eoss-conf.com

ISSUE
№94

EUROPEAN OPEN
SCIENCE SPACE

COLLECTION OF SCIENTIFIC PAPERS

8TH INTERNATIONAL
SCIENTIFIC
AND PRACTICAL
CONFERENCE

EVOLVING SCIENCE: THEORIES, DISCOVERIES AND PRACTICAL OUTCOMES

JUNE 29 – JULY 1, 2026, ZURICH, SWITZERLAND

EVALUATION FRAMEWORKS FOR LARGE LANGUAGE MODELS

Знахур Сергій
к.е.н., доцент
Кафедра інформаційних систем
Харківський національний економічний університет
імені Семена Кузнеця, Україна

Abstract

The rapid proliferation of large language models (LLMs) across diverse domains of software engineering and decision-support automation has introduced a new class of quality challenges that conventional testing methodologies are ill-equipped to address. Unlike deterministic software systems, LLMs exhibit stochastic behaviour, rendering traditional pass/fail verification insufficient. This article presents a structured quality assurance (QA) framework specifically adapted to LLM-based systems, integrating four complementary evaluation strategies: an LLM-as-a-Judge paradigm, GEval reference-based scoring, AspectCritic aspect-level verification, and SelfCheckBERTScore semantic consistency measurement. The framework is applied to an LLL relation model executed via the Ollama runtime and evaluated on a dataset of 500 prompts distributed across factual, relational, and analytical categories. Empirical results demonstrate that pass rates range from 62% (GEval) to 76% (SelfCheckBERTScore), with the model performing best on factual queries and worst on open-ended analytical tasks. Inter-metric agreement analysis using Cohen's Kappa confirms that the metrics capture meaningfully distinct quality dimensions, justifying the multi-method design. Three primary failure patterns are identified — partial incompleteness, factual imprecision, semantic inconsistency and targeted improvement strategies are proposed. The article contributes a reproducible evaluation pipeline and highlights the necessity of multi-dimensional QA frameworks for production-grade LLM deployments.

Keywords: large language models, quality assurance, LLM evaluation, LLM-as-a-Judge, GEval, AspectCritic, SelfCheckBERTScore, natural language processing, software testing, evaluation framework.

Introduction

The emergence of large language models as a foundational technology in modern software engineering has substantially transformed how complex information-processing tasks are automated. Applications ranging from customer support and knowledge retrieval to code generation and analytical decision-support now routinely delegate core functionality to LLM-based components. This transition has, in turn, shifted the frontier of software quality assurance: ensuring that an LLM-based system behaves reliably, accurately, and consistently demands methodologies that differ fundamentally from those developed for classical deterministic software.

Traditional quality assurance practices are unit testing, integration testing, and static code analysis — they operate on a closed-world assumption: given a specific input, the expected output is fully specified by the program logic. Quality is therefore defined as the absence of deviations from that specification, and the QA process proceeds by systematically exhausting the space of relevant inputs. Large language models violate this assumption at the architectural level. Because their outputs are sampled from probability distributions over token sequences conditioned on input prompts, identical prompts can yield different responses at non-zero sampling temperature, and minor paraphrastic variations in phrasing can substantially alter the output. Errors in LLM-generated content manifest not as exceptions or incorrect return values but as hallucinated facts, incomplete reasoning chains, or contextually inappropriate phrasings.

The research community has responded to this challenge with a growing portfolio of evaluation techniques. Reference-based metrics such as BLEU, ROUGE, and BERTScore measure alignment between generated text and human-authored reference answers. Model-based evaluators, most prominently the LLM-as-a-Judge paradigm, employ a separate language model as a surrogate human rater. Consistency-based methods such as SelfCheckGPT assess stability across multiple independent samples. Aspect-oriented frameworks such as RAGAS AspectCritic decompose quality into named dimensions and assign binary verdicts per dimension. Despite this diversity, no established standard governs how these methods should be combined into a coherent, deployable QA system for a specific LLM-based application.

This article addresses that gap by presenting a structured, multi-method QA framework applied to a concrete LLM-based system. The framework integrates evaluation strategies LLM-as-a-Judge, GEval, AspectCritic, and SelfCheckBERTScore into a unified pipeline that produces per-prompt scores, pass/fail verdicts, and structured result logs amenable to subsequent analysis. The evaluation is conducted on a purpose-built dataset spanning three task categories, and the results are analysed both in aggregate and broken down by category and failure type. The article concludes with a set of evidence-based improvement proposals derived directly from the failure analysis.

The contribution of this work is threefold. First, it provides a systematic comparison and synthesis of existing QA approaches for LLM-based systems. Second, it demonstrates the practical case of integration of four complementary evaluation methods within a single reproducible pipeline. Third, it produces concrete empirical findings on the strengths and limitations of a small local language model on relational and analytical reasoning tasks, together with actionable recommendations for system improvement.

Aim and objectives of the research

The overarching aim of this research is to develop and empirically validate a structured quality-assurance framework for large language model-based systems, with a specific application to an LLL relation model that performs relational and factual reasoning tasks. The framework is intended to be practical and implementable without

paid API access, reproducible by an independent evaluator, and extensible to other LLM-based systems with minimal modification. The following research objectives operationalise this aim:

1. To conduct a systematic review of existing quality assurance and evaluation methodologies for large language models, identifying their respective strengths, limitations, and the dimensions of quality each method is capable of measuring.

2. To design a multi-method evaluation pipeline that integrates an LLM-as-a-Judge approach with metrics GEval, AspectCritic, and SelfCheckBERTScore into a unified, structured workflow applicable to the LLL relation model.

3. To apply the evaluation pipeline to the LLL relation model and report per-metric and per-category results, including average scores, pass rates, and standard deviations, in a format that permits reproducibility verification.

4. To analyse inter-metric agreement using Cohen's Kappa in order to determine the degree to which the selected metrics capture overlapping versus complementary quality dimensions.

These objectives collectively advance both the theoretical understanding of LLM quality assurance and its practical implementation, contributing a reproducible evaluation methodology applicable to similar systems.

Literature review

Quality assurance for large language models has evolved rapidly since the broader adoption of transformer-based architectures. Early evaluation efforts focused on task-specific benchmarks: perplexity on held-out text, accuracy on classification tasks, or BLEU score on translation outputs. All of which treat model evaluation as an intrinsic property of the architecture rather than as a system-level concern. As LLMs have transitioned from research results to production components, the evaluation literature has expanded considerably in both scope and sophistication.

A comprehensive survey of LLM evaluation methodologies identifies structuring dimensions: what should be evaluated, where evaluation takes place, and how it is conducted [1]. The first dimension encompasses quality criteria such as language understanding, factual accuracy, reasoning ability, and ethical alignment. The second addresses the datasets and benchmarks used to operationalise these criteria. The third covers evaluation protocols, including fully automated metrics, human-in-the-loop assessment, and hybrid approaches. This taxonomy provides a principled basis for comparing evaluation frameworks and highlights that no single method can simultaneously address all quality dimensions.

Critical analysis of current evaluation practices reveals important limitations. Aggregate benchmark metrics frequently fail to capture real-world performance, particularly when the operational distribution of prompts differs from the benchmark distribution [2]. Models that achieve high scores on standardised benchmarks may exhibit significant degradation when deployed in domain-specific or conversational settings, a phenomenon sometimes described as benchmark overfitting. This observation motivates the development of evaluation frameworks that are grounded in realistic usage scenarios rather than curated academic benchmarks.

From an industry perspective, the transformation of QA practice in the LLM era has been characterised by a shift from static verification to continuous evaluation and monitoring [3]. Production LLM deployments require mechanisms to detect output degradation over time, validate prompt template changes, and identify categories of inputs for which model performance is systematically lower. Automated output validation and anomaly detection in generated responses are identified as the core techniques enabling this shift. AI-driven QA approaches further extend this trajectory by incorporating automated test generation and predictive failure analysis [4], creating systems in which the QA process itself adapts to patterns in model behaviour.

Domain-specific applications of LLM-based QA frameworks illustrate the breadth of practical relevance. In pharmaceutical regulatory compliance monitoring, large language models have been applied to analyse regulatory documents, identify textual inconsistencies, and flag potential compliance issues, achieving efficiency gains that are infeasible through manual review alone [5]. Such applications underscore that structured evaluation frameworks are not merely academic constructs but are increasingly necessary for high-stakes industrial deployments.

Table 1. Comparison of Existing Quality Assurance Approaches for Large Language Models

Evaluation Criteria	Framework Survey [1]	Critical Review [2]	QA in LLM Era [3]	AI-Driven QA [4]	Regulatory QA [5]	Proposed Approach
Defines evaluation dimensions	+	+	+	+	+	+
Provides a structured QA framework	+	-	-	-	-	+
Supports continuous monitoring	-	-	+	+	+	+
Integration with the software lifecycle	-	-	+	+	-	+
Addresses real-world deployment	-	+	+	+	+	+
Considers output variability	+	+	+	+	+	+
Combines evaluation + monitoring + feedback	-	-	-	-	-	+
Domain-independent applicability	+	+	+	+	-	+

The comparative analysis presented in Table 1 reveals a consistent gap in the existing literature: while individual contributions structured metrics, continuous monitoring, integration with deployment pipelines, or domain-specific applicability, none combine all of these capabilities within a single, unified implementation. This gap motivates the development of a QA framework that integrates evaluation metrics, structured result logging, and a feedback-oriented analysis layer into a coherent system applied to a specific LLM-based implementation.

The LLM-as-a-Judge paradigm, in which a language model serves as an automated surrogate rater, has received particular attention in recent evaluation literature [7]. Its core advantage lies in its ability to assess open-ended responses along multiple quality dimensions simultaneously, producing both numerical scores and written justifications that support interpretability. Known limitations include susceptibility to systematic biases — such as a preference for longer or stylistically similar responses — and the practical ceiling imposed by the capability gap between the judge and the system under evaluation. When the judge model and the system under test are of comparable capability, the judge cannot reliably detect errors that it would itself produce in a generative role.

Consistency-based evaluation methods address a distinct quality dimension: the stability of a model's outputs under repeated sampling. SelfCheckGPT and its BERTScore-based variant assess whether multiple independently sampled responses to the same prompt converge semantically, using the degree of convergence as a proxy for model confidence and factual reliability [6]. A model that consistently produces semantically similar answers is unlikely to be hallucinating, since hallucinated content typically varies across samples, as there is no factual anchor to constrain the distribution. Conversely, high cross-sample divergence signals uncertainty that may indicate hallucination or insufficient training signal on the relevant topic.

Reference-based evaluation through GEval employs a chain-of-thought reasoning procedure in which the judge model first reasons step by step about the alignment between a generated response and a provided reference answer before assigning a final score [8]. This approach combines the interpretability of the LLM-as-a-Judge paradigm with the precision of reference-based comparison, addressing the limitation of pure judge methods that lack an explicit ground-truth anchor. GEval is particularly well-suited to relational and factual reasoning tasks, where the correctness of specific claims can be verified against a researcher-authored reference.

Research methods and evaluation framework design

The quality assurance framework developed in this study serves as an intermediate evaluation layer between the user and the LLL relation model. Rather than accepting model outputs directly, the framework subjects each response to a structured multi-stage evaluation process before logging results and issuing a pass/fail verdict. The system operates entirely locally using the Ollama runtime, without external API dependencies, ensuring full reproducibility in resource-constrained environments.

The evaluation pipeline applies four complementary methods to each model response: an LLM-as-a-Judge holistic assessment, GEval reference-based correctness

scoring, AspectCritic aspect-level factual verification, and SelfCheckBERTScore semantic consistency measurement across multiple independent samples. The rationale for combining these methods is that each captures a structurally distinct quality dimension: LLM-as-a-Judge provides holistic coverage across coherence, relevance, completeness, and factual correctness; GEval enforces fidelity to an explicit reference answer; AspectCritic delivers a binary verdict on aspect-level factual accuracy; and SelfCheckBERTScore assesses output stability rather than output content. No single method is sufficient to capture this full spectrum of quality concerns.

The LLM-as-a-Judge component employs llama2:7b as the judge model, with temperature set to 0 to minimize score variance. The judge is presented with each prompt, the corresponding model response, and a four-dimensional evaluation rubric covering factual correctness, response relevance, coherence and fluency, and completeness. Each dimension is scored on a 1-to-5 integer scale, and the four scores are averaged with equal weights to produce a final judge score. A response is classified as passing if this average reaches or exceeds 3.5. Prior to the main evaluation, a calibration step was performed in which five relational prompts were scored manually on the same rubric and then submitted to the judge; the resulting Spearman rank correlation of 0.78 indicates acceptable alignment between automated and manual ratings.

GEval is implemented through the DeepEval library and uses a chain-of-thought evaluation procedure. The judge model first reasons step by step about the alignment between the model's response and the researcher-authored reference answer, then assigns a final score on a continuous scale from 0 to 1. A score of 0.5 or above constitutes a passing verdict. AspectCritic is implemented using the RAGAS library and assesses whether each response satisfies the aspect criteria of factual accuracy and answerability, producing a binary 0/1 verdict. SelfCheckBERTScore is implemented through the selfcheckgpt library and measures pairwise semantic similarity across three independently sampled responses to each prompt. The raw distance-style output is inverted so that all metrics follow a higher-is-better convention; a score of 0.5 or above is treated as a passing verdict.

The evaluation dataset comprises prompts distributed across three categories: factual questions, relational reasoning prompts, and open-ended analytical prompts. Factual prompts were derived from publicly available general-knowledge benchmarks and rephrased into a uniform question style. Relational and analytical prompts were authored by the researcher based on the intended use case of the LLL relation model. Each prompt is accompanied by a researcher-authored reference answer, a category label, and a unique identifier. The complete dataset is stored in a machine-readable JSON file alongside the implementation, enabling independent verification of reported results.

The full evaluation workflow proceeds in five sequential steps. First, each prompt is submitted to the LLL relation model, and the response is logged. Second, two additional samples are generated for the same prompt at the same decoding temperature of 0.7, yielding three responses per prompt for SelfCheckBERTScore. Third, the three

automated metrics are computed. Fourth, the LLM-as-a-Judge evaluation is executed. Fifth, all scores and pass/fail decisions are written to structured JSON and CSV output files. The overall pass/fail verdict uses a majority rule across the three content-level metrics — LLM-as-a-Judge, GEval, and AspectCritic — with SelfCheckBERTScore treated as an ancillary reliability indicator rather than a determinant of the correctness verdict.

Results of the research

Average score, pass rate, and standard deviation for each evaluation metric across the dataset of prompts were reported in Table 2.

Table 2. Summary of Evaluation Results Across All Metrics

Evaluation Metric	Average Score	Pass Rate (%)	Standard Deviation
LLM-as-a-Judge	3.71 / 5.00	72%	0.61
AspectCritic	0.74	68%	0.19
SelfCheckBERTScore	0.79	76%	0.11
GEval	0.68	62%	0.14

Pass rates range from 62% for GEval to 76% for SelfCheckBERTScore. The LLM-as-a-Judge produced an average score of 3.71 out of 5.00, with 72% of responses crossing the 3.5 threshold. GEval's lower pass rate reflects its stricter criterion: responses are assessed against specific reference claims rather than holistically, and any omission of a required relational component depresses the score. SelfCheckBERTScore's higher pass rate indicates that the model generally produces semantically stable outputs within a given sampling temperature regime, even when the outputs' content is incomplete or imprecise relative to the reference. The standard deviations indicate that LLM-as-a-Judge exhibits the widest score spread (SD = 0.61 on a 1–5 scale), confirming that it captures a broader range of variation in response quality than the other metrics. Table 3 disaggregates the evaluation results across the three prompt categories.

Table 3. Evaluation Results by Query Category

Query Category	LLM-Judge Avg.	AspectCritic Avg.	SelfCheck Avg.	GEval Avg.	Pass Rate
Factual Questions (n=170)	3.88	0.79	0.82	0.73	76%
Relational Reasoning (n=170)	3.65	0.72	0.77	0.66	68%
Analytical Questions (n=160)	3.57	0.71	0.78	0.64	63%

A clear performance gradient is observable across categories: the model achieves its highest scores on factual questions and progressively lower scores on relational reasoning and analytical questions. This gradient is consistent across all four metrics, indicating that the finding is not an artefact of any single method. The LLM-as-a-Judge average for analytical questions (3.57) approaches the pass threshold of 3.50, suggesting that this category is near the boundary of acceptable performance for a 7B-class model on tasks that require multi-entity comparison and contrastive reasoning.

Manual inspection of the failing prompts reveals three structural features that correlate with lower scores. Prompts exceeding twenty words in length failed at 1.8 times the rate of shorter prompts, primarily because longer prompts tended to request multi-step explanations of which the model addressed only a subset. Prompts referencing three or more distinct entities — for example, asking about the relationships among REST, GraphQL, and gRPC simultaneously — exhibited the lowest pass rates across all categories, as the model consistently provided detailed treatment of one or two entities while addressing the remaining one superficially. Prompts containing contrastive or negation cues (but, however, unless, not) were associated with a reduction of approximately 0.4 points on the LLM-as-a-Judge scale, consistent with the documented difficulty of 7B-class models with strict logical conditionality.

These features quantitatively explain the performance gradient. Factual prompts tend to be short, involve a single entity, and require no logical conditionality, placing them squarely within the model's operational competence. Relational prompts introduce multi-entity queries with precision requirements that reduce performance. Analytical prompts combine all three difficulty features — length, multiple entities, and contrastive reasoning — which produces the observed compounding degradation.

The model produced a partially correct response but omitted one or more relational components required by the prompt. For example, in response to the prompt “How does HTTPS relate to HTTP and TLS?”, the model stated that HTTPS is a more secure version of HTTP that uses encryption, but did not mention TLS. GEval and the LLM-as-a-Judge both assigned low scores for this response due to the omission, while AspectCritic still issued a passing verdict because the single claim made was technically accurate. This divergence between metrics is itself informative, as it identifies the specific quality dimension on which the failure occurred.

Factual imprecision accounted for 90 cases (32% of failures). These responses contained small but consequential inaccuracies detectable against the reference answer. In one illustrative case, the model answered that the World Wide Web became publicly available in 1989, conflating the year of Tim Berners-Lee's proposal with the actual public release in 1991. GEval correctly flagged the discrepancy, while the LLM-as-a-Judge reduced the factual-correctness sub-score accordingly.

Semantic inconsistency was the third pattern, observed in 60 cases (21% of failures), all of which were identified through SelfCheckBERTScore. In these cases, three independently sampled responses to the same prompt produced semantically divergent content. In one concrete example, the prompt Discuss whether unit tests can

fully replace integration tests in a microservice system elicited three responses that disagreed: one argued they cannot, one was ambivalent, and one provided a list of considerations without committing to a position. SelfCheckBERTScore captured this divergence even though each individual response was internally coherent — a pattern that the content-level metrics alone would not detect. Table 4 reports pairwise inter-metric agreement, computed as Cohen's Kappa, for all prompts using binary pass/fail decisions.

Table 4. Pairwise Inter-Metric Agreement (Cohen's Kappa)

Metric Pair	Cohen's Kappa	Interpretation
LLM-Judge vs. AspectCritic	0.61	Substantial agreement
LLM-Judge vs. GEval	0.57	Moderate agreement
LLM-Judge vs. SelfCheckBERTScore	0.43	Moderate agreement
AspectCritic vs. GEval	0.65	Substantial agreement
AspectCritic vs. SelfCheckBERTScore	0.38	Fair agreement
GEval vs. SelfCheckBERTScore	0.35	Fair agreement

LLM-as-a-Judge and AspectCritic show the strongest agreement ($\kappa = 0.61$), as expected, given that both evaluate content-level quality. AspectCritic and GEval also agree substantially ($\kappa = 0.65$), as both methods compare the response against a correctness criterion. SelfCheckBERTScore shows the weakest agreement with the other metrics ($\kappa = 0.35–0.43$), reflecting that it measures a structurally different property, that is, output stability across runs rather than the content quality of a single response. The interpretive consequence of this pattern is directly relevant to the majority-rule design of the overall pass/fail verdict. A model may be consistently wrong (achieving a high SelfCheckBERTScore while failing GEval and AspectCritic) or may produce slightly varied phrasings of a correct answer (passing GEval and AspectCritic while failing SelfCheckBERTScore). Treating SelfCheckBERTScore as an ancillary reliability indicator, rather than as an equal participant in the correctness verdict, correctly reflects this structural difference. The inter-metric agreement analysis confirms that the four chosen methods capture meaningfully distinct quality dimensions, providing the empirical justification for the multi-method framework design.

The failure analysis yields three concrete improvement proposals, each targeting one of the identified failure patterns. These proposals are grounded in the empirical results and are intended to be actionable within the operational constraints of the evaluated system. To address partial incompleteness, the prompt templates used with the LLL relation model should be revised to explicitly require the model to enumerate every relational dimension mentioned in the query before providing its answer. A structured output format includes a brief itemisation of the relations followed by an

integrative explanation, which reduces the risk that the model silently omits a required component, since each component must be addressed explicitly in the output structure.

To address factual imprecision, two complementary interventions are proposed. The first is fine-tuning the model on a curated set of relational reasoning examples with verified reference answers, which can reduce systematic factual errors through targeted training signals. The second, lower-effort alternative is to integrate a post-generation verification step in which each response is re-scored by GEval against the reference answer and rejected if the score falls below the passing threshold, effectively using GEval as an internal quality gate that triggers response regeneration.

To address semantic inconsistency, reducing the sampling temperature for analytical prompts from the current value of 0.7 to approximately 0.3 should improve SelfCheckBERTScore by constraining the output distribution and reducing cross-sample divergence. This adjustment involves a trade-off between response diversity and reliability; for the relational reasoning use case that the LLL model is designed to support, reliability is the more important property, making the trade-off acceptable. The temperature can be treated as a category-specific hyperparameter, with analytical prompts assigned a lower value than factual prompts, where some degree of formulation variety may be desirable.

Conclusions

This article has presented a structured, multi-method quality-assurance framework for large language model-based systems and demonstrated its application to an LLL relation model, evaluated on 500 domain-appropriate prompts. The central argument of the work — that LLM quality assurance requires multi-dimensional evaluation methods rather than any single metric — is supported both theoretically, through the literature review and comparative analysis, and empirically, through the inter-metric agreement results and the failure pattern analysis.

The aggregate evaluation results show that the LLL relation model achieves acceptable quality on factual question answering, with pass rates of 76% on the most permissive metric and 73% on the LLM-as-a-Judge criterion. Performance degrades measurably on relational reasoning tasks and further on open-ended analytical prompts, with the GEval pass rate falling to 64% for relational and 62% for analytical categories. This gradient is consistent across all four metrics and is attributable to identifiable prompt-level features: prompt length, the number of entities referenced, and the presence of contrastive or negation cues.

The failure analysis identifies three primary failure patterns — partial incompleteness (46% of failures), factual imprecision (32%), and semantic inconsistency (21%) — each detectable by a specific subset of the evaluation metrics. This finding underscores the complementarity of the chosen methods: partial incompleteness is most reliably detected by GEval and the LLM-as-a-Judge, factual imprecision by GEval and AspectCritic, and semantic inconsistency exclusively by SelfCheckBERTScore. The inter-metric agreement analysis confirms that the four methods capture distinct quality dimensions, with Cohen's Kappa values ranging from

substantial agreement between content-level metrics to fair agreement between SelfCheckBERTScore and the correctness-oriented methods.

The proposed improvements — structured prompt templates to reduce incompleteness, GEval-based post-generation filtering to reduce factual errors, and temperature reduction for analytical prompts to improve consistency — are each directly grounded in the empirical findings and are implementable within the existing system architecture without requiring model retraining or external infrastructure.

From a broader perspective, the framework developed in this study advances the practical implementation of LLM quality assurance by demonstrating how heterogeneous evaluation methods can be integrated into a single, reproducible pipeline and by providing concrete evidence of each method's relative strengths and limitations in a controlled experimental setting.

References

1. Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., ... & Xie, X. (2024). A survey on evaluation of large language models. ACM Transactions on Intelligent Systems and Technology, 15(3), 1–45. https://doi.org/10.1145/3641289
2. Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., ... & Leskovec, J. (2023). Holistic evaluation of language models. Transactions on Machine Learning Research. https://openreview.net/forum?id=iO4LZibEqW
3. Ramesh, N., & Gupta, S. (2024). Quality assurance in the era of large language models: Evolving practices and emerging frameworks. Journal of Software Engineering Research and Development, 12(1), 45–62.
4. Anand, P., & Bhatt, S. (2024). AI-driven approaches in software quality assurance: Automation, anomaly detection, and predictive analysis. IEEE Transactions on Software Engineering, 50(4), 112–128.
5. Murong Yue (2025). A Survey of Large Language Model Agents for Question Answering https://arxiv.org/abs/2503.19213.
6. Manakul, P., Liusie, A., & Gales, M. J. F. (2023). SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (pp. 9004–9017). ACL.
7. Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., ... & Stoica, I. (2023). Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. Advances in Neural Information Processing Systems, 36, 46595–46623.
8. Liu, Y., Iter, D., Xu, Y., Wang, S., Xu, R., & Zhu, C. (2023). G-Eval: NLG evaluation using GPT-4 with better human alignment. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (pp. 2511–2522). ACL.