UNIVERSITY OF GLOUCESTERSHIRE

SCHOOL OF COMPUTING AND ENGINEERING

ARTIFICIAL INTELLIGENCE BSC

Beyond Static Benchmarks: A Modular and Dynamic Framework for Domain-Specific LLM Evaluation

Author:
BRANDON TOEWS

Supervisor:
DR. MAH-RUKH FIDA

June 16, 2025

Contents

Declaration

I hereby declare that this dissertation is entirely my own work and that, to the best of my knowledge and belief, it does not contain material previously published or written by another person except where due acknowledgment has been made in the text. I further confirm that this dissertation has not been submitted either in whole or part for any degree or diploma at this or any other university.

Brandon Toews	Date

1 Introduction

1.1 Motivations

Large Language Models (LLMs) have revolutionized natural language processing (NLP), demonstrating remarkable capabilities across diverse domains (McIntosh et al. 2024). Their increasing integration into critical sectors such as healthcare, law, and cybersecurity necessitates rigorous evaluation of their performance (McIntosh et al. 2024). While general benchmarks have contributed to progress, these benchmarks often fail to capture specific requirements crucial to specialized applications, indicating the need for domain-specific evaluation frameworks.

1.2 Problem Statement

Current benchmarking practices face several critical challenges. Benchmark data contamination (BDC), resulting from overlaps between training datasets and test benchmarks, significantly inflates performance metrics and misleads stakeholders about the true capabilities of a model (Golchin & Surdeanu 2024). Many evaluation methods also use probability-based scoring in multiple-choice tests, which inadequately represents the true reasoning and generation capabilities of a model (Lyu, Wu, & Aji 2024). Traditional static benchmarks do not adequately evaluate LLM performance in evolving knowledge domains, as demonstrated by substantial performance declines observed when models are tested against dynamically evolving benchmarks (Xia, Deng, & L. Zhang 2024).

1.3 Research Questions

This research seeks to answer the following questions:

- How can a domain-adaptable LLM benchmarking framework mitigate common evaluation challenges such as data contamination and evaluation method misalignment?
- To what extent does retrieval-based augmentation improve LLM performance in specialized tasks?
- What multidimensional metrics best capture nuanced performance aspects of LLMs?
- How can benchmarks adapt to evolving domain knowledge while maintaining consistent performance evaluation?

1.4 Research Objectives

Aligned with these research questions, the objectives of this study are:

- Develop a modular and dynamic benchmarking framework that is adaptable to specialized domains.
- Implement and evaluate multiple LLM architectures and augmentation strategies, including retrieval augmentation and knowledge graph integration.
- Establish a comprehensive multimetric evaluation approach.
- Create mechanisms for continual benchmark evolution to reflect updates in domain knowledge.

1.5 Scope

This research focuses specifically on domain-specific benchmarking frameworks for LLMs demonstrated through cybersecurity compliance advisory tasks, addressing a rapidly evolving and complex knowledge domain. The applicability of this framework to similar specialized domains is also examined.

1.6 Ethical Considerations

Key ethical considerations include the following:

- Potential misinformation risks due to incorrect or misleading model output.
- Transparency regarding known model limitations.
- Fairness of evaluation across diverse domain requirements.
- Privacy concerns addressed through the use of publicly available and synthetic data.
- Mitigating automation bias by emphasizing justification and evidence-supported responses.

1.7 Layout

The dissertation proceeds with a Literature Review examining current practices, emerging frameworks, and some distilled conclusions based on the review; followed by Output Design detailing the methodology, architecture, and evaluation parameters of the proposed framework; and concludes with comprehensive references.

2 Literature Review

2.1 Background

Benchmarking LLMs serves several critical functions. Benchmarks provide a solid basis for performance evaluations of how well LLMs perform across different capabilities such as reasoning, knowledge retrieval, language understanding, etc. They in turn allow researchers to quantify improvements over time and compare different model architectures and training techniques. Organizations can then use benchmark results to inform the appropriate selection of models for specific applications based on their strengths and weaknesses. Benchmarks highlight areas where models struggle, guiding future research efforts. Many benchmarks now include evaluations of harmful outputs, biases, and other safety concerns. Despite their importance and utility, LLM benchmarks face several significant challenges.

Benchmarking LLMs has historically been crucial for measuring progress and comparing models. Standard benchmarks such as General Language Understanding Evaluation (GLUE), SuperGLUE and Massive Multitask Language Understanding (MMLU) provided fixed datasets and tasks to evaluate core NLP capabilities, enabling consistent comparisons between models (McIntosh et al. 2024). These benchmarks focus on end-to-end metrics for tasks such as answering questions, translating, or common-sense reasoning, treating the model as a black box that produces an answer per query. This approach helped quantify improvements and highlight strengths and weaknesses of the model over time. Organizations could use benchmark results to guide model selection for applications and researchers could identify where models struggle to spur future advances. Over time, benchmarks also began to assess ethical and safety dimensions, such as bias and harmful output, to ensure responsible artificial intelligence (AI) development. In short, LLM benchmarks serve as standardized yardsticks for performance, driving the evolution of the field.

However, foundational benchmark practices come with assumptions and limitations. One such assumption is that improvements in model performance on high-profile benchmarks like MMLU, HumanEval, etc., are indicative of the model developing a deeper understanding or acquiring certain abilities. In addition, a common practice in LLM evaluation is using the predicted probabilities of a model to choose answers in tasks, especially multiple-choice questions, instead of letting the model generate an answer in natural language. Many benchmark evaluations, for efficiency, will have the model score each possible answer (option A, B, C, etc.) and pick the highest-probability option as its answer. This method is label-based or probability-based evaluation, as opposed to generation-based evaluation, where the model actually produces an answer, sometimes with an explanation, and that output is checked. The main reason why many current evaluation frameworks default to probability-based scoring is due to computational constraints, namely, it is faster and cheaper to get the probability of a model on a fixed set of answers

than to generate text. The assumption is that this method of evaluation serves as a suitable proxy for the behavior of a model in generation-based tasks. Finally, long-standing statistical metrics such as accuracy for classification, F1, precision/recall, BLEU/ROUGE for generation, etc., have provided a way to objectively quantify performance. However, these statistical measures are limited in their ability to properly stratify and score nuanced outputs that often result from real-world application.

A growing trend in LLM research and application is the incorporation of tools, such as Retrieval-Augmented Generation (RAG), to enhance LLM capabilities and accuracy. A RAG system has (1) a retrieval component that fetches documents relevant to the query, and (2) a generation component that produces a final answer using both its internal knowledge and the retrieved context (see Figure 2.1), with the aim to reducing hallucinations and keeping knowledge up-to-date (Gao et al. 2024, p. 1). Traditional LLM benchmarks assume that the model's knowledge and reasoning are self-contained, that the model knows the answer from training or must infer it from a given prompt context. They do not explicitly test the model's ability to retrieve and use external information, nor do they decompose performance into subtasks. For example, MMLU assesses knowledge across domains with multiple-choice questions, but a model's score conflates knowledge recall and reasoning, without isolating whether an error was due to lack of information or incorrect reasoning. Similarly, holistic evaluation efforts like Holistic Evaluation of Language Models (HELM) provide diverse metrics (accuracy, calibration, bias, etc.), but still treat the model as a single black-box system producing an answer per query. In short, conventional benchmarks excel at measuring what answer the model gives, but not how it got that answer.

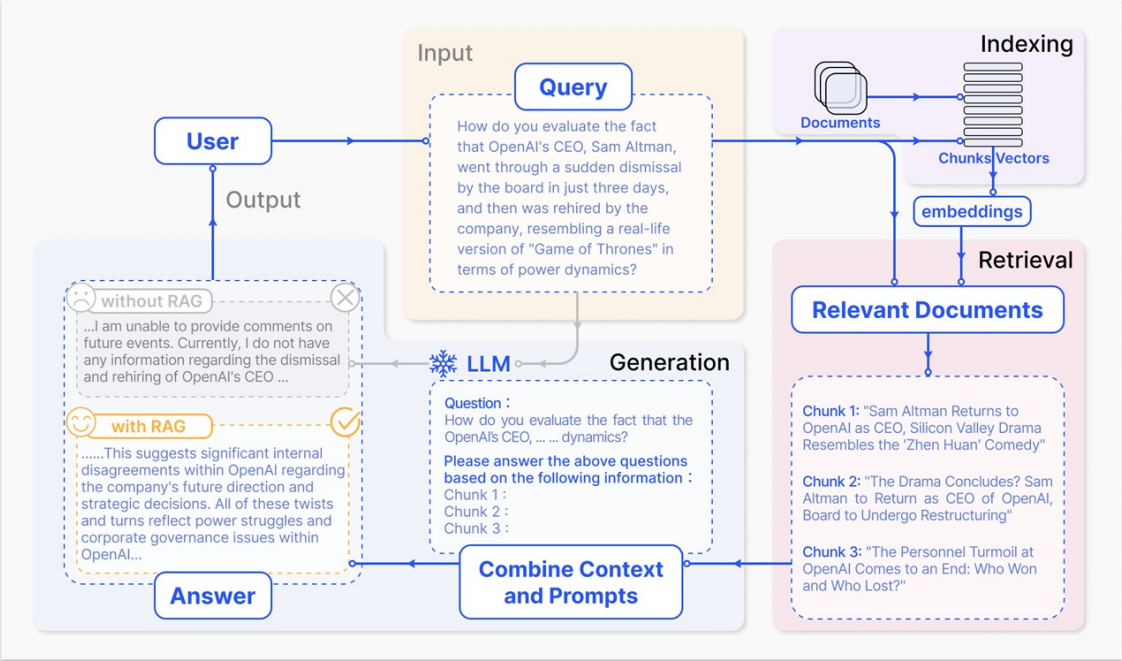

Figure 2.1: A representative instance of the RAG process applied to question answering. It mainly consists of 3 steps. 1) Indexing. Documents are split into chunks, encoded into vectors, and stored in a vector database. 2) Retrieval. Retrieve the Top k chunks most relevant to the question based on semantic similarity. 3) Generation. Input the original question and the retrieved chunks together into LLM to generate the final answer. Source: (Gao et al. 2024, p. 3)

2.2 Related Work

2.2.1 Benchmark Contamination Issues

One critical threat to benchmark integrity is BDC, which is the leakage of test examples into the training data of a model. Xu et al. (2024) underscore that BDC is widespread and often hard to detect, yet undermines the credibility of benchmark results. Essentially, many high-profile benchmarks have had some of their questions or answers seen by large models during training, for instance, models memorized solutions to portions of MMLU or HumanEval. The reasoning is that it leads to inflated evaluation scores as a model may appear to excel at a task not because it truly mastered the underlying skill, but because it recalls the answers from memory. The survey also finds that while some researchers originally argued that memorizing answers is not necessarily bad or unavoidable, the prevalent view is that it "poses significant challenges to the reliability and validity of LLM evaluations" (Xu et al. 2024, p. 1). Although the survey is limited to only compiling known techniques, one gap they identify is the lack of a unified, systematic approach to defining and tackling BDC. They highlight that no single mitigation will solve the issue completely given the scale of LLM training data, which often sweeps up entire Internet archives.

An analysis by Golchin & Surdeanu (2024) revealed specific contaminated benchmarks, for example, finding that datasets like AG News and XSum had leaked content when tested with GPT-4. These findings reinforce Xu et al. (2024)'s conclusions that training data contamination is common and widespread across many tasks.

2.2.2 Misalignment in LLM Evaluation Practices

Another issue is the misalignment between how we evaluate LLMs in benchmarks and how LLMs are actually used in the real world. McIntosh et al. (2024) argue that many benchmarks fail to measure what we really care about. In their audit of 23 state-of-the-art benchmarks, they found that evaluations often rely on simplistic proxies or narrow setups that do not reflect genuine performance in deployment. For example, several benchmarks credit a model for getting the correct answer but do not check whether the model's reasoning process made sense or if it just guessed patterns. Many benchmarks lack diversity in inputs or values, meaning a model can score well by overfitting to stereotyped prompts. McIntosh et al. (2024) observed issues such as cultural bias in test content and high sensitivity to prompt formatting, which indicates that benchmark scores might not translate to real-world reliability. This focus on final answers rather than the process means that models can sometimes 'game' benchmarks, thus achieving good scores through shortcuts or pattern matching rather than true understanding. In short, models could appear state-of-the-art on a leaderboard yet disappoint in practical usage because the evaluation was misaligned with real usage conditions. A limitation of the study by McIntosh et al. (2024) itself is that their critique is qualitative and does not provide a quantitative fix. However, the study unveils a diversity of issues present within benchmarking practices that highlight the need for intentional improvement.

A concrete example of evaluation misalignment is given by Lyu, Wu, & Aji (2024), which brings to light some compelling insights into the current limitations of predicted probability-based evaluations. The primary metrics considered in the study were (1) accuracy under each evaluation mode, probabilities-based evaluations versus generation-based, and (2) consistency between the two. They show that the probability-based method "inadequately aligns with generation-based prediction" (Lyu, Wu, & Aji 2024, p.1) creating a misrepresentation of the performance and behavior of a model (see Figure 2.2). For example, the option that the model assigns highest probability is not always the one it would output when asked to explain or answer directly. The misalignment could manifest as the model having a hidden preference it does not act on when forced to choose via probabilities. Essentially, the authors treat the generation-based outcome as the 'ground truth' of what the model really believes or would do in practice, and they check how often the probability proxy matches that. This underscores that the convenience and cost efficiency of probability-based evaluations come at the cost of not fully understanding the real-world behavior of a model. A limitation of Lyu, Wu, & Aji (2024)'s study is that they assume generation-based evaluation is the 'ground truth' measure of performance, but generation brings its own uncertainties. For example, a model might generate a correct answer phrased differently from the expected answer, and automatic evaluation could count that wrong unless carefully handled. They do note that evaluating generation often requires careful parsing or human judgment, which is why probabilistic methods gained

popularity in the first place, but they do not propose solutions to ameliorate the extra costs of doing so.

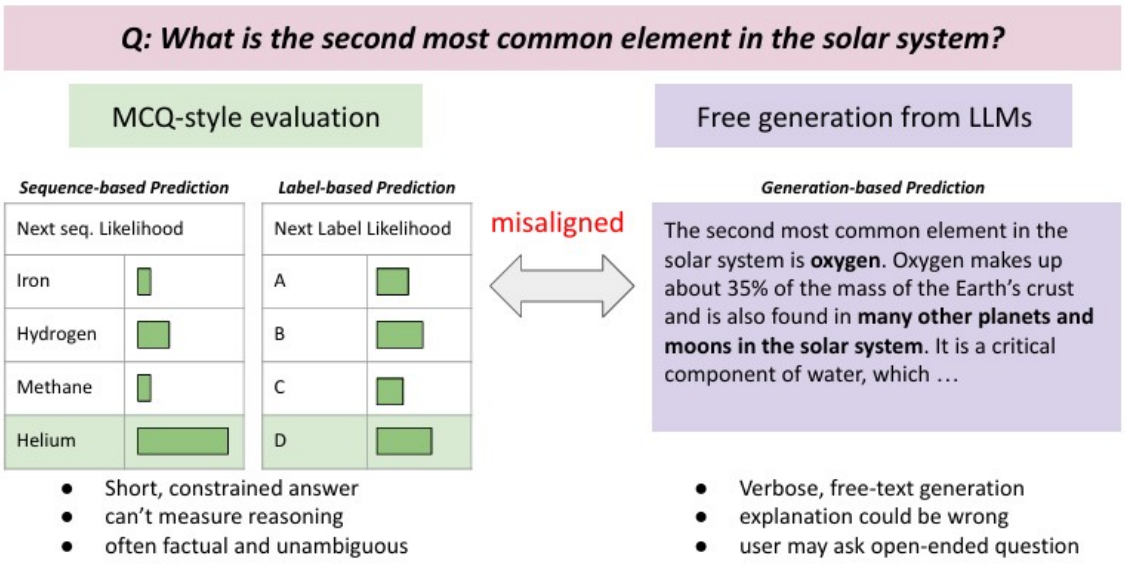

Figure 2.2: An illustration of label-based, sequence-based and generation-based predictions for evaluating LLMs on NLP benchmarks. Source: (Lyu, Wu, & Aji 2024, p.2)

2.2.3 Newer Benchmarking Frameworks

Xia, Deng, & L. Zhang 2024 tackle the problem of benchmarks becoming overfitted by introducing EvoEval, an approach to dynamically evolve coding challenges (see Figure 2.3). They start from popular coding benchmarks like HumanEval and MBPP and use LLMs themselves to generate new variations of these programming problems. The idea is to create challenges that are similar in spirit but sufficiently different in surface details or domain, so that an LLM which memorized the original solutions will be caught off guard. The study uses standard coding task metrics, such as the pass rate of generated code against unit tests (e.g. pass@k metrics), to evaluate model proficiency. However, they compared metrics on the original benchmarks versus the evolved ones. A key finding in their analysis is the drop in performance (%) when switching to evolved tasks. Models that previously topped the coding leaderboards saw absolute drops in accuracy of 20%–47% in the evolved problems, and many fell dramatically in ranking. This reveals that existing coding benchmarks probably overestimated true model competency, as models had effectively overfit on the narrow distribution or even leaked solutions. Xia, Deng, & L. Zhang 2024 highlight phenomena like brittleness to slight rewording, for instance, a prompt asking for a solution "in two sentences" might confuse a model that learned to expect a certain format. Some EvoEval tasks require combining two simpler tasks, which many models struggled with, showing weakness in multistep reasoning or code synthesis. Some notable limitations are: (1) using an LLM to generate benchmarks might introduce its own biases or errors. (2) The authors had to ensure that the new problems were neither trivial variations nor unsolvable; this probably required manual curation,

which could be time consuming. (3) EvoEval is specifically focused on the domain of coding. Although their findings pertain to code benchmarks, it is entirely possible that these trends are prevalent in other domains but would require their own "evolution" to prove definitively. (4) Finally, as models improve or new training data appear, possibly including EvoEval itself in the future, this approach would have to continually generate further evolved tasks.

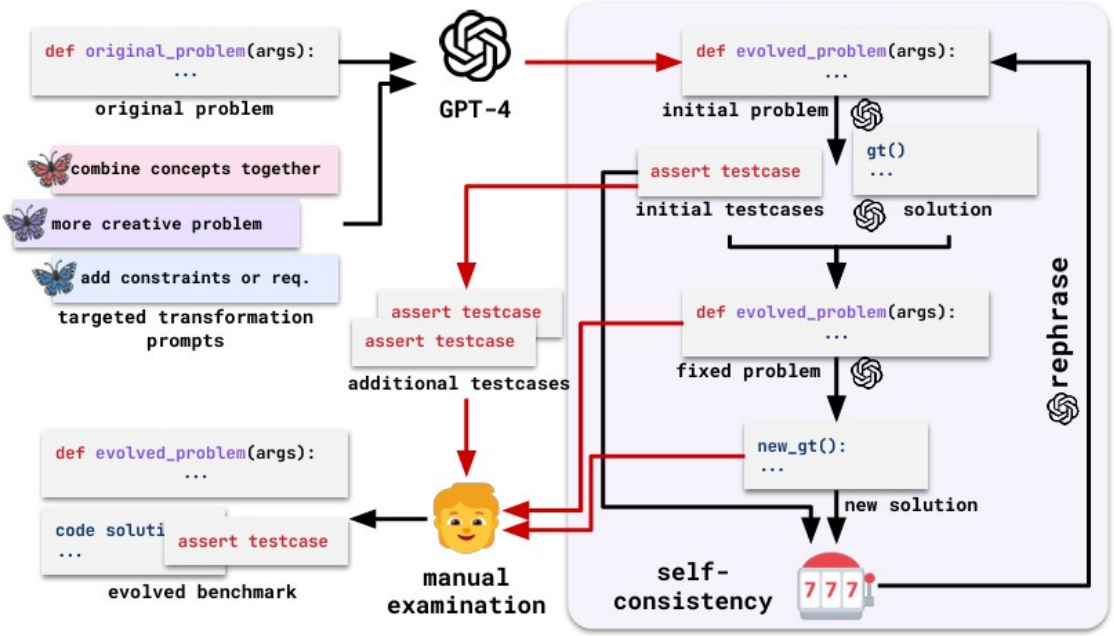

Figure 2.3: Overview of EvoEval evolving problem generation pipeline. Source: (Xia, Deng, & L. Zhang 2024, p.4)

Dalvi et al. (2024) take a very practical angle by introducing LLMeBench, a benchmarking framework aimed at making LLM evaluation flexible, extensible, and efficient. LLMeBench comes with generic dataset loaders, supports multiple model providers, and has many pre-implemented standard evaluation metrics (see Figure 2.4). Importantly, it supports in-context learning setups like zero-shot and few-shot, meaning it can automate prompts for models with given examples if needed. The framework does not invent new metrics, but streamlines the use of existing ones such as accuracy for classification, F1, precision/recall, BLEU/ROUGE, etc. Another aspect is that by supporting multiple tasks, LLMeBench encourages the use of holistic evaluation. LLMeBench can easily run a model through a battery of tasks such as translation, question-answering, and reasoning puzzles using a single framework. Therefore, this framework fosters comprehensive evaluation rather than a single-metric focus. One limitation is that LLMeBench, while flexible, is only as good as the benchmarks one feeds into it, but it does not solve what to evaluate, it helps with how. If one were to use poor-quality or biased benchmark data, the framework would faithfully report metrics, but the insights would still depend on the input. In addition, LLMeBench will not automatically flag contamination issues or suggest new tasks and is therefore susceptible to the issues already raised. In terms of real-world applicability,

LLMeBench addresses the barrier to entry, which could allow domain experts to focus more on writing good questions and let the tool handle the rest. McIntosh et al. (2024) noted that many benchmarks suffer from "implementation inconsistencies" and slow iteration, which LLMeBench directly tackles by providing a consistent implementation and enabling quick reruns of evaluations. To summarize key challenges in standard LLM benchmarking, Table 2.1 presents a comparative analysis of major studies on benchmark contamination, overfitting, and misalignment.

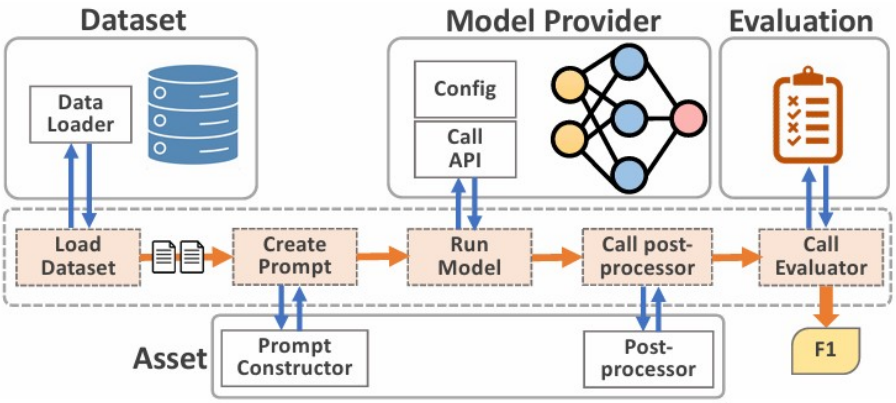

Figure 2.4: The architecture of the LLMeBench framework. The dotted boxes represent the core implemented modules of the architecture. Customization for new tasks, datasets, and models can be done on `Dataset`, `Model Provider`, `Evaluation`, and `Asset` modules. Source: (Dalvi et al. 2024, p.1)

Table 2.1: General LLM Benchmarking Issues

Paper & Authors	Benchmarking Issue	Proposed Approach	Key Findings	Future Directions
McIntosh et al. 2024	Overfitting in static benchmarks; misalignment with real-world use.	Analyzed 23 benchmarks, advocating for evolving test sets.	Benchmarks overestimate model reliability due to gaming.	Implementing truly dynamic benchmarks remains unsolved.
Xu et al. 2024	Test contamination leads to inflated scores.	Advocates systematic dataset filtering and leak detection.	Training data contamination is widespread.	Large-scale detection remains a challenge.
Lyu, Wu, & Aji 2024	Misalignment between probability-based and free-form evaluation.	Compares probability-based vs. generated response accuracy.	Probability selection misrepresents true model ability.	Efficient generation-based scoring remains an open problem.
Xia, Deng, & L. Zhang 2024	Overfitting in static coding benchmarks.	EvoEval: Dynamically mutates coding tasks.	Models show 20-47% drop on evolved problems.	Needs broader application beyond coding.
Dalvi et al. 2024	Fragmented evaluation tools.	*LLMeBench*: Unified multi-metric benchmarking.	Standardizes model comparisons.	Still depends on pre-existing datasets.

2.2.4 RAG & Multi-Task Evaluation

The survey by Gao et al. (2024) provides a comprehensive overview of RAG systems and specifically introduces metrics and benchmarks to assess RAG models alongside an up-to-date evaluation framework. RAG evaluation frameworks go one step further by explicitly measuring the different stages of a retrieve-and-generate pipeline (see Figure 2.5) (Es et al. 2023). Evaluating such systems involves multidimensional metrics:

- **Retrieval quality** – Is the retrieved context relevant and sufficient for the query?
- **Generation quality** – Is the final answer correct and is it faithful to the retrieved evidence?
- **Integration performance** – How well does the LLM incorporate the retrieved information? Does it avoid hallucination or ignore evidence?

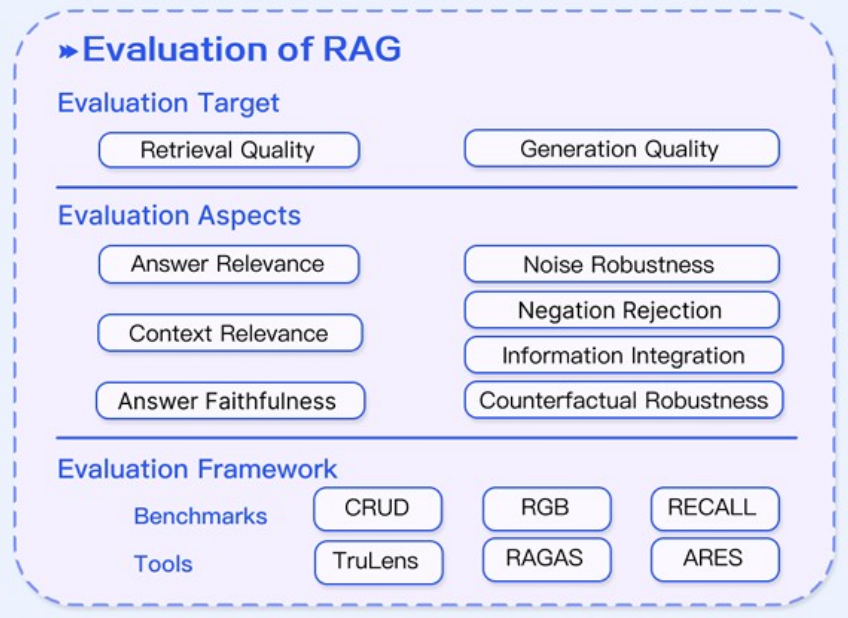

Figure 2.5: Overview of RAG Evaluations. Adapted from: (Gao et al. 2024, p. 16)

Recent RAG-specific evaluators such as Retrieval Augmented Generation Assessment (RAGAS) (Es et al. 2023) and the Automated RAG Evaluation System (ARES) (Saad-Falcon et al. 2024) exemplify this approach. RAGAS introduces a reference-free multimetric framework to automatically assess RAG pipelines. It proposes a suite of zero-shot LLM-based evaluation metrics that target each aspect of the pipeline, (1) the relevance of the retrieved passages, (2) the faithfulness of the LLM's response to those passages, and (3) the overall quality of the response. In practice, RAGAS uses prompt-based evaluation with an LLM, such as GPT-4, to score output on several dimensions, eliminating the need for ground truth answers for every query. Concretely, the RAGAS score combines two metrics for the generation stage, faithfulness and answer relevancy, and two metrics for retrieval, context precision, and context recall. Faithfulness measures whether the answer

accurately reflects the information found in the retrieved documents (i.e., no unsupported claims), and answer relevancy checks if the answer addresses the query directly without unnecessary content. Context precision/recall evaluate whether retrieved passages are relevant to the query and cover the needed information.

In contrast, existing LLM benchmarks rarely assess these facets separately. Traditional QA evaluations might reward an answer that is correct, but they will not detect whether the model had to hallucinate missing facts or if it ignored provided context. RAG evaluation frameworks fill this gap by explicitly rewarding answers that are correct and grounded in evidence. For example, Gao et al. (2024) note that RAG models are particularly aimed at mitigating hallucinations and providing traceable sources, so the evaluation criteria emphasize factuality and source attribution. This is aligned with emerging benchmarks such as Knowledge Intensive Language Tasks (KILT), a set of knowledge-intensive tasks that require models to retrieve supporting Wikipedia passages, where evaluations similarly combine a correctness measure with evidence retrieval accuracy. RAGAS and ARES add automation and finer granularity to this more comprehensive style of evaluation. ARES, for example, evaluates RAG systems along three dimensions, context relevance, answer faithfulness, and answer relevance. It creates synthetic QA pairs to fine-tune lightweight 'judge' models that can score each aspect, and calibrates them with a small number of human-annotated examples. This allowed ARES to reliably evaluate RAG performance on eight knowledge-intensive tasks, from benchmarks like KILT and SuperGLUE, with only a few hundred human labels, by having the learned judges predict ratings for thousands of cases. Table 2.2 summarizes recent advancements in RAG benchmarking, highlighting how different studies address retrieval accuracy, generation quality, and multistage evaluation.

Table 2.2: RAG / Multi-Task Benchmarking Issues

Paper & Authors	**Benchmarking Issue**	**Proposed Approach**	**Key Findings**	**Future Directions**
Gao et al. 2024	Conventional benchmarks ignore retrieval models.	Multi-stage RAG evaluation: retrieval, generation, integration.	Distinguishes retrieval quality from generated response.	Widespread adoption of multi-stage metrics is needed.
Es et al. 2023	Lack of automated RAG evaluation.	Uses LLM-based scoring for retrieval and accuracy.	Enables scalable evaluation without ground truth labels.	LLM-generated scores may introduce biases.
Saad-Falcon et al. 2024	Scaling RAG evaluation with minimal human labeling.	*ARES*: Uses few human labels to train evaluators.	Reduces manual scoring dependency.	Updating evaluators for evolving data remains an issue.
Rasiah et al. 2024	Benchmarks too simplistic for real-world applications.	*SCALE*: Legal domain benchmark with long-text, multilingual tasks.	Models struggle with long, domain-specific inputs.	Similar domain-specific benchmarks are needed.
Friel, Belyi, & Sanyal 2025	No standardized RAG benchmark or explainability.	*RAGBench*: 100k examples + *TRACe*, an explainability framework.	Enables large-scale, explainable RAG evaluation.	Ensuring alignment between automated and human scoring remains an issue.

2.3 Conclusions

Examining these works collectively, several common themes emerge with regard to general principles and best practices for domain-specific LLM benchmarks.

2.3.1 Maintaining Benchmark Integrity

A consistent concern is to ensure that the evaluation truly measures generalization, not memory. Xu et al. (2024)'s survey highlighted how pervasive this issue is, warning that models often get inflated scores by 'knowing' test answers in advance. It is important to distinguish that the memorization of the answers is not the problem in question, but can be an indicator of the real issue, which is brittleness and overfitting. Although the mere presence of model memorization does not definitively prove overfitting has occurred benchmarking solutions should be designed to elucidate this issue. Works such as Xia, Deng, & L. Zhang (2024)'s EvoEval, which regenerates and mutates test questions, show promise in this regard. The general principle made clear in this review is to keep the benchmarks novel and unpredictable. Echoing this theme, McIntosh et al. (2024) advocates for dynamic benchmarks that evolve so that models cannot simply overfit. For domain-specific benchmarks, this might mean using proprietary or freshly collected data that was not in common pre-training corpora, or continuously adding new test cases over time. In doing so, we maintain the integrity of the benchmark, ensuring that the scores remain a trustworthy signal of the capability of a model.

2.3.2 Evolving Evaluation to Reflect Real Usage

There is a clear trend towards making evaluations more holistic, realistic and aligned with how LLMs are actually used. Lyu, Wu, & Aji (2024) explicitly show that evaluation methods can be misaligned, if we optimize for convenience like in the case of multiple-choice probability evaluation, we might miss the true behavior of the model. Similarly, McIntosh et al. (2024) and Rasiah et al. (2024) push for benchmarks that test models in more complex scenarios (multi-turn interactions, long documents, diverse languages) because real-world tasks are complex. A general principle is that benchmarks should simulate the conditions under which we expect the model to perform. For instance, if an LLM will be used by non-English speakers, the benchmark should have multilingual components, as SCALE does. If the model will function as a dialogue agent, the benchmark should include interactive prompts or multistep reasoning tasks, not just single-turn queries. We see this in RAGBench and Li, Yuan, & Z. Zhang (2024)'s work by incorporating retrieval into the evaluation system. Since many real deployments use tools to assist LLMs, the benchmarks must evaluate that combined system. Another aspect of evolving evaluation is the use of multimetric assessment. Instead of a one-number accuracy or BLEU, there is a move to break down the performance into submetrics such as RAGBench's TRACe to get a more complete picture. This is especially important in domain-specific contexts like legal or medical, where an answer might need to be not only correct but also justified and safe. By having granular metrics such as correctness, justification adequacy, harmful content check, etc., benchmark results become more actionable, developers can see why a

model fails and improve it. In summary, best practices involve designing benchmarks that are high-fidelity proxies for deployment scenarios: dynamic, diverse, and evaluated on multiple axes of quality.

2.3.3 Leveraging Tools and Hybrid Approaches

Another emerging principle is that benchmarks can and should test a model's ability to use tools or external knowledge, rather than confining the evaluation to end-to-end prompting. Li, Yuan, & Z. Zhang (2024) and Friel, Belyi, & Sanyal (2025) both illustrate this by focusing on retrieval-augmented settings. This intersection of tool-use with benchmarking is increasingly relevant as advanced models often come with an ecosystem of plugins or support systems. A domain-specific example: A cybersecurity LLM might have access to a database of known vulnerabilities; a good benchmark would measure how well the LLM queries that database and integrates the results into its advice, not just what it remembers. By designing benchmarks that allow tool use, for example, providing an API or knowledge base as part of the test environment, we measure a more practical skill, the ability of an AI to know what it does not know and find out. This also helps combat hallucinations and data staleness, as seen in RAG approaches. In intersections, this addresses some contamination issues by relying on an external source rather than training memory. This in turn aligns with the goal of realistic evaluation, since human AI users often expect AI to cite sources or use web search. It is a shift from the old paradigm of closed-book QA towards an open-book evaluation model.

In particular, RAG evaluation highlights the importance of ground truth reference signals for factual tasks. In base LLM evaluation, this insight suggests incorporating open-book testing: instead of only closed-book QA, have benchmarks where the model can consult a knowledge source, as a form of RAG, and see if that boosts performance. If an LLM under closed-book conditions fails a question but succeeds when allowed to retrieve relevant text, that indicates the base model's limitation was missing knowledge, not reasoning ability. Conversely, if it fails even with the reference provided, the issue lies in understanding or reasoning. This differentiated evaluation, closed-book versus open-book, was historically done in QA research and can be informed by RAG frameworks. Gao et al. (2024) mention that RAG enables continuous knowledge updates and domain-specific info integration. Evaluating a base model in scenarios with and without such updates can quantify how much retrieval augments it. In summary, by borrowing RAG's metrics such as faithfulness, relevance, etc., and techniques such as LLM-based judging, multicomponent analysis, we can design more nuanced and robust evaluations for base models. This ensures that enhanced capabilities such as factual grounding are explicitly tested and that a model's score reflects not just whether it is right, but why and how it arrives at the answers.

3 Output Design

This framework provides a systematic, repeatable, and automated approach to benchmarking advisory LLMs across various domains. By integrating domain-specific knowledge, retrieval augmentation, and knowledge graphs, it ensures robust evaluations that align with real-world application needs. The modular design allows organizations to continuously update their benchmarking pipeline as domain requirements evolve, ensuring that advisory models remain accurate, trustworthy, and effective in real-world deployment. While demonstrated here with a cybersecurity compliance use case, the framework's architecture is intentionally domain-agnostic. The same methodology can be applied to financial advisory, legal consultation, healthcare guidance, or any other domain requiring specialized knowledge. This flexibility allows organizations to adapt the benchmarking process to their specific needs while maintaining rigorous evaluation standards (see Figure 3.1).

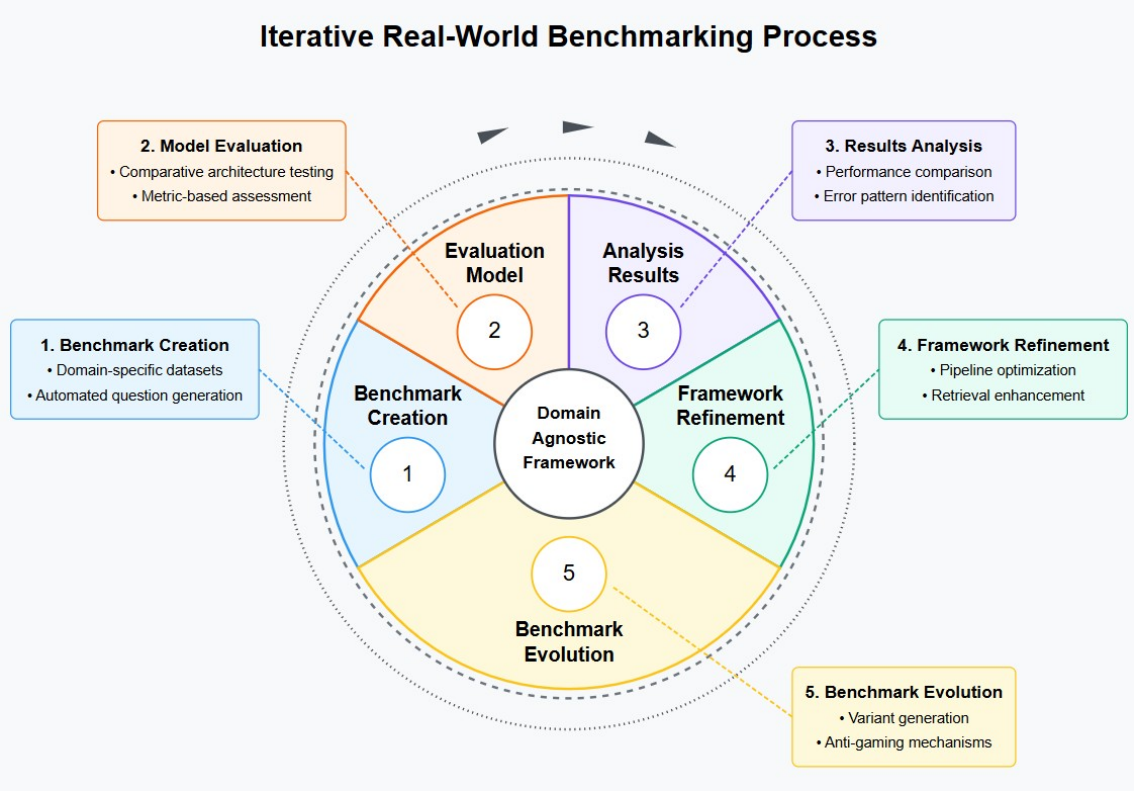

Figure 3.1: Depicts the cyclical nature of benchmarking with five phases: Benchmark Creation, Model Evaluation, Results Analysis, Framework Refinement, and Benchmark Evolution.

3.1 Methodologies

To develop an effective cybersecurity compliance advisory benchmarking framework, we employ a structured methodology that ensures real-world relevance and automation. The benchmark is designed around publicly available cybersecurity standards (e.g., **NIST 800-53, NIST Cybersecurity Framework (CSF) 2.0, CIS Controls, CSA Cloud Controls Matrix (CCM), GDPR, and MITRE ATT&CK**) and focuses on evaluating an LLM's ability to advise on compliance-related queries. This includes:

1. **Identifying Real-World Advisory Tasks** – Defining key use cases such as answering compliance-related questions, identifying gaps, providing policy recommendations, and cross-mapping standards (NIST 2020).

2. **Automating Benchmark Dataset Creation** – Extracting and structuring compliance questions from regulatory texts, case law, certification exams, and expert Q&A forums (McIntosh et al. 2024).

3. **Evaluating Multiple Architectures** – Comparing base models, fine-tuned models, RAG-enhanced models, and GraphRAG architectures (Xu et al. 2024).

4. **Continuous Refinement and Benchmark Evolution** – Preventing benchmark overfitting by generating test variants and monitoring for artificial performance inflation (Xia, Deng, & L. Zhang 2024). The framework addresses benchmark evolution through three practical mechanisms:

   (a) **Scheduled Resource Updates:** Periodic manual replacement of resource documents with the latest regulatory versions, recognizing that automated detection of domain knowledge evolution represents a complex research problem requiring continuous monitoring of regulatory bodies.

   (b) **Schema-Preserved Regeneration:** Maintaining consistent output schemas while regenerating questions with updated resource documents, producing differently worded questions with the same evaluative content to test genuine understanding versus memorization.

   (c) **Multi-Model Generation:** Using different LLMs (e.g., Gemma2:9b, Llama3.1, GPT-4) with identical schemas and resource documents to create diverse question formulations while ensuring benchmarks are not biased toward specific model families.

3.2 Software

The benchmarking framework is implemented with Docker and Ollama for seamless local model deployment and evaluation (Ollama 2024). Ollama enables running 7B-13B parameter open-source models on consumer GPUs, ensuring cost-effective testing. The TrustGraph framework is used for GraphRAG, leveraging knowledge graphs to enhance retrieval accuracy (TrustGraph 2024). Hugging Face's "evaluate" library provides standardized performance metrics, while Python-based scripts automate dataset curation, evaluation pipeline execution, and metric tracking. Additional tools include:

- **LangGraph** – For agent workflow orchestration and state management (LangGraph 2025).
- **LangChain** – For LLM-based retrieval, response generation, and logging interactions (LangChain 2024).
- **FAISS** – For vector-based document retrieval in standard RAG configurations (Douze et al. 2025).
- **Qdrant** – For knowledge graph storage and querying in GraphRAG experiments (Qdrant 2025).
- **Pydantic** – For dynamic schema generation and structured output validations (Pydantic 2025).

3.3 System

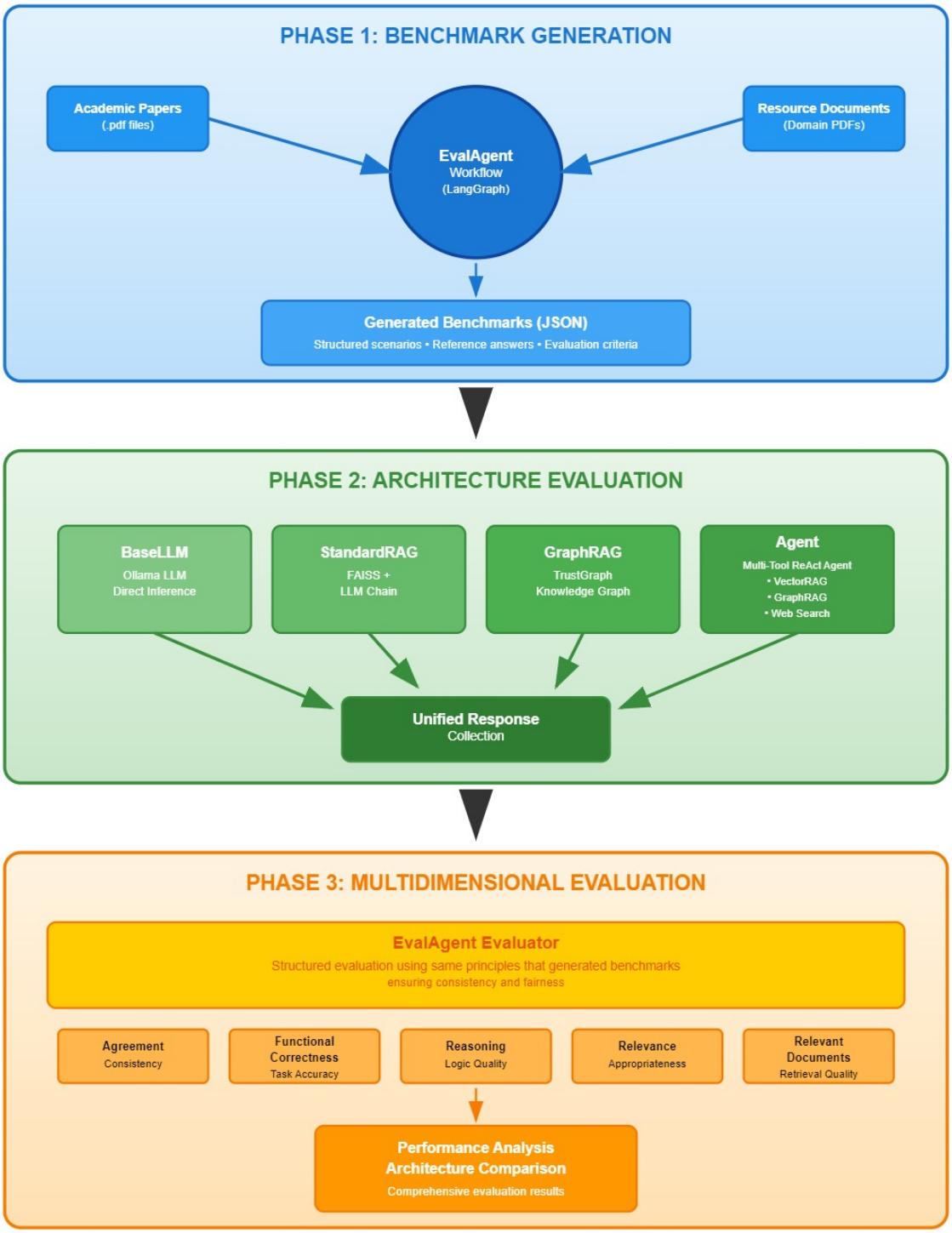

Figure 3.2: Domain-Bench framework architecture showing the complete three-phase evaluation process: (1) EvalAgent-driven benchmark generation using LangGraph workflows and multi-modal RAG, (2) systematic evaluation across four distinct model architectures, and (3) multidimensional assessment using five evaluation criteria.

The benchmarking system follows a modular three-phase pipeline (see Figure 3.2), comparing four LLM configurations (see Figure 3.3):

1. **Base LLM** - Direct model inference using Ollama (7B-13B open models).
2. **Standard RAG** - Vector-based retrieval augmentation using FAISS indexing.
3. **GraphRAG** - Knowledge graph-based retrieval using TrustGraph framework.
4. **Agent** - Multi-tool ReAct agent combining vector RAG, graph RAG, and web search capabilities.

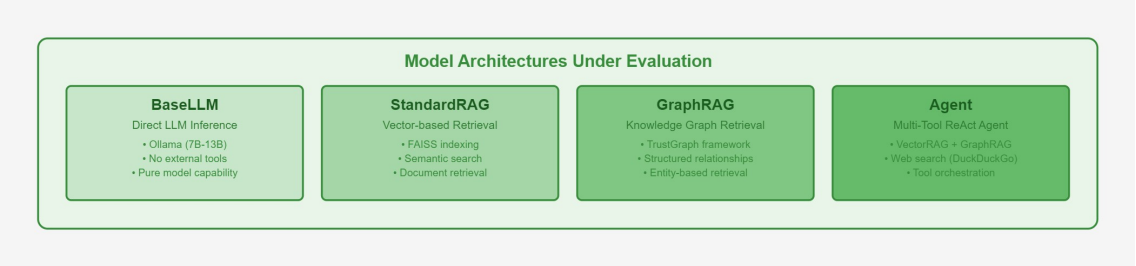

Figure 3.3: Four model architectures evaluated systematically: BaseLLM (direct inference), StandardRAG (vector-based retrieval), GraphRAG (knowledge graph retrieval using TrustGraph), and Agent (multi-tool ReAct agent combining multiple retrieval strategies).

Each architecture is deployed through a unified interface and evaluated on curated benchmarks of domain-specific queries. The pipeline executes queries across all architectures, retrieves relevant context (for RAG/GraphRAG/Agent models), logs outputs, and calculates performance metrics using the EvalAgent framework.

3.4 Resources

The implementation relies on publicly available compliance datasets and documentation:

- **NIST Special Publications (SP 800 series)**(NIST 2020)
- **NIST Cybersecurity Framework (CSF) 2.0** (NIST 2024)
- **CIS Critical Security Controls v8** (CIS 2023)
- **FedRAMP Security Controls & Compliance Guidelines** (FedRAMP 2023)
- **GDPR regulatory text and enforcement case studies** (EuropeanCommission 2023)

The system employs an automated EvalAgent framework that processes academic papers to extract evaluation principles, generates domain-specific system prompts, and creates benchmarks through multi-document RAG retrieval from compliance resources. (see Figure 3.4) (Krishna 2024).

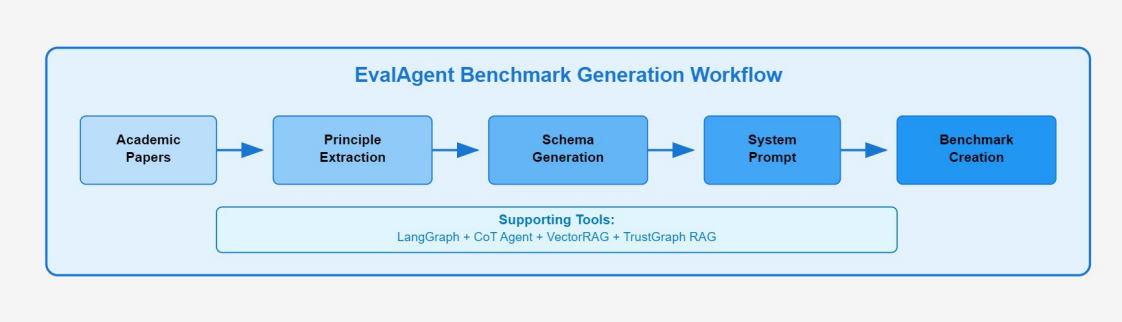

Figure 3.4: EvalAgent benchmark generation workflow showing the automated extraction of evaluation principles from academic papers through a five-step process: principle extraction, schema generation, system prompt creation, and benchmark generation using multi-modal RAG.

3.5 Evaluation Parameters

To measure the success of each implementation, the benchmarking framework (see Figure 3.5) evaluates models across multiple dimensions:

1. **Agreement** – Is there consistency between a model's predicted probabilities and its ability to generate coherent, relevant, and accurate text? High agreement implies that the model's probabilistic outputs accurately reflect its generative capabilities across diverse tasks, including those with both definitive solutions and open-ended inquiries. (Lyu, Wu, & Aji 2024)

2. **Function Correctness** – Does the answer reliably and accurately fulfill the intended function specified in a task or prompt? Does it properly address its real-world intended usage? (Xia, Deng, & L. Zhang 2024)

3. **Reasoning** – Can the model logically justify its recommendations? (Xu et al. 2024)

4. **Relevance** – Does the response directly address the compliance question? (Es et al. 2023)

5. **Retrieval Effectiveness (for RAG/GraphRAG)** – Are retrieved documents relevant and properly used? (Gao et al. 2024)

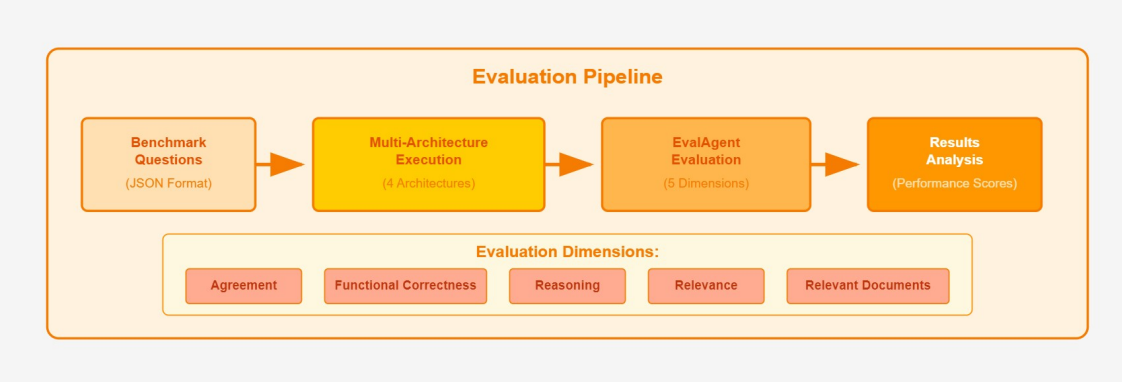

Figure 3.5: Evaluation pipeline demonstrating the systematic assessment process from benchmark questions through multi-architecture execution to comprehensive scoring across five evaluation dimensions.

Each model is tested on a standardized benchmark and its performance is logged across these metrics. The results are analyzed to determine:

- **Which model architecture performs best?**
- **Does retrieval improve reasoning and accuracy?**
- **Does GraphRAG reduce hallucinations and improve compliance adherence?**
- **Are improvements genuine or due to artificial benchmark gaming?** (Xia, Deng, & L. Zhang 2024)

If the best-performing model exhibits benchmark overfitting (e.g., memorization of test questions), the architecture is adjusted, re-tested, and iteratively refined (Rasiah et al. 2024).

3.6 Implementation Workflow

The Domain-Bench framework operates through an integrated workflow that combines automated benchmark generation with systematic architecture evaluation (as shown in the complete framework, Figure 3.2):

- **Principle Extraction:** Academic papers are processed to extract evaluation criteria using the EvalAgent's Chain-of-Thought reasoning capabilities.
- **Benchmark Generation:** Domain-specific questions are created through multi-modal RAG, combining vector search and knowledge graph retrieval from compliance resources.
- **Architecture Evaluation:** Each benchmark is executed across all four model configurations, with responses logged and contextualized.
- **Multidimensional Assessment:** The same EvalAgent framework that generated benchmarks evaluates responses across five dimensions, ensuring consistency and fairness.

This end-to-end automation enables scalable, reproducible evaluation while maintaining the flexibility to adapt to evolving domain requirements.

References

CIS (2023). *Critical Security Controls v8*. URL: https://www.cisecurity.org/controls/v8-1.

Dalvi, Fahim et al. (2024). *LLMeBench: A Flexible Framework for Accelerating LLMs Benchmarking*. arXiv: 2308.04945 [cs.CL]. URL: https://arxiv.org/abs/2308.04945.

Douze, Matthijs et al. (2025). *The Faiss library*. arXiv: 2401.08281 [cs.LG]. URL: https://arxiv.org/abs/2401.08281.

Es, Shahul et al. (2023). *RAGAS: Automated Evaluation of Retrieval Augmented Generation*. arXiv: 2309.15217 [cs.CL]. URL: https://arxiv.org/abs/2309.15217.

EuropeanCommission (2023). *GDPR Enforcement Cases: Official Public Reports*. Tech. rep. European Union. URL: https://gdpr.eu.

FedRAMP (2023). *Security Controls and Compliance Guidelines*. URL: https://www.fedramp.gov/documents-templates/.

Friel, Robert, Masha Belyi, & Atindriyo Sanyal (2025). *RAGBench: Explainable Benchmark for Retrieval-Augmented Generation Systems*. arXiv: 2407.11005 [cs.CL]. URL: https://arxiv.org/abs/2407.11005.

Gao, Yunfan et al. (2024). *Retrieval-Augmented Generation for Large Language Models: A Survey*. arXiv: 2312.10997 [cs.CL]. URL: https://arxiv.org/abs/2312.10997.

Golchin, Shahriar & Mihai Surdeanu (2024). *Time Travel in LLMs: Tracing Data Contamination in Large Language Models*. arXiv: 2308.08493 [cs.CL]. URL: https://arxiv.org/abs/2308.08493.

Krishna, Varun Badrinath (2024). *AttackQA: Development and Adoption of a Dataset for Assisting Cybersecurity Operations using Fine-tuned and Open-Source LLMs*. arXiv: 2411.01073 [cs.LG]. URL: https://arxiv.org/abs/2411.01073.

LangChain (2024). *Framework for developing applications powered by large language models*. URL: https://python.langchain.com/docs/introduction/.

LangGraph (2025). *Framework for building, managing, and deploying long running, stateful agents*. URL: https://langchain-ai.github.io/langgraph/.

Li, Jiarui, Ye Yuan, & Zehua Zhang (2024). *Enhancing LLM Factual Accuracy with RAG to Counter Hallucinations: A Case Study on Domain-Specific Queries in Private Knowledge-Bases*. arXiv: 2403.10446 [cs.CL]. URL: https://arxiv.org/abs/2403.10446.

Lyu, Chenyang, Minghao Wu, & Alham Fikri Aji (2024). *Beyond Probabilities: Unveiling the Misalignment in Evaluating Large Language Models*. arXiv: 2402.13887 [cs.CL]. URL: https://arxiv.org/abs/2402.13887.

McIntosh, Timothy R. et al. (2024). *Inadequacies of Large Language Model Benchmarks in the Era of Generative Artificial Intelligence*. arXiv: 2402.09880 [cs.AI]. URL: https://arxiv.org/abs/2402.09880.

NIST (Sept. 2020). *Security and Privacy Controls for Information Systems and Organizations*. NIST Special Publication 800-53 Rev. 5. Gaithersburg, MD: NIST. DOI: 10.6028/NIST.SP.800-53r5. URL: https://doi.org/10.6028/NIST.SP.800-53r5.

NIST (2024). *Cybersecurity Framework 2.0*. NIST Cybersecurity White Paper [CSWP] NIST CSWP 29. NIST. DOI: 10.6028/NIST.CSWP.29. URL: https://doi.org/10.6028/NIST.CSWP.29.

Ollama (2024). *Documentation for Local Model Deployment*. URL: https://github.com/ollama/ollama/tree/main/docs.

Pydantic (2025). *Data validation library for Python*. URL: https://docs.pydantic.dev/latest/.

Qdrant (2025). *Open source vector database and similarity search engine designed to handle high-dimensional vectors for performance and massive-scale AI applications*. URL: https://qdrant.tech/.

Rasiah, Vishvaksenan et al. (2024). *SCALE: Scaling up the Complexity for Advanced Language Model Evaluation*. URL: https://openreview.net/forum?id=aLXRYfIUUd.

Saad-Falcon, Jon et al. (2024). *ARES: An Automated Evaluation Framework for Retrieval-Augmented Generation Systems*. arXiv: 2311.09476 [cs.CL]. URL: https://arxiv.org/abs/2311.09476.

TrustGraph (2024). "Graph-Based Retrieval-Augmented Generation". In: *Technical Report*. URL: https://trustgraph.ai/docs/TrustGraph.

Xia, Chunqiu Steven, Yinlin Deng, & Lingming Zhang (2024). *Top Leaderboard Ranking = Top Coding Proficiency, Always? EvoEval: Evolving Coding Benchmarks via LLM*. arXiv: 2403.19114 [cs.SE]. URL: https://arxiv.org/abs/2403.19114.

Xu, Cheng et al. (2024). *Benchmark Data Contamination of Large Language Models: A Survey*. arXiv: 2406.04244 [cs.CL]. URL: https://arxiv.org/abs/2406.04244.