A Benchmarking Framework for Detecting and Reducing Hallucinations in Large Language Models

Prof. S. Abikayil Aarthi
Assistant Professor,
Department of Computer Science and Engineering, Kings College of Engineering, Punalkulam, TN, India
aarthi.cse@kingsengg.edu.in

Hariharan M
Department of Computer Science and Engineering, Kings College of Engineering, Punalkulam, TN, India
hariharanm1802@gmail.com

Akash K
Department of Computer Science and Engineering, Kings College of Engineering, Punalkulam, TN, India
akashkaliyamoorthy4@gmail.com

Gunanathan G S
Department of Computer Science and Engineering, Kings College of Engineering, Punalkulam, TN, India
gunanathan1827@gmail.com

Akash R
Department of Computer Science and Engineering, Kings College of Engineering, Punalkulam, TN, India
mailtor.akask@gmail.com

U. Kalanithi
Department of Computer Science and Engineering, Kings College of Engineering, Punalkulam, TN, India
06kalanithi@gmail.com

Karan K
Department of Computer Science and Engineering, Kings College of Engineering, Punalkulam, TN, India
karankannan705@gmail.com

ABSTRACT

Large Language Models (LLMs) have become central to question answering, summarization, coding assistance, and conversational systems, yet they frequently produce hallucinations: fluent, confident outputs that are factually incorrect, unsupported by the provided context, or internally inconsistent. Hallucinations undermine user trust and pose serious risks when LLMs are deployed in high-stakes domains such as healthcare, law, finance, and education. Existing evaluation practice is fragmented: individual benchmarks target a single task or a single type of error, detection methods are evaluated on incompatible datasets and metrics, and mitigation techniques are rarely compared under a common protocol. This paper proposes a unified benchmarking framework for both detecting and reducing hallucinations in LLMs. The framework defines a hallucination taxonomy (factual, contextual-faithfulness, logical, fabricated-reference, and instruction-inconsistency errors), a multi-task benchmark composition covering closed-book question answering, knowledge-grounded question answering, summarization, dialogue, domain-specific, false-premise, and adversarial prompts, and a structured annotation pipeline based on atomic claim extraction and evidence-based verification. Six representative detection approaches are examined and compared conceptually: token-probability and entropy scoring, sampling-based self-consistency, semantic entropy, natural language inference (NLI) verification, retrieval-augmented fact-checking, and LLM-as-a-judge evaluation. Five mitigation strategies are likewise considered: prompt engineering, retrieval-augmented generation (RAG), chain-of-verification, decoding and fine-tuning interventions, and confidence-based abstention. Evaluation metrics including hallucination rate, factual precision, detection precision, recall, F1-score, AUROC, expected calibration error, and relative hallucination reduction are specified as the basis for comparison. As no models were executed and no annotated dataset was collected for this study, the work is presented as a proposed framework and illustrative experimental design rather than a report of empirical results, and all performance figures shown are clearly marked as illustrative placeholders. The discussion highlights the practical significance of the framework for model developers, deployers, and regulators, and outlines its limitations and the future directions needed to move from a proposed protocol to a maintained public benchmark.

Keywords: *Large Language Models, Hallucination Detection, Hallucination Mitigation, Benchmarking, Factuality, Retrieval-Augmented Generation, Natural Language Processing, Trustworthy AI*

I. INTRODUCTION

A. Background of Large Language Models

Large Language Models are neural networks, typically based on the Transformer architecture, that are pretrained on very large text corpora to predict the next token in a sequence and are then adapted through instruction tuning and reinforcement learning from human feedback [24]. These models now generate human-quality text across a wide range of tasks, including answering questions, summarizing documents, translating languages, writing code, and acting as conversational assistants. Their fluency and generality have led to rapid adoption in search engines, customer-support systems, productivity tools, and domain-specific assistants for medicine, law, and education. However, the training objective rewards plausible continuations rather than verified truth, and the model has no built-in mechanism for distinguishing what it knows from what it is merely able to phrase convincingly.

B. The Problem of Hallucination

In natural language generation, the term hallucination refers to generated content that is nonsensical, unfaithful to the provided source, or factually incorrect [1]. In LLMs, hallucinations appear in several forms: a fabricated date, statistic, or biography; a summary that contradicts its source document; a citation to a paper that does not exist; a reasoning chain whose conclusion does not follow from its steps; or an answer that ignores an explicit instruction or constraint. Because these errors are delivered in the same confident, fluent style as correct answers, users often cannot tell them apart without independent verification. Surveys of the field identify hallucination as one of the most significant barriers to the safe deployment of LLMs [1], [2], [3].

C. Limitations of Current Hallucination Evaluation

Hallucination evaluation remains fragmented. Many benchmarks focus on a single task, such as open-domain question answering [6] or summarization [5], [17], and therefore measure only one type of error. Detection methods are frequently reported on different datasets with different labelling conventions, making their published scores difficult to compare. Traditional text-overlap metrics such as ROUGE and BLEU correlate poorly with factual

correctness, because an output can share many words with a reference while still asserting a false fact [5]. Human evaluation is more reliable but is slow, expensive, and inconsistent across annotators. Finally, mitigation techniques are usually evaluated in isolation, so it is unclear how much hallucination each technique removes, at what computational cost, and with what effect on the helpfulness of the model's answers.

D. Role of Benchmarking

A benchmark provides a fixed set of inputs, reference information, labelling rules, and metrics against which different systems can be compared fairly and reproducibly. In the context of hallucination, a well-designed benchmark must do three things simultaneously: elicit hallucinations across diverse tasks and error types, provide ground truth or evidence against which each generated claim can be verified, and define metrics that capture both how often a model hallucinates and how well a detector or mitigation method responds. A unified benchmark therefore turns hallucination from an anecdotal observation into a measurable, comparable property of a system.

E. Research Problem

Despite rapid progress in both hallucination detection and mitigation, there remains a need for a clearly specified, end-to-end framework that describes how benchmark data should be composed across tasks and error types, how model responses should be annotated at the level of individual claims, and how detection and mitigation methods should be evaluated together under common metrics. Existing studies typically address one of these components in isolation, leaving a gap for a framework that links taxonomy, dataset, annotation, detection, mitigation, and evaluation into a single reproducible protocol.

F. Research Objectives

1. To review existing literature on hallucination in LLMs, including taxonomies, benchmark datasets, detection methods, and mitigation techniques, and to identify gaps.
2. To define a hallucination taxonomy suitable for consistent annotation across tasks.
3. To design a multi-task benchmark composition and a claim-level annotation pipeline.
4. To describe and conceptually compare six representative hallucination detection approaches.
5. To describe and compare five hallucination mitigation strategies under a common protocol.
6. To specify evaluation metrics for both detection quality and mitigation effectiveness.
7. To identify practical limitations and future research directions for deploying such a benchmark.

G. Contributions of the Study

- A unified benchmarking framework that evaluates hallucination detection and hallucination mitigation together, rather than as separate problems.
- A five-category hallucination taxonomy with operational definitions and detection signals for each category.
- A structured comparison of six detection approaches and five mitigation strategies with respect to model access requirements, cost, and suitability.
- A literature comparison table summarizing the methods, data, and limitations of prior hallucination benchmarks and detectors.
- An explicit, clearly labelled experimental and evaluation design that can be populated with real model outputs in future work, avoiding unsupported claims of empirical accuracy.

II. LITERATURE REVIEW

This section reviews prior work on hallucination taxonomies, benchmark datasets, detection methods, and mitigation techniques, and identifies the gap that the proposed framework addresses.

A. Surveys and Taxonomies of Hallucination

Ji et al. [1] provided one of the first comprehensive surveys of hallucination in natural language generation, distinguishing intrinsic hallucinations, which contradict the source, from extrinsic hallucinations, which cannot be verified from the source, and reviewing metrics and mitigation methods across summarization, dialogue, and translation. Huang et al. [2] extended this analysis specifically to LLMs, proposing a taxonomy that separates factuality hallucinations from faithfulness hallucinations and tracing their causes to training data, training procedures, and inference. Zhang et al. [3] surveyed hallucination in LLMs with emphasis on evaluation and mitigation, and Tonmoy et al. [4] catalogued more than thirty mitigation techniques, grouping them by whether they act on prompts, retrieval, decoding, or model training.

B. Hallucination Benchmarks and Datasets

Maynez et al. [5] conducted a large-scale human evaluation of abstractive summarization systems and showed that a substantial portion of generated summaries contained hallucinated content, and that standard overlap metrics failed to reflect this. Lin et al. [6] introduced TruthfulQA, a question-answering benchmark designed so that imitating common human misconceptions produces false answers, and found that larger models were not necessarily more truthful. Li et al. [7] introduced HaluEval, a large collection of generated and human-annotated hallucinated samples covering question answering, dialogue, and summarization, and used it to test whether LLMs can recognise hallucinations. Thorne et al. [8] created FEVER, a dataset of claims labelled as supported, refuted, or not having enough information against Wikipedia evidence, which has become a foundation for claim verification research.

C. Hallucination Detection Methods

Min et al. [9] proposed FActScore, which decomposes long-form generations into atomic facts and measures the proportion supported by a knowledge source, and Wei et al. [10] extended this idea using search-augmented evaluation

of long-form factuality. Manakul et al. [11] introduced SelfCheckGPT, a zero-resource black-box approach that samples multiple responses and treats inconsistency among them as evidence of hallucination. Farquhar et al. [12], building on Kuhn et al. [13], proposed semantic entropy, which clusters sampled answers by meaning and measures uncertainty over meanings rather than token sequences. Kadavath et al. [14] showed that LLMs can be prompted to estimate the probability that their own answers are correct, and Azaria and Mitchell [15] showed that a classifier trained on a model's internal activations can predict the truthfulness of its statements. Honovich et al. [16] and Laban et al. [17] evaluated natural language inference-based methods for factual consistency, and Chern et al. [18] proposed FacTool, a tool-augmented framework that uses search engines and code execution to verify claims.

D. Hallucination Mitigation Methods

Lewis et al. [19] introduced retrieval-augmented generation (RAG), which conditions generation on documents retrieved from an external corpus, and Shuster et al. [20] demonstrated that retrieval augmentation substantially reduces hallucination in knowledge-grounded dialogue. Gao et al. [21] surveyed the rapidly growing RAG literature. Dhuliawala et al. [22] proposed chain-of-verification, in which the model drafts an answer, plans verification questions, answers them independently, and revises its draft. Wang et al. [23] showed that self-consistency, sampling several reasoning paths and selecting the most consistent answer, improves reasoning accuracy. Ouyang et al. [24] showed that reinforcement learning from human feedback can make outputs more truthful, and Li et al. [25] proposed inference-time intervention, shifting internal activations in directions associated with truthfulness. Es et al. [26] proposed RAGAS for evaluating the faithfulness of RAG outputs, and Guo et al. [27] showed that modern neural networks are often poorly calibrated, motivating calibration-aware evaluation.

E. Literature Comparison Table

Table I summarizes representative studies discussed above, organized by method, data used, task, key findings, and principal limitation.

Author/Year	Method	Data Used	Task / Application	Key Findings	Limitation
Ji et al. [1], 2023	Survey and taxonomy	Literature synthesis	General NLG	Intrinsic vs. extrinsic hallucination taxonomy	Pre-dates most modern LLM work
Huang et al. [2], 2025	Survey and taxonomy	Literature synthesis	LLMs, general	Factuality vs. faithfulness; causes across data, training, inference	Review-level; no unified benchmark protocol
Maynez et al. [5], 2020	Human evaluation	XSum summaries	Summarization	Many summaries hallucinated; ROUGE misses it	Single task; costly human labels
Lin et al. [6], 2022	Benchmark (TruthfulQA)	Adversarial questions	Closed-book QA	Larger models not necessarily more truthful	Short answers only; one error type
Li et al. [7], 2023	Benchmark (HaluEval)	Generated + annotated samples	QA, dialogue, summarization	LLMs struggle to recognise hallucinations	Many samples are synthetic
Thorne et al. [8], 2018	Dataset (FEVER)	Wikipedia claims	Claim verification	Foundation for evidence-based verification	Claims not LLM-generated
Min et al. [9], 2023	Atomic fact scoring	Biographies	Long-form generation	Claim-level precision more informative than global scores	Depends on a knowledge source
Manakul et al. [11], 2023	Sampling self-consistency	WikiBio passages	Black-box detection	No external knowledge needed	Multiple samples raise cost
Farquhar et al. [12], 2024	Semantic entropy	QA datasets	Uncertainty detection	Meaning-level uncertainty detects confabulations	Misses consistent false beliefs
Azaria & Mitchell [15], 2023	Internal-state classifier	True/false statements	White-box detection	Activations encode truthfulness	Needs model internals
Chern et al. [18], 2023	Tool-augmented checking	Multi-task claims	Fact-checking	Search and code tools improve verification	Depends on tool quality
Shuster et al. [20], 2021	Retrieval augmentation	Knowledge-grounded dialogue	Mitigation	Retrieval reduces hallucination	Bounded by retrieval quality
Dhuliawala et al. [22], 2024	Chain-of-verification	List and long-form QA	Mitigation	Self-verification lowers hallucination	Extra inference steps

TABLE I. Literature Comparison Summary

F. Research Gap

The reviewed literature shows three broad clusters: surveys that synthesize the problem without a common evaluation protocol; benchmarks that measure hallucination for a single task or error type; and detection or mitigation methods evaluated on their own chosen datasets and metrics. Relatively few studies specify a complete, reproducible pipeline that begins with a defined taxonomy, composes a multi-task benchmark covering several

hallucination types, annotates responses at the claim level, and then evaluates detection and mitigation methods together under common metrics. This gap motivates the framework proposed in Section III.

III. PROPOSED SYSTEM / RESEARCH METHODOLOGY

This section describes the proposed benchmarking framework, organized around the system architecture, hallucination taxonomy, benchmark dataset design, annotation pipeline, detection methods, and mitigation strategies.

A. System Architecture

The proposed pipeline follows the sequence: Benchmark Dataset → Prompt Construction → Target LLM → Response Logging → Hallucination Detection → Mitigation Module → Metric Computation → Benchmark Report, illustrated in Fig. 1. Benchmark items are converted into prompts using fixed templates so that every model receives identical inputs. The target LLM generates responses, which are logged together with metadata such as decoding settings, token probabilities where available, and latency. Each response is scored by one or more hallucination detectors. The same items are then re-run with mitigation strategies enabled, so that each strategy can be compared against the unmitigated baseline. Finally, metrics are computed and assembled into a benchmark report.

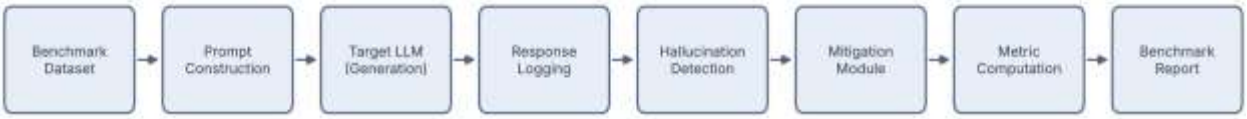


Fig. 1. Overall System Architecture

B. Hallucination Taxonomy

A consistent taxonomy is required so that annotators label the same error in the same way across tasks. Building on the factuality and faithfulness distinction of [2], the framework adopts the five categories in Table II. Each category is paired with an example and the primary signal that a detector can use to identify it.

Hallucination Type	Description	Example	Primary Detection Signal
Factual (world-knowledge)	Statement contradicts established real-world facts	Stating an incorrect year for a historical event	External knowledge base or web evidence
Contextual (faithfulness)	Output contradicts or goes beyond the provided source context	Summary adds a figure not present in the document	NLI entailment against the source
Logical / reasoning	Conclusion does not follow from the stated reasoning steps	Arithmetic chain with a correct method but wrong result	Step verification, self-consistency
Fabricated reference	Non-existent citation, URL, quote, or entity is produced	Citing a journal paper that does not exist	Citation and entity lookup
Instruction inconsistency	Output ignores or contradicts explicit user constraints	Answering in a different language than requested	Rule-based and LLM-judge checks

TABLE II. Hallucination Taxonomy Used in the Proposed Framework

C. Benchmark Dataset Design

Hallucination behaviour differs sharply across task types, so the benchmark is composed of seven task categories, shown in Fig. 2. Closed-book factual QA tests parametric knowledge; knowledge-grounded QA and summarization test faithfulness to a provided context; dialogue tests consistency over multiple turns; domain-specific prompts test specialised knowledge where errors are costly; unanswerable and false-premise questions test whether the model refuses or corrects rather than inventing an answer; and adversarial prompts test robustness to misleading phrasing. Items may be drawn from openly licensed sources such as TruthfulQA [6], HaluEval [7], and FEVER [8], supplemented by newly written items. The specific dataset size and provenance should be documented at the point of implementation, and no specific dataset size is claimed in this proposal. Table III summarizes the intended record structure.

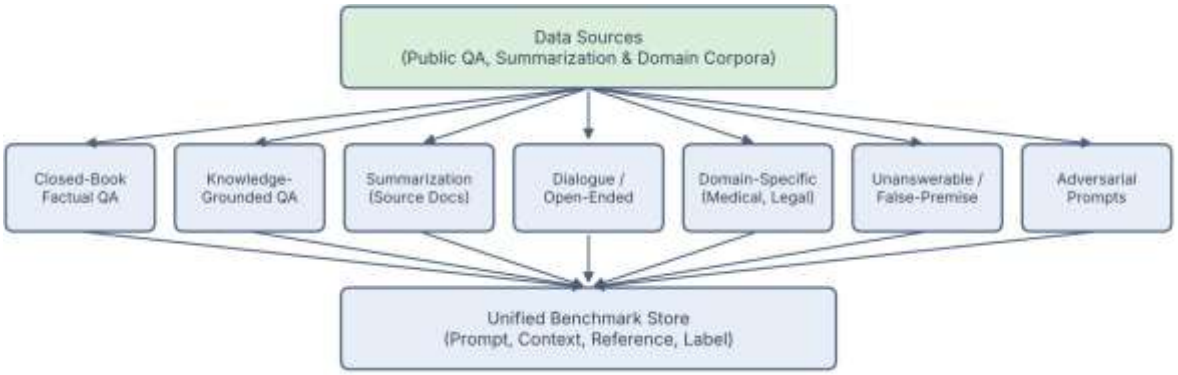


Fig. 2. Benchmark Dataset Composition Framework

Attribute	Description
Prompt	The question or instruction given to the model, rendered from a fixed template
Context	Source document or retrieved passages (for grounded tasks); empty for closed-book tasks
Reference	Gold answer, accepted answer set, or evidence passages used for verification
Task category	One of the seven categories in Fig. 2
Hallucination label	Per-claim label: supported, contradicted, unverifiable; plus taxonomy type (Table II)
Domain	General, medical, legal, financial, scientific, or other
Answerability	Answerable, unanswerable, or false-premise
Data records	To be determined at implementation, balanced across task categories
Split	Development and held-out test partitions, with test items kept private to limit contamination

TABLE III. Benchmark Record Attributes

D. Response Annotation and Preprocessing

Model responses are rarely suitable for direct scoring. The proposed annotation pipeline, shown in Fig. 3, addresses the following stages:

- Normalization and cleaning: responses are stripped of formatting artefacts, truncated at stop sequences, and normalized for whitespace, casing, numbers, and dates so that equivalent answers are matched.
- Atomic claim extraction: each response is decomposed into short, self-contained factual statements, following the atomic-fact approach of [9], so that a long answer with one wrong detail is not labelled entirely correct or entirely wrong.
- Evidence retrieval: for each claim, relevant evidence is retrieved from the provided context, the reference answer, or a fixed knowledge snapshot.
- Claim verification: each claim is labelled as supported, contradicted, or unverifiable against the retrieved evidence, using automatic verifiers as a first pass.
- Human annotation: a sample of automatic labels, together with all low-confidence cases, is reviewed by at least two trained annotators using written guidelines.
- Label adjudication: disagreements are resolved by a third annotator, and inter-annotator agreement (e.g., Cohen's kappa) is reported.
- Refusal handling: explicit refusals or statements of uncertainty are recorded separately from hallucinations, so that a model is not rewarded for confident errors or penalized for appropriate abstention.



Fig. 3. Response Annotation and Preprocessing Pipeline

E. Hallucination Detection Methods

Six representative detection approaches are considered, spanning white-box, grey-box, and black-box access. Table IV summarizes their comparative characteristics.

1) Token Probability and Entropy Scoring

This approach uses the probabilities a model assigns to its own output tokens. Low average log-probability or high predictive entropy indicates uncertainty, which is correlated with hallucination. It is cheap and requires only a single generation, but it requires access to token probabilities, and

token-level confidence does not always reflect confidence in the meaning of the answer.

2) Sampling-Based Self-Consistency

Following SelfCheckGPT [11], several responses are sampled for the same prompt, and each sentence of the main response is checked for support across the samples. Facts the model genuinely knows tend to be reproduced consistently, whereas hallucinated facts vary between samples. The method is black-box but multiplies inference cost by the number of samples.

3) Semantic Entropy

Semantic entropy [12], [13] clusters sampled answers by meaning using bidirectional entailment and computes entropy over the clusters. This avoids counting paraphrases of the same answer as disagreement. It is effective for detecting arbitrary confabulations, but cannot detect errors that the model makes consistently.

4) NLI-Based Consistency Verification

A natural language inference model checks whether each generated claim is entailed by the source context or by retrieved evidence [16], [17]. It is well suited to faithfulness errors in summarization and grounded QA, but depends on the accuracy of the NLI model, which may degrade on long or domain-specific text.

5) Retrieval-Augmented Fact-Checking

Atomic claims are verified against evidence retrieved from a knowledge base or search engine, as in FActScore [9] and FacTool [18]. This is the most direct test of factual correctness, but it inherits the coverage gaps and errors of the knowledge source and adds retrieval latency.

6) LLM-as-a-Judge

A separate, typically stronger LLM is prompted with the question, the response, and any available evidence, and asked to identify unsupported claims. It is flexible across tasks and error types, but it can share the biases of the evaluated model, is sensitive to prompt wording, and must itself be validated against human labels.

Detection Method	Model Access	Advantages	Disadvantages	Suitability
Token probability / entropy	White/grey-box	Single pass, very low cost	Needs log-probs; token confidence ≠ meaning confidence	Fast screening baseline
Sampling self-consistency	Black-box	No external knowledge needed	Cost scales with number of samples	Closed-book QA, open generation
Semantic entropy	Black-box (+ NLI)	Robust to paraphrase	Misses consistent errors	Short-answer QA
NLI verification	Black-box (+ NLI)	Strong for source faithfulness	Weak on long or specialised text	Summarization, grounded QA
Retrieval fact-checking	Black-box (+ KB)	Direct test of factual truth	Limited by knowledge coverage; latency	Long-form factual generation
LLM-as-a-judge	Black-box	Flexible across tasks and types	Judge bias; needs human validation	Mixed-task benchmarks

TABLE IV. Comparison of Hallucination Detection Methods

F. Hallucination Mitigation Strategies

Five mitigation strategies are evaluated against the unmitigated baseline, as summarized in Table V.

1) Prompt Engineering

Instructions such as asking the model to answer only from the given context, to state uncertainty, or to cite its sources are added to the prompt. This requires no infrastructure but provides limited and inconsistent improvement.

2) Retrieval-Augmented Generation

Relevant passages are retrieved from a trusted corpus and supplied with the prompt [19], [20], grounding the answer in external evidence. RAG is among the most effective strategies for knowledge-intensive tasks, but its benefit is bounded by retrieval quality, and the model can still ignore or misread retrieved evidence.

3) Chain-of-Verification and Self-Consistency

The model drafts an answer, generates verification questions, answers them independently, and revises the draft [22]; alternatively, multiple reasoning paths are sampled and the majority answer is selected [23]. These methods require no external data but increase inference cost.

4) Decoding and Training Interventions

Lower sampling temperature, factuality-aware decoding, activation steering such as inference-time intervention [25], and fine-tuning with human feedback [24] act on the model itself. They can provide lasting improvement but may require model access and retraining.

5) Confidence-Based Abstention

A detector score from Section III-E is used as a gate: when estimated hallucination risk exceeds a threshold, the system abstains, asks for clarification, or escalates to a human. This reduces the number of hallucinations delivered to users at the cost of answering fewer questions.

Mitigation Strategy	Requirements	Advantages	Disadvantages	Suitability
Prompt engineering	None	Zero cost, easy to apply	Small, inconsistent gains	Baseline improvement
Retrieval-augmented generation	Corpus + retriever	Grounds answers in evidence	Bounded by retrieval quality	Knowledge-intensive QA
Chain-of-verification	Extra inference	No external data needed	Higher latency and cost	Long-form and list answers
Decoding / training interventions	Model access	Lasting model-level improvement	Retraining cost; may reduce diversity	Model developers
Confidence-based abstention	Calibrated detector	Prevents risky answers reaching users	Lower coverage	High-stakes deployments

TABLE V. Comparison of Hallucination Mitigation Strategies

IV. PROPOSED BENCHMARKING FRAMEWORK

Building on the components described in Section III, this section specifies the end-to-end benchmarking workflow as a step-by-step algorithm, illustrated in Fig. 4.

1. Define the hallucination taxonomy and annotation guidelines (Table II).
2. Collect and curate benchmark items across the seven task categories.
3. Split items into development and held-out test partitions; keep the test set private.
4. Render each item into a prompt using fixed templates.
5. Generate responses from each target LLM with fixed decoding settings.
6. Log responses together with token probabilities, samples, and latency where available.
7. Normalize responses and extract atomic claims.
8. Verify each claim against evidence and obtain human-adjudicated labels.
9. Run each detection method and record a hallucination score per response and per claim.
10. Calibrate detector thresholds on the development partition.
11. Re-run generation on the test partition with each mitigation strategy enabled.
12. Compute detection and mitigation metrics (Section VI).
13. Compare methods on effectiveness, computational cost, and answer utility.
14. Publish a benchmark report and leaderboard with all settings documented.

A simplified pseudocode representation of the evaluation loop is as follows:

```
for llm in target_models:
  for strategy in [None, Prompt, RAG, CoVe,
                   Decoding, Abstain]:
    R = llm.generate(test_prompts, strategy)
    claims = extract_claims(R)
    labels = verify(claims, evidence)
    for detector in detectors:
      s = detector.score(R)
      record AUROC, F1(labels, s)
    record HR, FactPrecision, ECE
report best detector and mitigation
  per task category and cost
```

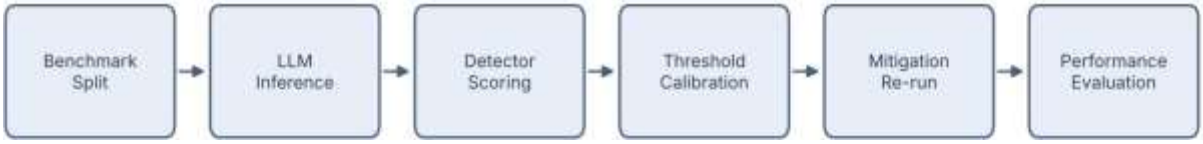


Fig. 4. Benchmark Evaluation Workflow

V. EXPERIMENTAL SETUP

This section describes how the proposed framework can be experimentally evaluated once model access and annotated data are obtained. As no models were run for this proposal, the setup below is presented as the intended implementation configuration rather than a record of experiments actually performed.

- Target models: a mix of open-weight LLMs of different sizes, which allow white-box detectors, and API-based models, which allow only black-box detectors.
- Hardware requirements: a GPU workstation or cloud instance for running open-weight models and NLI verifiers; API access for hosted models.
- Software requirements: Python 3.x as the primary development language.
- Libraries: Hugging Face Transformers for open-weight models and NLI models; a vector database and embedding model for retrieval; Pandas, NumPy, and Scikit-learn for metric computation; Matplotlib for visualization.
- Knowledge source: a fixed, dated snapshot of a reference corpus such as Wikipedia, so that verification results are reproducible.
- Decoding settings: identical temperature, maximum length, and stop sequences for all models, with multiple samples drawn for sampling-based detectors.

- Annotation: at least two annotators per item, with a third for adjudication, and reported inter-annotator agreement.
- Validation procedure: detector thresholds tuned only on the development partition and evaluated once on the held-out test partition to avoid optimistic bias.
- Contamination control: test items are kept private and periodically refreshed to reduce the risk that they appear in model training data.

VI. PERFORMANCE EVALUATION

Because the framework evaluates both how often a model hallucinates and how well detectors and mitigation strategies respond, the following metrics are proposed.

Hallucination Rate (HR)

HR = H / N, where H is the number of responses containing at least one contradicted or fabricated claim and N is the total number of evaluated responses. Lower values indicate a more reliable model. HR is reported per task category and per hallucination type.

Factual Precision (FP)

FP = S / C, where S is the number of supported atomic claims and C is the total number of atomic claims in a response, averaged over responses [9]. FP rewards responses whose individual statements are correct, rather than judging a long answer as a single unit.

Detection Precision, Recall, and F1-Score

Treating hallucinated responses as the positive class, Precision = TP / (TP + FP), Recall = TP / (TP + FN), and F1 = 2 · Precision · Recall / (Precision + Recall). High recall is important where a missed hallucination is costly; high precision is important where false alarms would cause unnecessary refusals.

Area Under the ROC Curve (AUROC)

AUROC measures how well a detector's continuous score ranks hallucinated responses above correct ones, independent of any threshold. A value of 0.5 corresponds to random ranking and 1.0 to perfect separation.

Expected Calibration Error (ECE)

ECE = Σ ($|B_m| / n$) · $|acc(B_m) - conf(B_m)|$, where responses are grouped into bins B_m by confidence, acc is the observed accuracy and conf the mean confidence in each bin [27]. Low ECE means a model's stated confidence can be trusted, which is essential for abstention-based mitigation.

Relative Hallucination Reduction (RHR)

RHR = (HR_base − HR_mit) / HR_base, the proportion of baseline hallucinations removed by a mitigation strategy. RHR is reported together with answer coverage (the fraction of questions answered rather than abstained) and task accuracy, so that a strategy is not credited for reducing hallucination simply by refusing to answer.

Each metric offers a different lens: HR and FP describe the model's reliability; precision, recall, F1, and AUROC describe the detector; ECE describes how trustworthy confidence estimates are; and RHR with coverage describes the real benefit of mitigation.

VII. RESULTS AND ANALYSIS

No models were executed and no benchmark responses were annotated for this study. Consequently, this section presents a proposed, illustrative results framework rather than genuine experimental findings. After implementation, the framework can be assessed using the table and figures below populated with actual computed values; the entries shown here are placeholders that describe the intended structure of the analysis and must not be interpreted as measured performance.

Configuration	HR	Factual Precision	Detector F1	AUROC	ECE	RHR	Coverage
Baseline (no mitigation)	—	—	—	—	—	—	—
Prompt engineering	—	—	—	—	—	—	—
Retrieval-augmented generation	—	—	—	—	—	—	—
Chain-of-verification	—	—	—	—	—	—	—
Decoding / training intervention	—	—	—	—	—	—	—
Confidence-based abstention	—	—	—	—	—	—	—

TABLE VI. Benchmark Evaluation Metrics — To Be Populated From Actual Experimental Data (Not Measured In This Study)

To illustrate what the evaluation outputs would look like once genuine data is available, Figs. 5–7 show representative, clearly labelled illustrative visualizations rather than results derived from this study. Fig. 5 shows an illustrative hallucination rate per task category, reflecting the qualitative pattern reported in prior work that false-premise, adversarial, and domain-specific prompts tend to elicit more hallucinations than grounded tasks [6], [7]. Fig. 6 shows illustrative ROC curves of the kind used to compare detectors, with evidence-based verifiers expected to rank above probability-only scores. Fig. 7 shows an illustrative comparison of hallucination rate after each mitigation strategy, consistent in relative ordering with studies reporting that retrieval and verification reduce hallucination [20], [22]; the specific values are for illustration only and are not derived from an executed experiment.

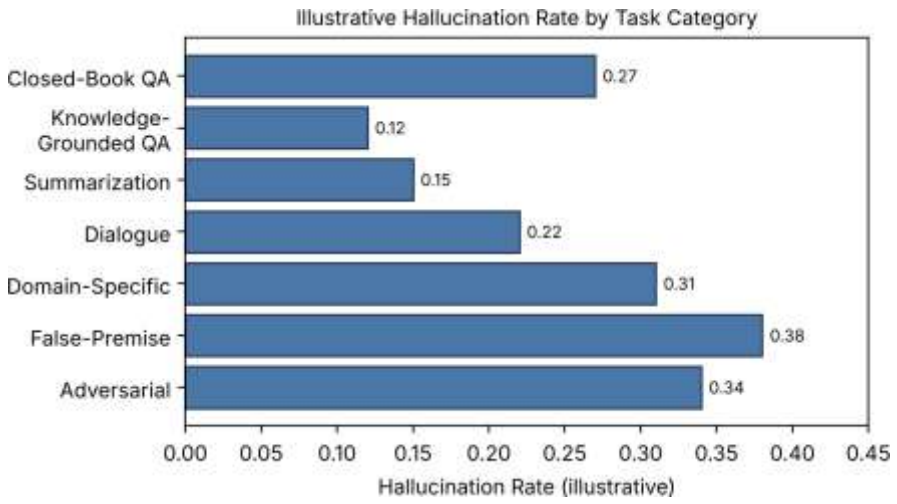


Fig. 5. Illustrative Hallucination Rate by Task Category (not measured in this study)

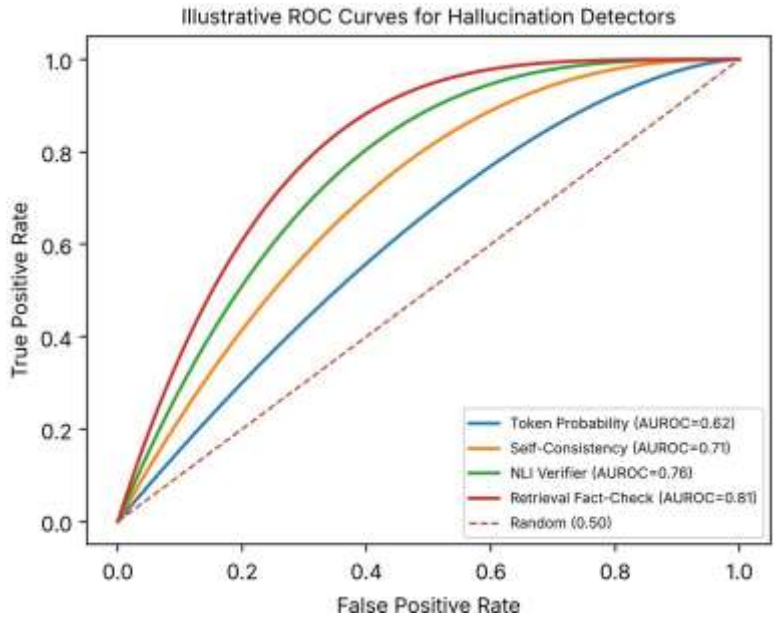


Fig. 6. Illustrative ROC Curves for Hallucination Detectors (not measured in this study)

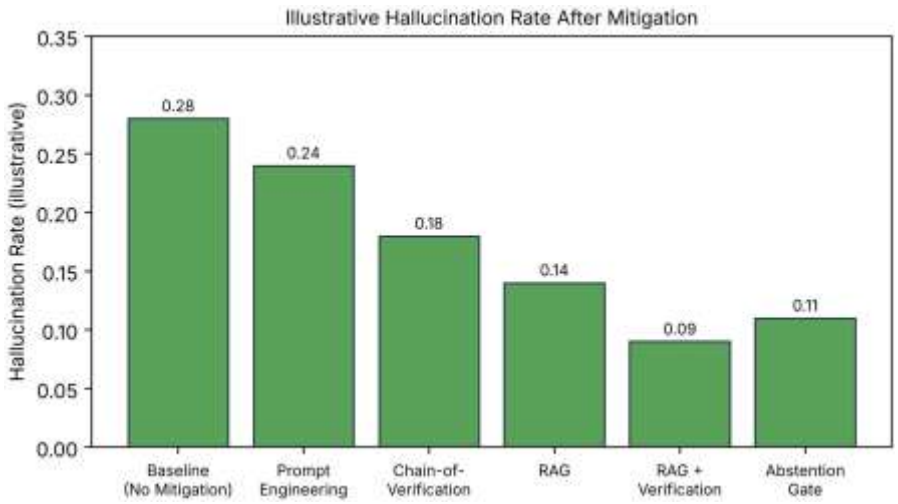


Fig. 7. Illustrative Hallucination Rate After Mitigation (not measured in this study)

Beyond the specific illustrative values, the qualitative pattern expected, and consistent with the reviewed literature, is that grounding-based approaches (retrieval-augmented generation and retrieval-based fact-checking) will be the most effective for factual hallucinations, while consistency- and uncertainty-based methods will be most useful where no external knowledge source is available. Combining mitigation with a detector-driven abstention gate is expected to give the lowest delivered hallucination rate, at the cost of reduced coverage.

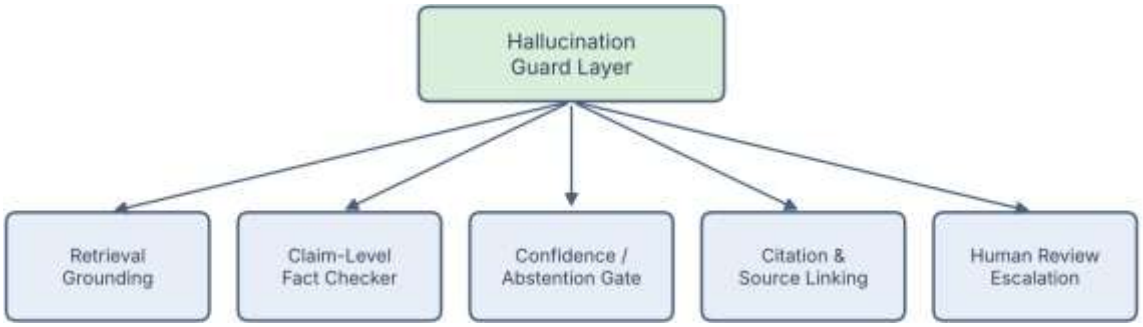


Fig. 8. Proposed Hallucination Guard Architecture for Deployment

Fig. 8 shows how the outputs of the benchmark are intended to inform a deployable hallucination guard layer, combining retrieval grounding, claim-level fact checking, a confidence-based abstention gate, citation and source linking, and escalation to human review, rather than terminating at a single benchmark score.

VIII. DISCUSSION

A unified benchmark improves hallucination research primarily by replacing incompatible, task-specific evaluations with a common protocol in which the same items, labels, and metrics are used to judge every model, detector, and mitigation strategy. Claim-level annotation makes it possible to locate exactly which statements in a long answer are wrong, which is more informative than a single correct-or-incorrect judgement and is closer to how users experience hallucinations. Evaluating detection and mitigation together also exposes trade-offs that isolated studies hide: a mitigation strategy may lower the hallucination rate mainly by refusing to answer, or a detector may perform well on summarization but poorly on closed-book questions.

No single detection method is sufficient. Probability-based methods are cheap but require model access and do not capture meaning; consistency-based methods are model-agnostic but miss errors the model makes confidently and consistently; evidence-based methods are the most direct test of truth but are limited by the coverage and freshness of the knowledge source. The framework is therefore designed to encourage layered combinations of methods, with the benchmark revealing which combination is most effective for each task category.

Several practical concerns affect real-world use. Benchmark contamination, where test items appear in a model's training data, can inflate results, which motivates private and periodically refreshed test sets. Annotation is expensive and subjective, particularly for long-form and domain-specific outputs, and automatic verifiers and LLM judges introduce their own errors that must be measured against human labels. Factual ground truth changes over time, so knowledge snapshots must be dated and versioned. Finally, the cost of sampling-based detection and multi-step verification must be weighed against latency and budget constraints in production systems.

IX. REAL-WORLD APPLICATIONS

1. Model selection: helping organizations choose the LLM with the lowest hallucination rate for their task and domain.
2. Healthcare assistants: screening clinical question-answering systems for fabricated drug, dosage, or guideline information before deployment.
3. Legal research tools: detecting fabricated case citations and statutes in generated legal text.
4. Education and tutoring: ensuring that AI tutors do not teach students incorrect facts.
5. Enterprise RAG systems: measuring how faithfully chatbots answer from internal documents.
6. Journalism and content creation: flagging unsupported claims in AI-assisted drafts before publication.
7. Customer support: preventing assistants from inventing policies, prices, or procedures.
8. Scientific writing: detecting fabricated references and misreported findings in AI-assisted literature reviews.
9. Regulatory auditing: providing standardized evidence of model reliability for AI governance and compliance.
10. Model development: giving developers a regression test to track hallucination across model versions.

X. LIMITATIONS

Table VII summarizes the principal limitations of the proposed framework.

Limitation	Description
Annotation cost	Claim-level human annotation is slow and expensive, especially for long-form outputs
Limitation	**Description**
Annotator subjectivity	Annotators may disagree on whether a claim is unverifiable or contradicted
Knowledge-source coverage	Evidence-based verification cannot confirm facts absent from the knowledge base
Temporal drift	Facts change over time, so labels can become outdated without versioned snapshots
Benchmark contamination	Public benchmark items may leak into model training data and inflate scores
Judge bias	LLM-as-a-judge evaluators may share the biases and errors of the evaluated models
Language coverage	The framework as proposed focuses on English; other languages need separate data
Model access	White-box detectors cannot be applied to closed API-based models
Computational cost	Sampling-based detection and multi-step verification multiply inference cost
Metric trade-offs	Lower hallucination rates can be achieved by over-refusal, reducing usefulness

TABLE VII. Limitations of the Proposed Framework

These limitations mean the proposed benchmark should not be regarded as a complete measure of model trustworthiness; its conclusions will depend on the quality of annotation, the coverage of the knowledge source, and careful, task- and domain-specific validation before its results are relied upon for high-stakes decisions.

XI. FUTURE WORK

- Multilingual extension: building benchmark items and annotators for Indian and other low-resource languages, including Tamil.
- Multimodal hallucination: extending the framework to vision-language models that describe objects or text not present in an image.
- Dynamic benchmarks: automatically refreshing test items from recent events to reduce contamination and test up-to-date knowledge.
- Agentic evaluation: measuring hallucination in tool-using agents, where fabricated tool outputs or actions compound over multiple steps.
- Explainable detection: highlighting the exact span and evidence behind each hallucination flag to support user trust.
- Cost-aware evaluation: reporting accuracy alongside latency and inference cost to guide practical deployment.
- Automated annotation: improving claim extraction and verification models to reduce reliance on human labelling.
- Domain-specific suites: dedicated healthcare, legal, and financial modules with expert annotators.
- Open leaderboard: publishing the benchmark with a public leaderboard and private test server.
- Training integration: using benchmark failures to create training data that reduces hallucination in future models.

XII. CONCLUSION

This paper has proposed a unified benchmarking framework for detecting and reducing hallucinations in Large Language Models, addressing the fragmentation of current evaluation practice in which benchmarks, detectors, and mitigation techniques are studied in isolation. The framework specifies a five-category hallucination taxonomy, a multi-task benchmark composition, a claim-level annotation pipeline, and a common evaluation protocol, and conceptually compares six detection approaches — token probability and entropy scoring, sampling-based self-consistency, semantic entropy, NLI verification, retrieval-augmented fact-checking, and LLM-as-a-judge — together with five mitigation strategies. Because no models were executed and no annotated data was collected for this study, the work is presented explicitly as a proposed framework and experimental design rather than a report of measured results, and all illustrative figures and tables are labelled accordingly. If implemented, the framework is expected to give model developers, deployers, and regulators a consistent, reproducible way to measure how often LLMs hallucinate and which techniques most effectively reduce it. Realizing this potential will require addressing the limitations discussed in Section X, particularly annotation cost, knowledge coverage, and contamination, and pursuing the future directions outlined in Section XI, from multilingual and multimodal extension to an open public leaderboard.

ACKNOWLEDGMENT

The author gratefully acknowledges the guidance and support of the faculty guide and the Department of Computer Science and Engineering, Kings College of Engineering, Punalkulam, Tamil Nadu, India, in the preparation of this work. Appreciation is also extended to the institution for providing the academic environment and resources that made this research possible.

REFERENCES

[1] Z. Ji et al., “Survey of hallucination in natural language generation,” ACM Computing Surveys, vol. 55, no. 12, Art. 248, 2023, doi: 10.1145/3571730.

[2] L. Huang et al., “A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions,” ACM Transactions on Information Systems, vol. 43, no. 2, 2025, doi: 10.1145/3703155.

[3] Y. Zhang et al., “Siren's song in the AI ocean: A survey on hallucination in large language models,” arXiv:2309.01219, 2023.

[4] S. M. T. I. Tonmoy et al., “A comprehensive survey of hallucination mitigation techniques in large language models,” arXiv:2401.01313, 2024.

[5] J. Maynez, S. Narayan, B. Bohnet, and R. McDonald, “On faithfulness and factuality in abstractive summarization,” in Proc. ACL, 2020, pp. 1906–1919, doi: 10.18653/v1/2020.acl-main.173.

[6] S. Lin, J. Hilton, and O. Evans, "TruthfulQA: Measuring how models mimic human falsehoods," in Proc. ACL, 2022, pp. 3214–3252, doi: 10.18653/v1/2022.acl-long.229.

[7] J. Li, X. Cheng, W. X. Zhao, J.-Y. Nie, and J.-R. Wen, "HaluEval: A large-scale hallucination evaluation benchmark for large language models," in Proc. EMNLP, 2023, pp. 6449–6464, doi: 10.18653/v1/2023.emnlp-main.397.

[8] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal, "FEVER: A large-scale dataset for fact extraction and VERification," in Proc. NAACL-HLT, 2018, pp. 809–819, doi: 10.18653/v1/N18-1074.

[9] S. Min et al., "FActScore: Fine-grained atomic evaluation of factual precision in long form text generation," in Proc. EMNLP, 2023, pp. 12076–12100, doi: 10.18653/v1/2023.emnlp-main.741.

[10] J. Wei et al., "Long-form factuality in large language models," in Proc. NeurIPS, 2024, arXiv:2403.18802.

[11] P. Manakul, A. Liusie, and M. J. F. Gales, "SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models," in Proc. EMNLP, 2023, pp. 9004–9017, doi: 10.18653/v1/2023.emnlp-main.557.

[12] S. Farquhar, J. Kossen, L. Kuhn, and Y. Gal, "Detecting hallucinations in large language models using semantic entropy," Nature, vol. 630, pp. 625–630, 2024, doi: 10.1038/s41586-024-07421-0.

[13] L. Kuhn, Y. Gal, and S. Farquhar, "Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation," in Proc. ICLR, 2023.

[14] S. Kadavath et al., "Language models (mostly) know what they know," arXiv:2207.05221, 2022.

[15] A. Azaria and T. Mitchell, "The internal state of an LLM knows when it's lying," in Findings of EMNLP, 2023, pp. 967–976.

[16] O. Honovich et al., "TRUE: Re-evaluating factual consistency evaluation," in Proc. NAACL, 2022, pp. 3905–3920.

[17] P. Laban, T. Schnabel, P. N. Bennett, and M. A. Hearst, "SummaC: Re-visiting NLI-based models for inconsistency detection in summarization," Transactions of the ACL, vol. 10, pp. 163–177, 2022.

[18] I. Chern et al., "FacTool: Factuality detection in generative AI — A tool augmented framework for multi-task and multi-domain scenarios," arXiv:2307.13528, 2023.

[19] P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in Proc. NeurIPS, vol. 33, 2020, pp. 9459–9474.

[20] K. Shuster, S. Poff, M. Chen, D. Kiela, and J. Weston, "Retrieval augmentation reduces hallucination in conversation," in Findings of EMNLP, 2021, pp. 3784–3803.

[21] Y. Gao et al., "Retrieval-augmented generation for large language models: A survey," arXiv:2312.10997, 2023.

[22] S. Dhuliawala et al., "Chain-of-verification reduces hallucination in large language models," in Findings of ACL, 2024, pp. 3563–3578.

[23] X. Wang et al., "Self-consistency improves chain of thought reasoning in language models," in Proc. ICLR, 2023.

[24] L. Ouyang et al., "Training language models to follow instructions with human feedback," in Proc. NeurIPS, vol. 35, 2022, pp. 27730–27744.

[25] K. Li, O. Patel, F. Viégas, H. Pfister, and M. Wattenberg, "Inference-time intervention: Eliciting truthful answers from a language model," in Proc. NeurIPS, 2023.

[26] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "RAGAS: Automated evaluation of retrieval augmented generation," in Proc. EACL: System Demonstrations, 2024, pp. 150–158.

[27] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in Proc. ICML, 2017, pp. 1321–1330.