designed for density and quality rather than raw scale. Even so, both selected thresholds came from the fallback-best-cohen-kappa rule rather than from satisfying the strict recall target, UIEFID contributed almost no binary separation in the reviewed rerun, and the resulting configuration was calibrated on one concrete enterprise summary task. This limits how strongly the results can be generalised beyond the evaluated use case. Accordingly, the current configuration should be read as a conservative review-oriented comparison setup, not as proof that future updates will remain behaviourally consistent under all operating conditions. Platform behaviour is an additional limitation: the successful OpenLayer run provided a valuable cross-check, but the row-count issue showed that the platform cannot yet reflect the full semantic benchmark on its own. A related

limitation is that the thesis does not yet include a dedicated throughput, latency, or resource-usage benchmark under larger workload sizes. The scalability claim is therefore architectural and workflow-oriented rather than a full runtime-scaling study.

The benchmark-comparison branch has its own limitations. The 100-case public comparison case is sampled rather than exhaustive, and metric coverage is incomplete for some OpenLayer signals. More importantly, the public branch uses paper-style real UIEFID and QAFactEval backends, whereas the local 45-case comparison files use calibrated proxy variants for the same metric families. The public-versus-local contrast should therefore be interpreted as evidence of benchmark and implementation-surface transfer instability rather than as a pure ceteris paribus comparison. The broader literature surface remains wider than the executed artefacts in the repository: for example, AUTOSUMM remains useful contextual literature, but it was not operationalised as an evaluation benchmark in the present study.

8.6 Lessons Learned

A central lesson is that evaluator governance must be treated as a product component, not only as an analysis step. Another lesson is that benchmark calibration and platform validation should be kept analytically separate: one establishes the candidate benchmark configuration, while the other determines which parts of that configuration can actually be reflected in the serving environment. A third RQ2 lesson is that public benchmark wins do not identify a universal OpenLayer proxy. Metric relevance has to be established against the local benchmark that actually represents local evaluation risk. A further RQ1 lesson is that case agnosticism depends less on abstract claims than on explicit interface contracts: canonical evaluation fields, versioned benchmark profiles, and exportable evidence. For RQ2 specifically, the reusable part is the calibration and alignment process, not the numeric thresholds themselves. Finally, both the calibration workflow and the review-packet workflow should be treated as reusable infrastructure, because the same artefacts can be rerun later with updated human adjudications or alternative handling of retained `review` cases without rescoring every case.

Chapter 9

Conclusion and Future Work

9.1 Conclusion

The main research question can be answered as follows: an automated supervision pipeline for non-deterministic enterprise GenAI applications can be designed as a reusable evaluation architecture intended to be case agnostic and model agnostic, and evaluated through a benchmark-calibrated workflow that combines semantic scoring, hard integrity checks, explicit review routing, and audit-ready evidence export.

This thesis now answers two sub-questions. RQ1 is answered by a supervision architecture designed to be case agnostic, built around source adapters, a canonical evaluation schema, versioned benchmark configuration, execution paths, and audit-ready outputs. In the current company implementation, this architecture is instantiated through a code-based template repository around OpenLayer and documented Dataiku workflows, while the broader design target remains support for both code-based and interactive GenAI applications.

RQ2 is answered by a bounded empirical result for the KPN call-summary use case. The answer combines three linked elements: calibration of a minimum review-oriented comparison configuration on the fixed 45-case KPN benchmark after manual review, a comparative benchmark-to-platform alignment study on the 45-case benchmark and a separate 100-case public sample, and a practical comparison path built around review routing and platform cross-checking.

On the fixed 45-case benchmark derived from sensitive internal data, manual review produced 12 pass, 13 review, and 20 fail labels. The final recalibration fit thresholds on the 32 binary pass/fail cases, selected UIEFID ≥ 0.00 and QAFactEval ≥ 0.01 via the fallback-best-cohen-kappa rule, and combined them with a 0.02 review band into a heavily review-oriented comparison configuration rather than a balanced automatic triage rule. Concretely, the resulting minimum viable test set still consists of two semantic factuality tests, three hard integrity checks, and explicit review routing for borderline cases, but in its final reviewed form it functions mainly as a conservative review-routing mechanism and benchmark-inspection aid.

The comparative alignment study adds an important transfer result. On the 100-case public comparison surface, OpenLayer built-ins such as meanSemanticSimilarity and answerCorrectness separate acceptable from not-acceptable cases almost perfectly, and the best three-metric public combination reaches kappa 0.9356. On the local 45-case enterprise benchmark, by contrast, the strongest built-in single metrics remain below 0.30, the best built-in pair reaches 0.4098, and the best built-in three-metric combination reaches 0.3231. This shows that OpenLayer metric relevance is strongly benchmark-dependent and should be established locally rather than inferred from public benchmark success. More broadly, the benchmark itself must be shaped by the operational demands of the specific use case, because each domain brings different critical fields, failure risks, and acceptance criteria.

Beyond the local calibration itself, the current RQ2 output also includes a reusable minimum-benchmark process note, a reusable template, and a privacy-safe audit bundle, which turn the calibration into a rerunnable method package rather than a one-off benchmark exercise. The review-packet and merge-back workflow fed directly into the final rerun and confirmed that the review-routing branch can be operated with stable provenance.

However, because 13 cases were deliberately retained as `review` and excluded from binary

threshold fitting, the current artefacts still do not behave like a clean binary calibration set. The benchmark-comparison branch also remains bounded by implementation-surface differences: the public comparison uses paper-style real UIEFID and QAFactEval backends, whereas the local 45-case comparison files use calibrated proxy variants for some of the local ranking outputs. The thesis therefore argues that minimum viable testing for a non-deterministic enterprise GenAI task can be derived empirically inside a reusable supervision architecture rather than asserted only through generic best practice. This configuration is intended as a benchmark-based comparison and review-routing mechanism for future model, prompt, or system updates on the fixed benchmark, even though the thesis does not yet claim exhaustive longitudinal validation across many update variants. The result is bounded rather than universal, but it is already strong enough to support structured assessment of update effects, review prioritisation, and local validation of platform-visible proxy metrics for the concrete use case studied.

9.2 Future Work

The most immediate RQ1 next step is not to create the first repository from scratch, but to generalise the current code-based template repository into a broader reusable template repository so new use cases can adopt the same canonical schema, benchmark-profile structure, and export logic with less setup effort. That packaging work no longer starts from zero: the current project already includes a standard minimum-benchmark process note, a reusable template, and a privacy-safe audit bundle. The next step is to package those artefacts beyond the present KPN summary use case while keeping the result platform agnostic, with adapters for repository-based, CI/CD, and interactive-platform workflows rather than hardwiring one company tool chain.

The most important RQ2 next step is to expand the human-reviewed benchmark and test alternative treatments of the retained `review` cases. The current reviewed rerun shows that preserving ambiguity is methodologically honest, but it also weakens binary threshold fitting and pushes the operational rule toward near-total escalation. Future work should therefore evaluate whether a larger reviewed sample, a ternary calibration design, or a confidence-aware decision policy can preserve the explicit review category without collapsing the automatic gate. Beyond that, future work should focus on recalibration on at least one additional enterprise use case, repeated-run stability analysis, broader cross-version validation, backend-parity reruns of the public-versus-local alignment study, and tighter integration of the public-comparison and review workflows into the main evaluation package. As models, prompts, and business processes evolve, the scenario set and evaluator configuration should remain living artefacts rather than fixed one-off benchmarks. An additional priority is governance evolution: maintaining a living control mapping between pipeline components and regulatory expectations (for example AI Act implementation guidance, GDPR accountability practices, and NIS2 operational requirements) as these frameworks and sector-specific interpretations evolve over time.

Appendix A

Project Artefacts

This appendix stores operational detail that supports reproducibility without overloading the main thesis. The values below reflect the current RQ2 benchmark package and the broader supervision design. Where the appendix reports reusable controls or rerun structures rather than measured outputs, that role is stated explicitly instead of implying unsupported numeric results.

A.1 RQ1 Canonical Evaluation Schema and Adapter Pattern

The current RQ1 architecture separates reusable supervision components from platform-specific adapters.

Source adapters Code-based applications, CI/CD workflows, or interactive platforms export records for evaluation.

Canonical fields The shared evaluation schema centres on `input`, `context`, `ground_truth`, `generated_summary`, and `output`, with optional traceability fields such as `conversation_id`, `model_name`, `dataset_version`, and timestamps.

Versioned benchmark configuration Profiles, thresholds, dataset mappings, and run metadata are stored in code so the same evaluation logic can be reused across use cases.

Execution paths The current project supports local execution, CI/CD execution, and documented Dataiku workflows that prepare the same canonical schema before benchmarking.

Exported artefacts The pipeline returns run histories, threshold tables, logs, failure diagnostics, and review artefacts so results can feed dashboards and release decisions.

A broader generic template repository remains future packaging work, but the current project already includes partial packaging surfaces in the form of a standard minimum-benchmark process note, a reusable benchmark template, and a privacy-safe RQ2 bundle. The Dataiku-oriented adapter implementation discussed in this thesis is available at https://github.com/enekoreto/scalable-oversight-dataiku-adapter.

A.2 RQ2 Metric Definitions and Scoring Sketches

The RQ2 calibration combines several artefact layers that should not be conflated.

Reusable process artefacts A standard minimum-benchmark process note, a reusable benchmark template, and a privacy-safe RQ2 bundle document the rerun and audit structure without exposing raw KPN benchmark content.

Local 45-case calibration benchmark The fixed internal benchmark slice from `MI_GT_final.xlsx` is the authoritative dataset used to derive thresholds and review logic.

Semantic metrics UIEFID and QAFactEval are the two summary-level factuality signals computed for each benchmark case.

OpenLayer benchmark-profile cross-check The OpenLayer run is a platform-side operational test used to determine which checks remain enforceable in-platform; it is not a third semantic metric.

Post-calibration review artefacts The 13 review-band cases were exported into a dedicated review packet and merged back into a consolidated workbook and final human-adjudication export. In the final rerun, those merged labels became the authoritative local calibration source. Because all inspected borderline cases remained `review`, the reviewed artefacts improve provenance and label honesty while preserving an explicit ambiguity category.

Auxiliary public evidence The QMSum and TofuEval runs reuse the already-fixed KPN thresholds on public summarisation data. They provide secondary triangulation only and do not recalibrate the KPN gate.

The metric sketches below are simplified operational summaries used to explain how the scores behave in this thesis. They are not substitutes for the full published methods.

UIEFID Let x denote the source transcript, y the candidate summary, and $U(y)$ the set of structured information units extracted from the summary. A simplified scoring sketch is

$$\mathrm{UIEFID}(x, y) \approx \frac{1}{|U(y)|} \sum_{u \in U(y)} \mathrm{Consistent}(u, x).$$

Here, $\mathrm{Consistent}(u, x) \in [0, 1]$ represents whether an extracted information unit from the summary is supported by the source after universal information extraction and LLM-assisted reasoning. In the thesis, higher values indicate stronger structured factual alignment and fewer summary-level inconsistencies.

QAFactEval Let $Q(y)$ be the question set generated from the claims expressed in the candidate summary, let $a_x(q)$ be the answer to question q from the source transcript, and let $a_y(q)$ be the answer from the candidate summary. A simplified scoring sketch is

$$\mathrm{QAFactEval}(x, y) \approx \frac{1}{|Q(y)|} \sum_{q \in Q(y)} \mathrm{Agree}(a_x(q), a_y(q)).$$

Here, $\mathrm{Agree}(a_x(q), a_y(q)) \in [0, 1]$ represents answer agreement between the source-grounded answer and the summary-grounded answer. In the thesis, higher values indicate that the claims made in the summary are better supported by the transcript.

After metric scoring, the thesis does not use either semantic score in isolation. The benchmark-derived thresholds are combined with hard integrity checks and explicit review routing, as detailed in Appendix A.4. This means the operative RQ2 gate is a composite screening rule rather than a single-metric benchmark.

A.3 RQ2 Artefact Roles and Boundaries

The current RQ2 artefacts have different roles and should not be mixed.

Calibration source `MI_GT_final.xlsx` is the authoritative KPN benchmark source used for threshold setting.

Threshold outputs The earlier provisional rerun outputs and the final reviewed-label threshold-selection directory store the threshold-selection outputs, bootstrap summaries, and decision-rule summaries reported in the thesis.

Review packet The review-packet workbook is a convenience subset for manual review of the 13 retained review-band cases and forms part of the final provenance chain; it is not a separate benchmark.

Merged review outputs The merged review workbook and related CSV and JSON summaries document provenance-preserving human review results and feed the final reviewed-label recalibration reported in the thesis.

Auxiliary public evidence The QMSum and TofuEval folders under `auxiliary_public_evidence/` are fixed-threshold external checks only. They do not retune the KPN gate.

Auxiliary human-agreement metric In auxiliary public runs, `human_agreement_rate` is the aggregate share of cases where the automated decision agrees with the available human signal used by the export workflow. In those runs, that human signal is proxy based (derived from dataset ground truth and other available human-authored reference fields) rather than explicit per-case adjudicated pass/fail labels.

Privacy-safe bundle The RQ2 bundle packages code, tests, configs, and privacy-safe derived artefacts for audit or transfer of method, while intentionally excluding sensitive benchmark content and auxiliary public evidence.

A.4 Final RQ2 Minimum Requirements and Reviewed Calibration Outputs

The final reviewed-label RQ2 gate for the KPN call-summary use case is:

Primary semantic thresholds UIEFID ≥ 0.00 and QAFactEval ≥ 0.01.

Hard integrity checks Null-row count ≤ 0, duplicate-row count ≤ 0, and generated-summary maximum character length ≤ 350.

Review routing Human review is required when either metric is within 0.02 of its threshold or when exactly one semantic metric passes.

Final human-review distribution On the 45-case benchmark, manual review produced 12 `pass` labels, 13 `review` labels, and 20 `fail` labels.

Binary calibration subset The threshold fit used the 32 explicit pass/fail cases, while the 13 retained `review` cases were excluded from the binary fit.

Final benchmark surface Applied to all 45 benchmark cases, the resulting configuration routed almost the entire benchmark into review, with only a small number of clean automatic passes and no automatic fail outcome.

Platform-safe checks The current OpenLayer operational layer is limited to null rows, duplicate rows, maximum summary length, and a row-count smoke-test floor, because exact row-count enforcement did not remain reliable in the successful benchmark run.

The appendix tables below collect the reviewed-label calibration outputs in one place, while Appendix B reproduces the full ranked local and public metric tables.

Table A.1: Final reviewed-label RQ2 semantic thresholds and binary-fit diagnostics.

Metric	Recommended	Selection status	Recall$_{bad}$	Good fail rate	BA	κ
UIEFID	0.00	fallback best κ	0.0000	0.0000	0.5000	0.0000
QAFactEval	0.01	fallback best κ	0.8000	0.5833	0.6083	0.2281

Table A.2: Bootstrap stability summary for the final reviewed-label thresholds.

Metric	Recommended	Mean	Median	$p05$	$p95$
UIEFID	0.00	0.169935	0.00	0.00	0.84
QAFactEval	0.01	0.686123	0.01	0.01	4.6384

Table A.3: Final reviewed-label calibration and decision-rule counts.

Quantity	Value
Reviewed benchmark cases	45
Final human pass labels	12
Final human review labels	13
Final human fail labels	20
Binary pass/fail cases used for calibration	32
Review cases excluded from binary fitting	13
Automatic pass cases under final rule	3
Review cases under final rule	42
Automatic fail cases under final rule	0
Threshold-adjacent cases	42
Exactly-one-metric-pass cases	31
Overlength cases	0

A.5 Disagreement Attribution Taxonomy

The disagreement taxonomy below assigns one primary cause label to each material mismatch between automated scores and human ratings.

Judge agreeableness bias The automated judge accepts a fluent or confident output despite weak evidence, missing facts, or unsupported claims.

Rubric mismatch The human expert and automated judge apply different interpretations of the scoring criteria.

Ambiguous scenario The prompt, source context, or expected behaviour leaves multiple reasonable interpretations.

Human inconsistency Human ratings differ because of fatigue, unclear instructions, or inconsistent application of the rubric.

Model-quality error The generated output contains a factual, safety, instruction-following, or usefulness error that one evaluator misses.

For each material disagreement, the report should store the scenario identifier, automated rationale, human rationale, cause label, and adjudication decision.

A.6 Governance Control Mapping

The governance mapping links regulatory or audit expectations to concrete pipeline controls.

Traceability Versioned scenarios, prompts, retrieved context, model settings, generated outputs, scores, and release decisions are retained for later inspection.

Human oversight Low-confidence, high-impact, or contradictory results are escalated to a human reviewer, with final override decisions logged.

Accountability Each release decision package records the evaluated version, scoring configuration, responsible reviewer or owner, and pass/fail rationale.

Operational risk management Failures are categorised by severity and type so they can support incident review, root-cause analysis, and future test-set updates.

A.7 Summary Evaluation Rubric

For the call-summary use case in this project, the following rubric dimensions serve as scoring anchors.

Factual faithfulness Claims in the summary are supported by the call transcript or source context.

Coverage The summary captures key decisions, issues, customer needs, and action items.

Relevance The summary excludes distracting or low-value details.

Actionability Follow-up tasks, owners, deadlines, or commitments are clear when present in the source.

Clarity The summary is concise, readable, and suitable for enterprise use.

Compliance sensitivity The summary avoids unsupported sensitive claims and respects domain policy constraints.

Appendix B

Full RQ2 OpenLayer Ranking Tables

This appendix reproduces the full ranked local and public OpenLayer alignment outputs underlying the compact RQ2 comparison table in the main text. All tables are ordered by Cohen's kappa in descending order. Single-metric tables report AUC, balanced accuracy, and kappa at the best threshold; pairwise and three-metric tables report coverage, balanced accuracy, and kappa for the corresponding combination rule.

B.1 Local 45-case benchmark rankings

These local rankings reproduce the final reviewed-label rerun. The ranking fit uses the 32 binary human-labelled cases from the 45-case benchmark, while the 13 retained `review` cases remain outside the binary ranking fit but are still part of the operational decision analysis reported in the main text. The full tables below therefore include both OpenLayer built-ins and the local proxy reference-family scores, whereas the compact comparison table in the results chapter isolates built-in-only highlights separately.

B.1.1 Single metrics

Table B.1: Full local reviewed-label 45-case single-metric ranking.

Rank	Metric	Source	Cov.	AUC	Bal. acc.	Kappa
1	proxy_qafacteval_score	Proxy	32	0.9500	0.9750	0.9344
2	proxy_uiefid_score	Proxy	32	0.9438	0.9333	0.8667
3	answerRelevancy	OpenLayer	32	0.6563	0.6667	0.2941
4	answerCorrectness	OpenLayer	32	0.6458	0.6667	0.2941
5	contextUtilization	OpenLayer	32	0.6000	0.6000	0.2308
6	meanEditDistance	OpenLayer	32	0.5375	0.6083	0.2281
7	meanSemanticSimilarity	OpenLayer	32	0.5292	0.6083	0.2281
8	contextRecall	OpenLayer	32	0.5958	0.6250	0.2239
9	hallucination	OpenLayer	29	0.5051	0.5707	0.1493
10	faithfulness	OpenLayer	32	0.5292	0.5583	0.1186
11	meanBleu4	OpenLayer	32	0.5042	0.5583	0.1186
12	meanBleu3	OpenLayer	32	0.5042	0.5583	0.1186
13	meanBleu2	OpenLayer	32	0.5042	0.5583	0.1186
14	meanBleu1	OpenLayer	32	0.5042	0.5583	0.1186
15	correctness	OpenLayer	32	0.5500	0.5500	0.0968
16	coherence	OpenLayer	32	0.5083	0.5083	0.0130
17	contextRelevancy	OpenLayer	24	0.5000	0.5000	0.0000
18	meanQuasiExactMatch	OpenLayer	32	0.5000	0.5000	0.0000
19	meanJsonScore	OpenLayer	32	0.5000	0.5000	0.0000
20	meanExactMatch	OpenLayer	32	0.5000	0.5000	0.0000
21	maliciousness	OpenLayer	32	0.5000	0.5000	0.0000
22	harmfulness	OpenLayer	32	0.5000	0.5000	0.0000

B.1.2 Pairwise combinations

Table B.2: Full local reviewed-label 45-case pairwise ranking.

Rank	Metric combination	Cov.	Bal. acc.	Kappa
1	hallucination + proxy_qafacteval_score	29	1.0000	1.0000
2	contextRelevancy + proxy_qafacteval_score	24	1.0000	1.0000
3	proxy_qafacteval_score + proxy_uiefid_score	32	0.9750	0.9344
4	meanSemanticSimilarity + proxy_qafacteval_score	32	0.9750	0.9344

Rank	Metric combination	Cov.	Bal. acc.	Kappa
5	meanQuasiExactMatch + proxy_qafacteval_score	32	0.9750	0.9344
6	meanJsonScore + proxy_qafacteval_score	32	0.9750	0.9344
7	meanExactMatch + proxy_qafacteval_score	32	0.9750	0.9344
8	meanEditDistance + proxy_qafacteval_score	32	0.9750	0.9344
9	meanBleu4 + proxy_qafacteval_score	32	0.9750	0.9344
10	meanBleu3 + proxy_qafacteval_score	32	0.9750	0.9344
11	meanBleu2 + proxy_qafacteval_score	32	0.9750	0.9344
12	meanBleu1 + proxy_qafacteval_score	32	0.9750	0.9344
13	maliciousness + proxy_qafacteval_score	32	0.9750	0.9344
14	harmfulness + proxy_qafacteval_score	32	0.9750	0.9344
15	faithfulness + proxy_qafacteval_score	32	0.9750	0.9344
16	correctness + proxy_qafacteval_score	32	0.9750	0.9344
17	contextUtilization + proxy_qafacteval_score	32	0.9750	0.9344
18	contextRecall + proxy_qafacteval_score	32	0.9750	0.9344
19	coherence + proxy_qafacteval_score	32	0.9750	0.9344
20	answerRelevancy + proxy_qafacteval_score	32	0.9750	0.9344
21	answerCorrectness + proxy_qafacteval_score	32	0.9750	0.9344
22	hallucination + proxy_uiefid_score	29	0.9545	0.9254
23	contextRelevancy + proxy_uiefid_score	24	0.9643	0.9155
24	meanSemanticSimilarity + proxy_uiefid_score	32	0.9333	0.8667
25	meanQuasiExactMatch + proxy_uiefid_score	32	0.9333	0.8667
26	meanJsonScore + proxy_uiefid_score	32	0.9333	0.8667
27	meanExactMatch + proxy_uiefid_score	32	0.9333	0.8667
28	meanEditDistance + proxy_uiefid_score	32	0.9333	0.8667
29	meanBleu4 + proxy_uiefid_score	32	0.9333	0.8667
30	meanBleu3 + proxy_uiefid_score	32	0.9333	0.8667
31	meanBleu2 + proxy_uiefid_score	32	0.9333	0.8667
32	meanBleu1 + proxy_uiefid_score	32	0.9333	0.8667
33	maliciousness + proxy_uiefid_score	32	0.9333	0.8667
34	harmfulness + proxy_uiefid_score	32	0.9333	0.8667
35	faithfulness + proxy_uiefid_score	32	0.9333	0.8667
36	correctness + proxy_uiefid_score	32	0.9333	0.8667
37	contextUtilization + proxy_uiefid_score	32	0.9333	0.8667
38	contextRecall + proxy_uiefid_score	32	0.9333	0.8667
39	coherence + proxy_uiefid_score	32	0.9333	0.8667
40	answerRelevancy + proxy_uiefid_score	32	0.9333	0.8667
41	answerCorrectness + proxy_uiefid_score	32	0.9333	0.8667
42	answerRelevancy + faithfulness	32	0.7083	0.4098
43	answerCorrectness + faithfulness	32	0.7083	0.4098
44	contextRecall + hallucination	29	0.6869	0.3379
45	answerRelevancy + hallucination	29	0.6869	0.3379
46	answerCorrectness + hallucination	29	0.6869	0.3379
47	answerRelevancy + meanSemanticSimilarity	32	0.6750	0.3231
48	answerRelevancy + contextRecall	32	0.6750	0.3231
49	answerRelevancy + coherence	32	0.6750	0.3231
50	answerCorrectness + meanSemanticSimilarity	32	0.6750	0.3231
51	answerCorrectness + contextRecall	32	0.6750	0.3231
52	answerCorrectness + coherence	32	0.6750	0.3231
53	contextRecall + faithfulness	32	0.6583	0.3016
54	answerRelevancy + contextRelevancy	24	0.6643	0.2987
55	answerCorrectness + contextRelevancy	24	0.6643	0.2987
56	answerRelevancy + meanQuasiExactMatch	32	0.6667	0.2941
57	answerRelevancy + meanJsonScore	32	0.6667	0.2941
58	answerRelevancy + meanExactMatch	32	0.6667	0.2941
59	answerRelevancy + meanEditDistance	32	0.6667	0.2941
60	answerRelevancy + meanBleu4	32	0.6667	0.2941
61	answerRelevancy + meanBleu3	32	0.6667	0.2941
62	answerRelevancy + meanBleu2	32	0.6667	0.2941
63	answerRelevancy + meanBleu1	32	0.6667	0.2941
64	answerRelevancy + maliciousness	32	0.6667	0.2941
65	answerRelevancy + harmfulness	32	0.6667	0.2941
66	answerRelevancy + correctness	32	0.6667	0.2941
67	answerRelevancy + contextUtilization	32	0.6667	0.2941
68	answerCorrectness + meanQuasiExactMatch	32	0.6667	0.2941
69	answerCorrectness + meanJsonScore	32	0.6667	0.2941
70	answerCorrectness + meanExactMatch	32	0.6667	0.2941
71	answerCorrectness + meanEditDistance	32	0.6667	0.2941
72	answerCorrectness + meanBleu4	32	0.6667	0.2941
73	answerCorrectness + meanBleu3	32	0.6667	0.2941
74	answerCorrectness + meanBleu2	32	0.6667	0.2941
75	answerCorrectness + meanBleu1	32	0.6667	0.2941
76	answerCorrectness + maliciousness	32	0.6667	0.2941
77	answerCorrectness + harmfulness	32	0.6667	0.2941
78	answerCorrectness + correctness	32	0.6667	0.2941
79	answerCorrectness + contextUtilization	32	0.6667	0.2941
80	answerCorrectness + answerRelevancy	32	0.6667	0.2941
81	contextRelevancy + contextUtilization	24	0.6143	0.2500
82	faithfulness + hallucination	29	0.6086	0.2478
83	contextUtilization + hallucination	29	0.6086	0.2478
84	hallucination + meanSemanticSimilarity	29	0.6162	0.2408

Rank	Metric combination	Cov.	Bal. acc.	Kappa
85	hallucination + meanEditDistance	29	0.6162	0.2408
86	faithfulness + meanBleu4	32	0.6000	0.2308
87	faithfulness + meanBleu3	32	0.6000	0.2308
88	faithfulness + meanBleu2	32	0.6000	0.2308
89	faithfulness + meanBleu1	32	0.6000	0.2308
90	contextUtilization + meanSemanticSimilarity	32	0.6000	0.2308
91	contextUtilization + meanQuasiExactMatch	32	0.6000	0.2308
92	contextUtilization + meanJsonScore	32	0.6000	0.2308
93	contextUtilization + meanExactMatch	32	0.6000	0.2308
94	contextUtilization + meanEditDistance	32	0.6000	0.2308
95	contextUtilization + meanBleu4	32	0.6000	0.2308
96	contextUtilization + meanBleu3	32	0.6000	0.2308
97	contextUtilization + meanBleu2	32	0.6000	0.2308
98	contextUtilization + meanBleu1	32	0.6000	0.2308
99	contextUtilization + maliciousness	32	0.6000	0.2308
100	contextUtilization + harmfulness	32	0.6000	0.2308
101	contextUtilization + faithfulness	32	0.6000	0.2308
102	contextUtilization + correctness	32	0.6000	0.2308
103	contextRecall + contextUtilization	32	0.6000	0.2308
104	coherence + contextUtilization	32	0.6000	0.2308
105	meanQuasiExactMatch + meanSemanticSimilarity	32	0.6083	0.2281
106	meanJsonScore + meanSemanticSimilarity	32	0.6083	0.2281
107	meanExactMatch + meanSemanticSimilarity	32	0.6083	0.2281
108	meanEditDistance + meanSemanticSimilarity	32	0.6083	0.2281
109	meanEditDistance + meanQuasiExactMatch	32	0.6083	0.2281
110	meanEditDistance + meanJsonScore	32	0.6083	0.2281
111	meanEditDistance + meanExactMatch	32	0.6083	0.2281
112	meanBleu4 + meanSemanticSimilarity	32	0.6083	0.2281
113	meanBleu4 + meanEditDistance	32	0.6083	0.2281
114	meanBleu3 + meanSemanticSimilarity	32	0.6083	0.2281
115	meanBleu3 + meanEditDistance	32	0.6083	0.2281
116	meanBleu2 + meanSemanticSimilarity	32	0.6083	0.2281
117	meanBleu2 + meanEditDistance	32	0.6083	0.2281
118	meanBleu1 + meanSemanticSimilarity	32	0.6083	0.2281
119	meanBleu1 + meanEditDistance	32	0.6083	0.2281
120	maliciousness + meanSemanticSimilarity	32	0.6083	0.2281
121	maliciousness + meanEditDistance	32	0.6083	0.2281
122	harmfulness + meanSemanticSimilarity	32	0.6083	0.2281
123	harmfulness + meanEditDistance	32	0.6083	0.2281
124	faithfulness + meanSemanticSimilarity	32	0.6083	0.2281
125	faithfulness + meanEditDistance	32	0.6083	0.2281
126	correctness + meanSemanticSimilarity	32	0.6083	0.2281
127	correctness + meanEditDistance	32	0.6083	0.2281
128	correctness + faithfulness	32	0.6083	0.2281
129	contextRecall + meanSemanticSimilarity	32	0.6083	0.2281
130	contextRecall + meanEditDistance	32	0.6083	0.2281
131	contextRecall + correctness	32	0.6083	0.2281
132	coherence + meanSemanticSimilarity	32	0.6083	0.2281
133	coherence + meanEditDistance	32	0.6083	0.2281
134	contextRecall + meanQuasiExactMatch	32	0.6250	0.2239
135	contextRecall + meanJsonScore	32	0.6250	0.2239
136	contextRecall + meanExactMatch	32	0.6250	0.2239
137	contextRecall + meanBleu4	32	0.6250	0.2239
138	contextRecall + meanBleu3	32	0.6250	0.2239
139	contextRecall + meanBleu2	32	0.6250	0.2239
140	contextRecall + meanBleu1	32	0.6250	0.2239
141	contextRecall + maliciousness	32	0.6250	0.2239
142	contextRecall + harmfulness	32	0.6250	0.2239
143	coherence + contextRecall	32	0.6250	0.2239
144	contextRelevancy + meanSemanticSimilarity	24	0.6071	0.2174
145	contextRelevancy + meanEditDistance	24	0.6071	0.2174
146	hallucination + meanBleu4	29	0.6212	0.2030
147	hallucination + meanBleu3	29	0.6212	0.2030
148	hallucination + meanBleu2	29	0.6212	0.2030
149	hallucination + meanBleu1	29	0.6212	0.2030
150	contextRecall + contextRelevancy	24	0.5929	0.1646
151	hallucination + meanQuasiExactMatch	29	0.5707	0.1493
152	hallucination + meanJsonScore	29	0.5707	0.1493
153	hallucination + meanExactMatch	29	0.5707	0.1493
154	hallucination + maliciousness	29	0.5707	0.1493
155	hallucination + harmfulness	29	0.5707	0.1493
156	correctness + hallucination	29	0.5707	0.1493
157	coherence + hallucination	29	0.5707	0.1493
158	coherence + meanBleu4	32	0.5667	0.1429
159	coherence + meanBleu3	32	0.5667	0.1429
160	coherence + meanBleu2	32	0.5667	0.1429
161	coherence + meanBleu1	32	0.5667	0.1429
162	contextRelevancy + hallucination	24	0.5714	0.1220
163	meanBleu4 + meanQuasiExactMatch	32	0.5583	0.1186
164	meanBleu4 + meanJsonScore	32	0.5583	0.1186

Rank	Metric combination	Cov.	Bal. acc.	Kappa
165	meanBleu4 + meanExactMatch	32	0.5583	0.1186
166	meanBleu3 + meanQuasiExactMatch	32	0.5583	0.1186
167	meanBleu3 + meanJsonScore	32	0.5583	0.1186
168	meanBleu3 + meanExactMatch	32	0.5583	0.1186
169	meanBleu3 + meanBleu4	32	0.5583	0.1186
170	meanBleu2 + meanQuasiExactMatch	32	0.5583	0.1186
171	meanBleu2 + meanJsonScore	32	0.5583	0.1186
172	meanBleu2 + meanExactMatch	32	0.5583	0.1186
173	meanBleu2 + meanBleu4	32	0.5583	0.1186
174	meanBleu2 + meanBleu3	32	0.5583	0.1186
175	meanBleu1 + meanQuasiExactMatch	32	0.5583	0.1186
176	meanBleu1 + meanJsonScore	32	0.5583	0.1186
177	meanBleu1 + meanExactMatch	32	0.5583	0.1186
178	meanBleu1 + meanBleu4	32	0.5583	0.1186
179	meanBleu1 + meanBleu3	32	0.5583	0.1186
180	meanBleu1 + meanBleu2	32	0.5583	0.1186
181	maliciousness + meanBleu4	32	0.5583	0.1186
182	maliciousness + meanBleu3	32	0.5583	0.1186
183	maliciousness + meanBleu2	32	0.5583	0.1186
184	maliciousness + meanBleu1	32	0.5583	0.1186
185	harmfulness + meanBleu4	32	0.5583	0.1186
186	harmfulness + meanBleu3	32	0.5583	0.1186
187	harmfulness + meanBleu2	32	0.5583	0.1186
188	harmfulness + meanBleu1	32	0.5583	0.1186
189	faithfulness + meanQuasiExactMatch	32	0.5583	0.1186
190	faithfulness + meanJsonScore	32	0.5583	0.1186
191	faithfulness + meanExactMatch	32	0.5583	0.1186
192	faithfulness + maliciousness	32	0.5583	0.1186
193	faithfulness + harmfulness	32	0.5583	0.1186
194	correctness + meanBleu4	32	0.5583	0.1186
195	correctness + meanBleu3	32	0.5583	0.1186
196	correctness + meanBleu2	32	0.5583	0.1186
197	correctness + meanBleu1	32	0.5583	0.1186
198	coherence + faithfulness	32	0.5583	0.1186
199	contextRelevancy + meanBleu4	24	0.5571	0.1176
200	contextRelevancy + meanBleu3	24	0.5571	0.1176
201	contextRelevancy + meanBleu2	24	0.5571	0.1176
202	contextRelevancy + meanBleu1	24	0.5571	0.1176
203	contextRelevancy + faithfulness	24	0.5571	0.1000
204	correctness + meanQuasiExactMatch	32	0.5500	0.0968
205	correctness + meanJsonScore	32	0.5500	0.0968
206	correctness + meanExactMatch	32	0.5500	0.0968
207	correctness + maliciousness	32	0.5500	0.0968
208	correctness + harmfulness	32	0.5500	0.0968
209	coherence + correctness	32	0.5500	0.0968
210	contextRelevancy + correctness	24	0.5143	0.0270
211	coherence + meanQuasiExactMatch	32	0.5083	0.0130
212	coherence + meanJsonScore	32	0.5083	0.0130
213	coherence + meanExactMatch	32	0.5083	0.0130
214	coherence + maliciousness	32	0.5083	0.0130
215	coherence + harmfulness	32	0.5083	0.0130
216	contextRelevancy + meanQuasiExactMatch	24	0.5000	0.0000
217	contextRelevancy + meanJsonScore	24	0.5000	0.0000
218	contextRelevancy + meanExactMatch	24	0.5000	0.0000
219	contextRelevancy + maliciousness	24	0.5000	0.0000
220	contextRelevancy + harmfulness	24	0.5000	0.0000
221	coherence + contextRelevancy	24	0.5000	0.0000
222	meanJsonScore + meanQuasiExactMatch	32	0.5000	0.0000
223	meanExactMatch + meanQuasiExactMatch	32	0.5000	0.0000
224	meanExactMatch + meanJsonScore	32	0.5000	0.0000
225	maliciousness + meanQuasiExactMatch	32	0.5000	0.0000
226	maliciousness + meanJsonScore	32	0.5000	0.0000
227	maliciousness + meanExactMatch	32	0.5000	0.0000
228	harmfulness + meanQuasiExactMatch	32	0.5000	0.0000
229	harmfulness + meanJsonScore	32	0.5000	0.0000
230	harmfulness + meanExactMatch	32	0.5000	0.0000
231	harmfulness + maliciousness	32	0.5000	0.0000

B.1.3 Three-metric combinations

Table B.3: Full local reviewed-label 45-case three-metric ranking.

Rank	Metric combination	Cov.	Bal. acc.	Kappa
1	proxy_qafacteval_score + answerRelevancy + answerCorrectness	32	0.8917	0.7966
2	proxy_qafacteval_score + answerRelevancy + contextRecall	32	0.8750	0.7895
3	proxy_qafacteval_score + answerCorrectness + contextRecall	32	0.8750	0.7895
4	proxy_uiefid_score + answerRelevancy + answerCorrectness	32	0.8500	0.7241
5	proxy_qafacteval_score + proxy_uiefid_score + answerRelevancy	32	0.8500	0.7241
6	proxy_qafacteval_score + proxy_uiefid_score + answerCorrectness	32	0.8500	0.7241

Rank	Metric combination	Cov.	Bal. acc.	Kappa
7	proxy_uiefid_score + answerRelevancy + contextRecall	32	0.8333	0.7143
8	proxy_uiefid_score + answerCorrectness + contextRecall	32	0.8333	0.7143
9	proxy_qafacteval_score + proxy_uiefid_score + contextRecall	32	0.8333	0.7143
10	proxy_qafacteval_score + meanSemanticSimilarity + contextRecall	32	0.7083	0.4717
11	proxy_qafacteval_score + meanEditDistance + meanSemanticSimilarity	32	0.7083	0.4717
12	proxy_qafacteval_score + meanEditDistance + contextRecall	32	0.7083	0.4717
13	proxy_qafacteval_score + answerRelevancy + meanSemanticSimilarity	32	0.7083	0.4717
14	proxy_qafacteval_score + answerRelevancy + meanEditDistance	32	0.7083	0.4717
15	proxy_qafacteval_score + answerCorrectness + meanSemanticSimilarity	32	0.7083	0.4717
16	proxy_qafacteval_score + answerCorrectness + meanEditDistance	32	0.7083	0.4717
17	proxy_uiefid_score + meanSemanticSimilarity + contextRecall	32	0.6667	0.3846
18	proxy_uiefid_score + meanEditDistance + meanSemanticSimilarity	32	0.6667	0.3846
19	proxy_uiefid_score + meanEditDistance + contextRecall	32	0.6667	0.3846
20	proxy_uiefid_score + answerRelevancy + meanSemanticSimilarity	32	0.6667	0.3846
21	proxy_uiefid_score + answerRelevancy + meanEditDistance	32	0.6667	0.3846
22	proxy_uiefid_score + answerCorrectness + meanSemanticSimilarity	32	0.6667	0.3846
23	proxy_uiefid_score + answerCorrectness + meanEditDistance	32	0.6667	0.3846
24	proxy_qafacteval_score + proxy_uiefid_score + meanSemanticSimilarity	32	0.6667	0.3846
25	proxy_qafacteval_score + proxy_uiefid_score + meanEditDistance	32	0.6667	0.3846
26	answerRelevancy + answerCorrectness + contextRecall	32	0.6750	0.3231
27	proxy_uiefid_score + contextUtilization + contextRecall	32	0.6250	0.2941
28	proxy_uiefid_score + answerRelevancy + contextUtilization	32	0.6250	0.2941
29	proxy_uiefid_score + answerCorrectness + contextUtilization	32	0.6250	0.2941
30	proxy_qafacteval_score + proxy_uiefid_score + contextUtilization	32	0.6250	0.2941
31	proxy_qafacteval_score + contextUtilization + contextRecall	32	0.6250	0.2941
32	proxy_qafacteval_score + answerRelevancy + contextUtilization	32	0.6250	0.2941
33	proxy_qafacteval_score + answerCorrectness + contextUtilization	32	0.6250	0.2941
34	answerRelevancy + contextUtilization + contextRecall	32	0.6000	0.2308
35	answerRelevancy + answerCorrectness + contextUtilization	32	0.6000	0.2308
36	answerCorrectness + contextUtilization + contextRecall	32	0.6000	0.2308
37	meanEditDistance + meanSemanticSimilarity + contextRecall	32	0.6083	0.2281
38	answerRelevancy + meanSemanticSimilarity + contextRecall	32	0.6083	0.2281
39	answerRelevancy + meanEditDistance + meanSemanticSimilarity	32	0.6083	0.2281
40	answerRelevancy + meanEditDistance + contextRecall	32	0.6083	0.2281
41	answerRelevancy + answerCorrectness + meanSemanticSimilarity	32	0.6083	0.2281
42	answerRelevancy + answerCorrectness + meanEditDistance	32	0.6083	0.2281
43	answerCorrectness + meanSemanticSimilarity + contextRecall	32	0.6083	0.2281
44	answerCorrectness + meanEditDistance + meanSemanticSimilarity	32	0.6083	0.2281
45	answerCorrectness + meanEditDistance + contextRecall	32	0.6083	0.2281
46	proxy_uiefid_score + contextUtilization + meanSemanticSimilarity	32	0.5000	0.0000
47	proxy_uiefid_score + contextUtilization + meanEditDistance	32	0.5000	0.0000
48	proxy_qafacteval_score + contextUtilization + meanSemanticSimilarity	32	0.5000	0.0000
49	proxy_qafacteval_score + contextUtilization + meanEditDistance	32	0.5000	0.0000
50	contextUtilization + meanSemanticSimilarity + contextRecall	32	0.5000	0.0000
51	contextUtilization + meanEditDistance + meanSemanticSimilarity	32	0.5000	0.0000
52	contextUtilization + meanEditDistance + contextRecall	32	0.5000	0.0000
53	answerRelevancy + contextUtilization + meanSemanticSimilarity	32	0.5000	0.0000
54	answerRelevancy + contextUtilization + meanEditDistance	32	0.5000	0.0000
55	answerCorrectness + contextUtilization + meanSemanticSimilarity	32	0.5000	0.0000
56	answerCorrectness + contextUtilization + meanEditDistance	32	0.5000	0.0000

B.2 Public 100-case benchmark rankings

These public rankings correspond to the 100-case comparison surface derived from the expanded QAConv sample and scored with the paper-style real UIEFID and QAFactEval backends used in the public transfer analysis.

B.2.1 Single metrics

Table B.4: Full public 100-case single-metric ranking.

Rank	Metric	Source	Cov.	AUC	Bal. acc.	Kappa
1	meanSemanticSimilarity	OpenLayer	63	1.0000	1.0000	1.0000
2	answerCorrectness	OpenLayer	62	1.0000	1.0000	1.0000
3	meanEditDistance	OpenLayer	63	0.9677	0.9365	0.8730
4	meanBleu4	OpenLayer	63	0.8594	0.8594	0.7155
5	meanBleu3	OpenLayer	63	0.8594	0.8594	0.7155
6	meanBleu2	OpenLayer	63	0.8594	0.8594	0.7155
7	meanBleu1	OpenLayer	63	0.8594	0.8594	0.7155
8	paper_uiefid_score	Paper backend	67	0.7941	0.7941	0.5846
9	hallucination	OpenLayer	52	0.6964	0.6964	0.3739
10	contextUtilization	OpenLayer	59	0.6897	0.6897	0.3833
11	paper_qafacteval_score	Paper backend	67	0.6618	0.6618	0.3202
12	coherence	OpenLayer	63	0.6396	0.6396	0.2766

Rank	Metric	Source	Cov.	AUC	Bal. acc.	Kappa
13	faithfulness	OpenLayer	63	0.6094	0.6094	0.2160
14	correctness	OpenLayer	63	0.5313	0.5313	0.0616
15	answerRelevancy	OpenLayer	63	0.5207	0.5534	0.1073
16	meanQuasiExactMatch	OpenLayer	63	0.5156	0.5156	0.0308
17	meanExactMatch	OpenLayer	63	0.5156	0.5156	0.0308
18	contextRelevancy	OpenLayer	60	0.5150	0.5150	0.0291
19	contextRecall	OpenLayer	63	0.5005	0.5005	0.0010
20	meanJsonScore	OpenLayer	63	0.5000	0.5000	0.0000
21	maliciousness	OpenLayer	63	0.5000	0.5000	0.0000
22	harmfulness	OpenLayer	63	0.5000	0.5000	0.0000

B.2.2 Pairwise combinations

Table B.5: Full public 100-case pairwise ranking.

Rank	Metric combination	Cov.	Bal. acc.	Kappa
1	meanSemanticSimilarity + paper_uiefid_score	63	1.0000	1.0000
2	meanSemanticSimilarity + paper_qafacteval_score	63	1.0000	1.0000
3	meanQuasiExactMatch + meanSemanticSimilarity	63	1.0000	1.0000
4	meanJsonScore + meanSemanticSimilarity	63	1.0000	1.0000
5	meanExactMatch + meanSemanticSimilarity	63	1.0000	1.0000
6	meanEditDistance + meanSemanticSimilarity	63	1.0000	1.0000
7	meanBleu4 + meanSemanticSimilarity	63	1.0000	1.0000
8	meanBleu3 + meanSemanticSimilarity	63	1.0000	1.0000
9	meanBleu2 + meanSemanticSimilarity	63	1.0000	1.0000
10	meanBleu1 + meanSemanticSimilarity	63	1.0000	1.0000
11	maliciousness + meanSemanticSimilarity	63	1.0000	1.0000
12	harmfulness + meanSemanticSimilarity	63	1.0000	1.0000
13	hallucination + meanSemanticSimilarity	52	1.0000	1.0000
14	faithfulness + meanSemanticSimilarity	63	1.0000	1.0000
15	correctness + meanSemanticSimilarity	63	1.0000	1.0000
16	contextUtilization + meanSemanticSimilarity	59	1.0000	1.0000
17	contextRelevancy + meanSemanticSimilarity	60	1.0000	1.0000
18	contextRecall + meanSemanticSimilarity	63	1.0000	1.0000
19	coherence + meanSemanticSimilarity	63	1.0000	1.0000
20	answerRelevancy + meanSemanticSimilarity	63	1.0000	1.0000
21	answerCorrectness + paper_uiefid_score	62	1.0000	1.0000
22	answerCorrectness + paper_qafacteval_score	62	1.0000	1.0000
23	answerCorrectness + meanSemanticSimilarity	62	1.0000	1.0000
24	answerCorrectness + meanQuasiExactMatch	62	1.0000	1.0000
25	answerCorrectness + meanJsonScore	62	1.0000	1.0000
26	answerCorrectness + meanExactMatch	62	1.0000	1.0000
27	answerCorrectness + meanEditDistance	62	1.0000	1.0000
28	answerCorrectness + meanBleu4	62	1.0000	1.0000
29	answerCorrectness + meanBleu3	62	1.0000	1.0000
30	answerCorrectness + meanBleu2	62	1.0000	1.0000
31	answerCorrectness + meanBleu1	62	1.0000	1.0000
32	answerCorrectness + maliciousness	62	1.0000	1.0000
33	answerCorrectness + harmfulness	62	1.0000	1.0000
34	answerCorrectness + hallucination	52	1.0000	1.0000
35	answerCorrectness + faithfulness	62	1.0000	1.0000
36	answerCorrectness + correctness	62	1.0000	1.0000
37	answerCorrectness + contextUtilization	58	1.0000	1.0000
38	answerCorrectness + contextRelevancy	59	1.0000	1.0000
39	answerCorrectness + contextRecall	62	1.0000	1.0000
40	answerCorrectness + coherence	62	1.0000	1.0000
41	answerCorrectness + answerRelevancy	62	1.0000	1.0000
42	contextRelevancy + meanEditDistance	60	0.9505	0.9000
43	hallucination + meanEditDistance	52	0.9435	0.8843
44	answerRelevancy + meanEditDistance	63	0.9370	0.8731
45	meanEditDistance + paper_uiefid_score	63	0.9365	0.8730
46	meanEditDistance + paper_qafacteval_score	63	0.9365	0.8730
47	meanEditDistance + meanQuasiExactMatch	63	0.9365	0.8730
48	meanEditDistance + meanJsonScore	63	0.9365	0.8730
49	meanEditDistance + meanExactMatch	63	0.9365	0.8730
50	meanBleu4 + meanEditDistance	63	0.9365	0.8730
51	meanBleu3 + meanEditDistance	63	0.9365	0.8730
52	meanBleu2 + meanEditDistance	63	0.9365	0.8730
53	meanBleu1 + meanEditDistance	63	0.9365	0.8730
54	maliciousness + meanEditDistance	63	0.9365	0.8730
55	harmfulness + meanEditDistance	63	0.9365	0.8730
56	faithfulness + meanEditDistance	63	0.9365	0.8730
57	correctness + meanEditDistance	63	0.9365	0.8730
58	contextRecall + meanEditDistance	63	0.9365	0.8730
59	coherence + meanEditDistance	63	0.9365	0.8730
60	contextUtilization + meanEditDistance	59	0.9322	0.8644
61	contextRelevancy + meanBleu4	60	0.8710	0.7354
62	contextRelevancy + meanBleu3	60	0.8710	0.7354
63	contextRelevancy + meanBleu2	60	0.8710	0.7354

Rank	Metric combination	Cov.	Bal. acc.	Kappa
64	contextRelevancy + meanBleu1	60	0.8710	0.7354
65	hallucination + meanBleu4	52	0.8750	0.7347
66	hallucination + meanBleu3	52	0.8750	0.7347
67	hallucination + meanBleu2	52	0.8750	0.7347
68	hallucination + meanBleu1	52	0.8750	0.7347
69	meanBleu4 + paper_uiefid_score	63	0.8594	0.7155
70	meanBleu4 + paper_qafacteval_score	63	0.8594	0.7155
71	meanBleu4 + meanQuasiExactMatch	63	0.8594	0.7155
72	meanBleu4 + meanJsonScore	63	0.8594	0.7155
73	meanBleu4 + meanExactMatch	63	0.8594	0.7155
74	meanBleu3 + paper_uiefid_score	63	0.8594	0.7155
75	meanBleu3 + paper_qafacteval_score	63	0.8594	0.7155
76	meanBleu3 + meanQuasiExactMatch	63	0.8594	0.7155
77	meanBleu3 + meanJsonScore	63	0.8594	0.7155
78	meanBleu3 + meanExactMatch	63	0.8594	0.7155
79	meanBleu3 + meanBleu4	63	0.8594	0.7155
80	meanBleu2 + paper_uiefid_score	63	0.8594	0.7155
81	meanBleu2 + paper_qafacteval_score	63	0.8594	0.7155
82	meanBleu2 + meanQuasiExactMatch	63	0.8594	0.7155
83	meanBleu2 + meanJsonScore	63	0.8594	0.7155
84	meanBleu2 + meanExactMatch	63	0.8594	0.7155
85	meanBleu2 + meanBleu4	63	0.8594	0.7155
86	meanBleu2 + meanBleu3	63	0.8594	0.7155
87	meanBleu1 + paper_uiefid_score	63	0.8594	0.7155
88	meanBleu1 + paper_qafacteval_score	63	0.8594	0.7155
89	meanBleu1 + meanQuasiExactMatch	63	0.8594	0.7155
90	meanBleu1 + meanJsonScore	63	0.8594	0.7155
91	meanBleu1 + meanExactMatch	63	0.8594	0.7155
92	meanBleu1 + meanBleu4	63	0.8594	0.7155
93	meanBleu1 + meanBleu3	63	0.8594	0.7155
94	meanBleu1 + meanBleu2	63	0.8594	0.7155
95	maliciousness + meanBleu4	63	0.8594	0.7155
96	maliciousness + meanBleu3	63	0.8594	0.7155
97	maliciousness + meanBleu2	63	0.8594	0.7155
98	maliciousness + meanBleu1	63	0.8594	0.7155
99	harmfulness + meanBleu4	63	0.8594	0.7155
100	harmfulness + meanBleu3	63	0.8594	0.7155
101	harmfulness + meanBleu2	63	0.8594	0.7155
102	harmfulness + meanBleu1	63	0.8594	0.7155
103	faithfulness + meanBleu4	63	0.8594	0.7155
104	faithfulness + meanBleu3	63	0.8594	0.7155
105	faithfulness + meanBleu2	63	0.8594	0.7155
106	faithfulness + meanBleu1	63	0.8594	0.7155
107	correctness + meanBleu4	63	0.8594	0.7155
108	correctness + meanBleu3	63	0.8594	0.7155
109	correctness + meanBleu2	63	0.8594	0.7155
110	correctness + meanBleu1	63	0.8594	0.7155
111	contextRecall + meanBleu4	63	0.8594	0.7155
112	contextRecall + meanBleu3	63	0.8594	0.7155
113	contextRecall + meanBleu2	63	0.8594	0.7155
114	contextRecall + meanBleu1	63	0.8594	0.7155
115	coherence + meanBleu4	63	0.8594	0.7155
116	coherence + meanBleu3	63	0.8594	0.7155
117	coherence + meanBleu2	63	0.8594	0.7155
118	coherence + meanBleu1	63	0.8594	0.7155
119	answerRelevancy + meanBleu4	63	0.8594	0.7155
120	answerRelevancy + meanBleu3	63	0.8594	0.7155
121	answerRelevancy + meanBleu2	63	0.8594	0.7155
122	answerRelevancy + meanBleu1	63	0.8594	0.7155
123	contextUtilization + meanBleu4	59	0.8448	0.6932
124	contextUtilization + meanBleu3	59	0.8448	0.6932
125	contextUtilization + meanBleu2	59	0.8448	0.6932
126	contextUtilization + meanBleu1	59	0.8448	0.6932
127	hallucination + paper_uiefid_score	52	0.8036	0.5879
128	paper_qafacteval_score + paper_uiefid_score	67	0.7941	0.5846
129	contextRelevancy + paper_uiefid_score	60	0.7903	0.5724
130	meanQuasiExactMatch + paper_uiefid_score	63	0.7813	0.5586
131	meanJsonScore + paper_uiefid_score	63	0.7813	0.5586
132	meanExactMatch + paper_uiefid_score	63	0.7813	0.5586
133	maliciousness + paper_uiefid_score	63	0.7813	0.5586
134	harmfulness + paper_uiefid_score	63	0.7813	0.5586
135	faithfulness + paper_uiefid_score	63	0.7813	0.5586
136	correctness + paper_uiefid_score	63	0.7813	0.5586
137	contextRecall + paper_uiefid_score	63	0.7813	0.5586
138	coherence + paper_uiefid_score	63	0.7813	0.5586
139	answerRelevancy + paper_uiefid_score	63	0.7813	0.5586
140	contextUtilization + paper_uiefid_score	59	0.7759	0.5559
141	contextUtilization + hallucination	49	0.7115	0.4077
142	contextRelevancy + contextUtilization	57	0.6964	0.3970
143	contextUtilization + paper_qafacteval_score	59	0.6897	0.3833

Rank	Metric combination	Cov.	Bal. acc.	Kappa
144	contextUtilization + meanQuasiExactMatch	59	0.6897	0.3833
145	contextUtilization + meanJsonScore	59	0.6897	0.3833
146	contextUtilization + meanExactMatch	59	0.6897	0.3833
147	contextUtilization + maliciousness	59	0.6897	0.3833
148	contextUtilization + harmfulness	59	0.6897	0.3833
149	contextUtilization + faithfulness	59	0.6897	0.3833
150	contextUtilization + correctness	59	0.6897	0.3833
151	contextRecall + contextUtilization	59	0.6897	0.3833
152	coherence + contextUtilization	59	0.6897	0.3833
153	answerRelevancy + contextUtilization	59	0.6897	0.3833
154	hallucination + paper_qafacteval_score	52	0.6964	0.3739
155	hallucination + meanQuasiExactMatch	52	0.6964	0.3739
156	hallucination + meanJsonScore	52	0.6964	0.3739
157	hallucination + meanExactMatch	52	0.6964	0.3739
158	hallucination + maliciousness	52	0.6964	0.3739
159	hallucination + harmfulness	52	0.6964	0.3739
160	faithfulness + hallucination	52	0.6964	0.3739
161	correctness + hallucination	52	0.6964	0.3739
162	contextRecall + hallucination	52	0.6964	0.3739
163	coherence + hallucination	52	0.6964	0.3739
164	answerRelevancy + hallucination	52	0.6964	0.3739
165	contextRelevancy + hallucination	51	0.6964	0.3685
166	contextRelevancy + paper_qafacteval_score	60	0.6774	0.3471
167	meanQuasiExactMatch + paper_qafacteval_score	63	0.6719	0.3401
168	meanJsonScore + paper_qafacteval_score	63	0.6719	0.3401
169	meanExactMatch + paper_qafacteval_score	63	0.6719	0.3401
170	maliciousness + paper_qafacteval_score	63	0.6719	0.3401
171	harmfulness + paper_qafacteval_score	63	0.6719	0.3401
172	faithfulness + paper_qafacteval_score	63	0.6719	0.3401
173	correctness + paper_qafacteval_score	63	0.6719	0.3401
174	contextRecall + paper_qafacteval_score	63	0.6719	0.3401
175	coherence + paper_qafacteval_score	63	0.6719	0.3401
176	answerRelevancy + paper_qafacteval_score	63	0.6719	0.3401
177	coherence + contextRelevancy	60	0.6429	0.2803
178	coherence + meanQuasiExactMatch	63	0.6396	0.2766
179	coherence + meanJsonScore	63	0.6396	0.2766
180	coherence + meanExactMatch	63	0.6396	0.2766
181	coherence + maliciousness	63	0.6396	0.2766
182	coherence + harmfulness	63	0.6396	0.2766
183	coherence + faithfulness	63	0.6396	0.2766
184	coherence + correctness	63	0.6396	0.2766
185	coherence + contextRecall	63	0.6396	0.2766
186	answerRelevancy + coherence	63	0.6396	0.2766
187	contextRelevancy + faithfulness	60	0.6129	0.2199
188	faithfulness + meanQuasiExactMatch	63	0.6094	0.2160
189	faithfulness + meanJsonScore	63	0.6094	0.2160
190	faithfulness + meanExactMatch	63	0.6094	0.2160
191	faithfulness + maliciousness	63	0.6094	0.2160
192	faithfulness + harmfulness	63	0.6094	0.2160
193	correctness + faithfulness	63	0.6094	0.2160
194	contextRecall + faithfulness	63	0.6094	0.2160
195	answerRelevancy + faithfulness	63	0.6094	0.2160
196	answerRelevancy + contextRelevancy	60	0.5590	0.1157
197	answerRelevancy + meanQuasiExactMatch	63	0.5534	0.1073
198	answerRelevancy + meanJsonScore	63	0.5534	0.1073
199	answerRelevancy + meanExactMatch	63	0.5534	0.1073
200	answerRelevancy + maliciousness	63	0.5534	0.1073
201	answerRelevancy + harmfulness	63	0.5534	0.1073
202	answerRelevancy + correctness	63	0.5534	0.1073
203	answerRelevancy + contextRecall	63	0.5534	0.1073
204	contextRelevancy + correctness	60	0.5323	0.0625
205	correctness + meanQuasiExactMatch	63	0.5313	0.0616
206	correctness + meanJsonScore	63	0.5313	0.0616
207	correctness + meanExactMatch	63	0.5313	0.0616
208	correctness + maliciousness	63	0.5313	0.0616
209	correctness + harmfulness	63	0.5313	0.0616
210	contextRecall + correctness	63	0.5313	0.0616
211	contextRelevancy + meanQuasiExactMatch	60	0.5161	0.0312
212	contextRelevancy + meanExactMatch	60	0.5161	0.0312
213	meanJsonScore + meanQuasiExactMatch	63	0.5156	0.0308
214	meanExactMatch + meanQuasiExactMatch	63	0.5156	0.0308
215	meanExactMatch + meanJsonScore	63	0.5156	0.0308
216	maliciousness + meanQuasiExactMatch	63	0.5156	0.0308
217	maliciousness + meanExactMatch	63	0.5156	0.0308
218	harmfulness + meanQuasiExactMatch	63	0.5156	0.0308
219	harmfulness + meanExactMatch	63	0.5156	0.0308
220	contextRecall + meanQuasiExactMatch	63	0.5156	0.0308
221	contextRecall + meanExactMatch	63	0.5156	0.0308
222	contextRelevancy + meanJsonScore	60	0.5150	0.0291
223	contextRelevancy + maliciousness	60	0.5150	0.0291

Rank	Metric combination	Cov.	Bal. acc.	Kappa
224	contextRelevancy + harmfulness	60	0.5150	0.0291
225	contextRecall + contextRelevancy	60	0.5150	0.0291
226	contextRecall + meanJsonScore	63	0.5005	0.0010
227	contextRecall + maliciousness	63	0.5005	0.0010
228	contextRecall + harmfulness	63	0.5005	0.0010
229	maliciousness + meanJsonScore	63	0.5000	0.0000
230	harmfulness + meanJsonScore	63	0.5000	0.0000
231	harmfulness + maliciousness	63	0.5000	0.0000

B.2.3 Three-metric combinations

Table B.6: Full public 100-case three-metric ranking.

Rank	Metric combination	Cov.	Bal. acc.	Kappa
1	meanSemanticSimilarity + answerCorrectness + meanEditDistance	62	0.9688	0.9356
2	meanSemanticSimilarity + meanBleu4 + meanBleu3	63	0.8594	0.7155
3	meanSemanticSimilarity + meanBleu4 + meanBleu2	63	0.8594	0.7155
4	meanSemanticSimilarity + meanBleu4 + meanBleu1	63	0.8594	0.7155
5	meanSemanticSimilarity + meanBleu3 + meanBleu2	63	0.8594	0.7155
6	meanSemanticSimilarity + meanBleu3 + meanBleu1	63	0.8594	0.7155
7	meanSemanticSimilarity + meanBleu2 + meanBleu1	63	0.8594	0.7155
8	meanBleu4 + meanBleu3 + meanBleu2	63	0.8594	0.7155
9	meanBleu4 + meanBleu3 + meanBleu1	63	0.8594	0.7155
10	meanBleu4 + meanBleu2 + meanBleu1	63	0.8594	0.7155
11	meanBleu3 + meanBleu2 + meanBleu1	63	0.8594	0.7155
12	meanSemanticSimilarity + answerCorrectness + meanBleu4	62	0.8594	0.7121
13	meanSemanticSimilarity + answerCorrectness + meanBleu3	62	0.8594	0.7121
14	meanSemanticSimilarity + answerCorrectness + meanBleu2	62	0.8594	0.7121
15	meanSemanticSimilarity + answerCorrectness + meanBleu1	62	0.8594	0.7121
16	answerCorrectness + meanBleu4 + meanBleu3	62	0.8594	0.7121
17	answerCorrectness + meanBleu4 + meanBleu2	62	0.8594	0.7121
18	answerCorrectness + meanBleu4 + meanBleu1	62	0.8594	0.7121
19	answerCorrectness + meanBleu3 + meanBleu2	62	0.8594	0.7121
20	answerCorrectness + meanBleu3 + meanBleu1	62	0.8594	0.7121
21	answerCorrectness + meanBleu2 + meanBleu1	62	0.8594	0.7121
22	meanSemanticSimilarity + meanEditDistance + meanBleu4	63	0.8281	0.6526
23	meanSemanticSimilarity + meanEditDistance + meanBleu3	63	0.8281	0.6526
24	meanSemanticSimilarity + meanEditDistance + meanBleu2	63	0.8281	0.6526
25	meanSemanticSimilarity + meanEditDistance + meanBleu1	63	0.8281	0.6526
26	meanEditDistance + meanBleu4 + meanBleu3	63	0.8281	0.6526
27	meanEditDistance + meanBleu4 + meanBleu2	63	0.8281	0.6526
28	meanEditDistance + meanBleu4 + meanBleu1	63	0.8281	0.6526
29	meanEditDistance + meanBleu3 + meanBleu2	63	0.8281	0.6526
30	meanEditDistance + meanBleu3 + meanBleu1	63	0.8281	0.6526
31	meanEditDistance + meanBleu2 + meanBleu1	63	0.8281	0.6526
32	answerCorrectness + meanEditDistance + meanBleu4	62	0.8281	0.6488
33	answerCorrectness + meanEditDistance + meanBleu3	62	0.8281	0.6488
34	answerCorrectness + meanEditDistance + meanBleu2	62	0.8281	0.6488
35	answerCorrectness + meanEditDistance + meanBleu1	62	0.8281	0.6488
36	meanSemanticSimilarity + answerCorrectness + paper_uiefid_score	62	0.7813	0.5544
37	meanSemanticSimilarity + meanEditDistance + paper_uiefid_score	63	0.7656	0.5273
38	answerCorrectness + meanEditDistance + paper_uiefid_score	62	0.7656	0.5231
39	meanSemanticSimilarity + meanBleu4 + paper_uiefid_score	63	0.7188	0.4336
40	meanSemanticSimilarity + meanBleu3 + paper_uiefid_score	63	0.7188	0.4336
41	meanSemanticSimilarity + meanBleu2 + paper_uiefid_score	63	0.7188	0.4336
42	meanSemanticSimilarity + meanBleu1 + paper_uiefid_score	63	0.7188	0.4336
43	meanBleu4 + meanBleu3 + paper_uiefid_score	63	0.7188	0.4336
44	meanBleu4 + meanBleu2 + paper_uiefid_score	63	0.7188	0.4336
45	meanBleu4 + meanBleu1 + paper_uiefid_score	63	0.7188	0.4336
46	meanBleu3 + meanBleu2 + paper_uiefid_score	63	0.7188	0.4336
47	meanBleu3 + meanBleu1 + paper_uiefid_score	63	0.7188	0.4336
48	meanBleu2 + meanBleu1 + paper_uiefid_score	63	0.7188	0.4336
49	answerCorrectness + meanBleu4 + paper_uiefid_score	62	0.7188	0.4294
50	answerCorrectness + meanBleu3 + paper_uiefid_score	62	0.7188	0.4294
51	answerCorrectness + meanBleu2 + paper_uiefid_score	62	0.7188	0.4294
52	answerCorrectness + meanBleu1 + paper_uiefid_score	62	0.7188	0.4294
53	meanEditDistance + meanBleu4 + paper_uiefid_score	63	0.7031	0.4024
54	meanEditDistance + meanBleu3 + paper_uiefid_score	63	0.7031	0.4024
55	meanEditDistance + meanBleu2 + paper_uiefid_score	63	0.7031	0.4024
56	meanEditDistance + meanBleu1 + paper_uiefid_score	63	0.7031	0.4024

Bibliography

Azaria, A. and T. Mitchell (2023). 'The Internal State of an LLM Knows When It's Lying'. en. In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Source: DOI.org (Crossref). Singapore: Association for Computational Linguistics, pp. 967–976. DOI: 10.18653/v1/2023.findings-emnlp.68. URL: https://aclanthology.org/2023.findings-emnlp.68 (visited on 28/01/2026).

Binkowski, J., D. Janiak, A. Sawczyn, B. Gabrys and T. Kajdanowicz (2025). *Hallucination Detection in LLMs Using Spectral Features of Attention Maps*. Source: arXiv.org | arXiv:2502.17598 [cs]. DOI: 10.48550/arXiv.2502.17598. arXiv: arXiv:2502.17598. URL: http://arxiv.org/abs/2502.17598 (visited on 20/01/2026).

Chang, Y., X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang, W. Ye, Y. Zhang, Y. Chang, P. S. Yu, Q. Yang and X. Xie (2023). *A Survey on Evaluation of Large Language Models*. Source: arXiv.org | arXiv:2307.03109 [cs]. DOI: 10.48550/arXiv.2307.03109. arXiv: arXiv:2307.03109. URL: http://arxiv.org/abs/2307.03109 (visited on 09/01/2026).

Commission, E. (2026a). *AI Act | Shaping Europe's digital future*. URL: https://digital-strategy.ec.europa.eu/en/policies/regulatory-framework-ai (visited on 21/04/2026).

— (2026b). *How can I demonstrate that my organisation is compliant with the GDPR?* URL: https://commission.europa.eu/law/law-topic/data-protection/rules-business-and-organisations/obligations/how-can-i-demonstrate-my-organisation-compliant-gdpr_en (visited on 21/04/2026).

— (2026c). *NIS2 Directive: securing network and information systems*. URL: https://digital-strategy.ec.europa.eu/en/policies/nis2-directive (visited on 21/04/2026).

Criscione, C. and S. Lekies (2024). 'Automated Regression Testing Framework for Large Language Models'. In.

Dataiku (2025). *Dataiku DSS - Reference documentation*. Official Dataiku DSS documentation, version 14. URL: https://doc.dataiku.com/dss/latest/ (visited on 31/05/2026).

Es, S., J. James, L. Espinosa-Anke and S. Schockaert (2025). *Ragas: Automated Evaluation of Retrieval Augmented Generation*. Source: arXiv.org | arXiv:2309.15217 [cs]. DOI: 10.48550/arXiv.2309.15217. arXiv: arXiv:2309.15217. URL: http://arxiv.org/abs/2309.15217 (visited on 09/01/2026).

Fabbri, A., C.-S. Wu, W. Liu and C. Xiong (2022). 'QAFactEval: Improved QA-Based Factual Consistency Evaluation for Summarization'. In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, pp. 2587–2601. DOI: 10.18653/v1/2022.naacl-main.187. URL: https://aclanthology.org/2022.naacl-main.187/.

Fandina, O. N., E. Farchi, S. Froimovich, R. Katan, A. Podolsky, O. Raz and A. Ziv (2025). *Automated Validation of LLM-based Evaluators for Software Engineering Artifacts*. Source: arXiv.org | arXiv:2508.02827 [cs]. DOI: 10.48550/arXiv.2508.02827. arXiv: arXiv:2508.02827. URL: http://arxiv.org/abs/2508.02827 (visited on 20/01/2026).

Gu, J., X. Jiang, Z. Shi, H. Tan, X. Zhai, C. Xu, W. Li, Y. Shen, S. Ma, H. Liu, S. Wang, K. Zhang, Y. Wang, W. Gao, L. Ni and J. Guo (2025). *A Survey on LLM-as-a-Judge*. Source: arXiv.org | arXiv:2411.15594 [cs]. DOI: 10.48550/arXiv.2411.15594. arXiv: arXiv:2411.15594. URL: http://arxiv.org/abs/2411.15594 (visited on 27/01/2026).

Gupta, A., D. Singh, G. A. Cowan, N. Kadhiresan, S. Srivastava, Y. Sriraja and Y. K. Mantri (2025). 'AUTOSUMM: A Comprehensive Framework for LLM-Based Conversation Summarization'. In: *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*. Vienna, Austria: Association for Computational Linguistics, pp. 500–509. DOI: 10.18653/v1/2025.acl-industry.35. URL: https://aclanthology.org/2025.acl-industry.35/.

Jain, S., U. Z. Ahmed, S. Sahai and B. Leong (2025). *Beyond Consensus: Mitigating the Agreeableness Bias in LLM Judge Evaluations*. Source: arXiv.org | arXiv:2510.11822 [cs]. DOI: 10.48550/arXiv.2510.11822. arXiv: arXiv:2510.11822. URL: http://arxiv.org/abs/2510.11822 (visited on 28/01/2026).

Khattab, O., A. Singhvi, P. Maheshwari, Z. Zhang, K. Santhanam, S. Vardhamanan, S. Haq, A. Sharma, T. T. Joshi, H. Moazam, H. Miller, M. Zaharia and C. Potts (2023). *DSPy: Compiling Declarative Language Model Calls into Self-Improving Pipelines*. Source: arXiv.org | arXiv:2310.03714 [cs]. DOI: 10.48550/arXiv.2310.03714. arXiv: arXiv:2310.03714. URL: http://arxiv.org/abs/2310.03714 (visited on 09/01/2026).

Kim, S., J. Shin, Y. Cho, J. Jang, S. Longpre, H. Lee, S. Yun, S. Shin, S. Kim, J. Thorne and M. Seo (2024). *Prometheus: Inducing Fine-grained Evaluation Capability in Language Models*. Source: arXiv.org | arXiv:2310.08491 [cs]. DOI: 10.48550/arXiv.2310.08491. arXiv: arXiv:2310.08491. URL: http://arxiv.org/abs/2310.08491 (visited on 09/01/2026).

Kim, S., J. Suk, S. Longpre, B. Y. Lin, J. Shin, S. Welleck, G. Neubig, M. Lee, K. Lee and M. Seo (2024). 'Prometheus 2: An Open Source Language Model Specialized in Evaluating Other Language Models'. In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Miami, Florida, USA: Association for Computational Linguistics, pp. 4334–4353. DOI: 10.18653/v1/2024.emnlp-main.248. URL: https://aclanthology.org/2024.emnlp-main.248/.

Landesberg, E. and M. Narayan (2026). *Causal Judge Evaluation: Calibrated Surrogate Metrics for LLM Systems*. Source: arXiv.org | arXiv:2512.11150 [stat]. DOI: 10.48550/arXiv.2512.11150. arXiv: arXiv:2512.11150. URL: http://arxiv.org/abs/2512.11150 (visited on 28/01/2026).

Li, A. and L. Yu (2025). 'Summary Factual Inconsistency Detection Based on LLMs Enhanced by Universal Information Extraction'. In: *Findings of the Association for Computational Linguistics: ACL 2025*. Vienna, Austria: Association for Computational Linguistics, pp. 25450–25465. DOI: 10.18653/v1/2025.findings-acl.1305. URL: https://aclanthology.org/2025.findings-acl.1305/.

Lin, C.-Y. (2004). 'ROUGE: A Package for Automatic Evaluation of Summaries'. In: *Text Summarization Branches Out*. Source: ACLWeb. Barcelona, Spain: Association for Computational Linguistics, pp. 74–81. URL: https://aclanthology.org/W04-1013/ (visited on 09/01/2026).

Liu, X., L. Zhang, S. Munir, Y. Gu and L. Wang (2025). 'VeriFact: Enhancing Long-Form Factuality Evaluation with Refined Fact Extraction and Reference Facts'. In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. Suzhou, China: Association for Computational Linguistics, pp. 17908–17925. DOI: 10.18653/v1/2025.emnlp-main.905. URL: https://aclanthology.org/2025.emnlp-main.905/.

Liu, Y., D. Iter, Y. Xu, S. Wang, R. Xu and C. Zhu (2023). *G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment*. Source: arXiv.org | arXiv:2303.16634 [cs]. DOI: 10.48550/arXiv.2303.16634. arXiv: arXiv:2303.16634. URL: http://arxiv.org/abs/2303.16634 (visited on 09/01/2026).

OpenLayer (2026a). *OpenLayer Docs: Aggregate metrics*. Official OpenLayer documentation. URL: https://docs.openlayer.com/tests/catalog/aggregate-metrics.md (visited on 06/07/2026).

— (2026b). *OpenLayer Docs: Tests Overview*. Official OpenLayer documentation. URL: https://docs.openlayer.com/tests/overview.md (visited on 06/07/2026).

Papineni, K., S. Roukos, T. Ward and W.-J. Zhu (2001). 'BLEU: a method for automatic evaluation of machine translation'. en. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*. Source: DOI.org (Crossref). Philadelphia, Pennsylvania: Association for Computational Linguistics, p. 311. DOI: 10.3115/1073083.1073135. URL: http://portal.acm.org/citation.cfm?doid=1073083.1073135 (visited on 09/01/2026).

Parliament, E. and C. of the European Union (2016). *Regulation (EU) 2016/679 (GDPR)*. URL: https://eur-lex.europa.eu/eli/reg/2016/679/oj (visited on 21/04/2026).

— (2022). *Directive (EU) 2022/2555 (NIS2 Directive)*. URL: https://eur-lex.europa.eu/eli/dir/2022/2555/oj (visited on 21/04/2026).

Parliament, E. and C. of the European Union (2024). *Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (AI Act)*. URL: https://eur-lex.europa.eu/eli/reg/2024/1689/oj (visited on 21/04/2026).

Sohapy, O. (2024). 'Continuous Evaluation And Testing Pipelines For Generative AI Systems'. In: URL: https://www.researchgate.net/publication/398861986.

Tang, L., I. Shalyminov, A. Wong, J. Burnsky, J. Vincent, Y. Yang, S. Singh, S. Feng, H. Song, H. Su, L. Sun, Y. Zhang, S. Mansour and K. McKeown (2024). 'TofuEval: Evaluating Hallucinations of LLMs on Topic-Focused Dialogue Summarization'. In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Mexico City, Mexico: Association for Computational Linguistics, pp. 4455–4480. DOI: 10.18653/v1/2024.naacl-long.251. URL: https://aclanthology.org/2024.naacl-long.251/.

Verga, P., S. Hofstatter, S. Althammer, Y. Su, A. Piktus, A. Arkhangorodsky, M. Xu, N. White and P. Lewis (2024). *Replacing Judges with Juries: Evaluating LLM Generations with a Panel of Diverse Models*. Source: arXiv.org | arXiv:2404.18796 [cs]. DOI: 10.48550/arXiv.2404.18796. arXiv: arXiv:2404.18796. URL: http://arxiv.org/abs/2404.18796 (visited on 27/01/2026).

Wan, Y., H. Tan, X. Zhu, X. Zhou, Z. Li, Q. Lv, C. Sun, J. Zeng, Y. Xu, J. Lu, Y. Liu and Z. Guo (2025). 'FaStFact: Faster, Stronger Long-Form Factuality Evaluations in LLMs'. In: *Findings of the Association for Computational Linguistics: EMNLP 2025*. Suzhou, China: Association for Computational Linguistics, pp. 23814–23854. DOI: 10.18653/v1/2025.findings-emnlp.1295. URL: https://aclanthology.org/2025.findings-emnlp.1295/.

Wu, C.-S., A. Madotto, W. Liu, P. Fung and C. Xiong (2022). 'QAConv: Question Answering on Informative Conversations'. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pp. 5389–5411. DOI: 10.18653/v1/2022.acl-long.370. URL: https://aclanthology.org/2022.acl-long.370/.

You, Z. and Y. Guo (June 2026). 'PlainQAFact: Retrieval-augmented factual consistency evaluation metric for biomedical plain language summarization'. In: *Journal of Biomedical Informatics* 178, p. 105019. DOI: 10.1016/j.jbi.2026.105019. URL: https://doi.org/10.1016/j.jbi.2026.105019.

Zhang, Z., C. Zheng, Y. Wu, B. Zhang, R. Lin, B. Yu, D. Liu, J. Zhou and J. Lin (2025). *The Lessons of Developing Process Reward Models in Mathematical Reasoning*. Source: arXiv.org | arXiv:2501.07301 [cs]. DOI: 10.48550/arXiv.2501.07301. arXiv: arXiv:2501.07301. URL: http://arxiv.org/abs/2501.07301 (visited on 20/01/2026).

Zhang, Z., S. Cui, Y. Lu, J. Zhou, J. Yang, H. Wang and M. Huang (2025). *Agent-SafetyBench: Evaluating the Safety of LLM Agents*. Source: arXiv.org | arXiv:2412.14470 [cs]. DOI: 10.48550/arXiv.2412.14470. arXiv: arXiv:2412.14470. URL: http://arxiv.org/abs/2412.14470 (visited on 20/01/2026).

Zheng, L., W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez and I. Stoica (2023). *Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena*. Source: arXiv.org | arXiv:2306.05685 [cs]. DOI: 10.48550/arXiv.2306.05685. arXiv: arXiv:2306.05685. URL: http://arxiv.org/abs/2306.05685 (visited on 21/01/2026).