LITERATURE SURVEY

1. Foundations: Understanding Hallucination in LLMs

Recent comprehensive surveys have established the taxonomy and root causes of hallucinations, defining the scope of the problem your project addresses.

- **Ji et al. (2023) –** ***Survey of Hallucination in Large Language Models***
  - **Definition:** This work defines hallucination as content generated by an LLM that is fluent and syntactically correct but factually inaccurate or unsupported by external evidence.
  - **Taxonomy:** It distinguishes between Intrinsic Hallucinations (where the generated output contradicts the source material or itself) and Extrinsic Hallucinations (where the output cannot be verified from the source or contradicts real-world knowledge).
  - **Causes:** The survey identifies that hallucinations stem from both data collection issues (misinformation in training data) and inference mechanisms (decoding strategies like top-k sampling that prioritize diversity over accuracy).
- **Huang et al. (2023) –** ***A Survey on Hallucination in Large Language Models***
  - **Categorization:** This survey introduces a distinction between Factuality Hallucination (deviating from established real-world facts) and Faithfulness Hallucination (diverging from the user's provided input context).

- **Relevance to Project:** It highlights that while LLMs excel at reasoning, they struggle with "knowledge-intensive" tasks due to limited parametric memory, validating your project's focus on QA tasks.

---

2. The "Reactive" Standard: Post-Hoc Detection

Current industry standards largely rely on checking the model's work *after* it has been generated. This represents the "reactive" approach your project aims to replace.

- **Manakul et al. (2023) – *SelfCheckGPT***
  - **Methodology:** This is a "zero-resource" black-box method. It detects hallucinations by sampling multiple stochastic responses (e.g., 5–10 variations) from the LLM for the same query.
  - **The Core Idea:** It operates on the principle of consistency: if an LLM "knows" a fact, its sampled responses will likely be similar. If it is hallucinating, the samples will diverge and contradict one another.
  - **Limitations (The Gap):**
    - **Computational Cost:** It requires generating multiple answers for every single query, making it roughly 4.2x more expensive than standard generation.
    - Latency: It is slow because it acts only after the initial generation is complete, making it unsuitable for real-time low-latency applications.
    -

3. The "Preventative" Standard: Retrieval-Augmented Generation

To prevent hallucinations, RAG was introduced to ground LLMs in external data.

- **Lewis et al. (2020) – *Retrieval-Augmented Generation (RAG)***
    - **Methodology:** This seminal work combined a pre-trained sequence-to-sequence generator (like BART) with a dense vector retriever (DPR). Before answering, the model retrieves relevant documents (non-parametric memory) to inform its response.
    - **Impact:** It significantly reduced hallucinations in knowledge-intensive tasks by reducing reliance on the model's internal (parametric) memory.
    - **Limitations (The Gap):**
        - **"Always-On" Inefficiency:** Standard RAG retrieves documents for *every* query, even simple ones (e.g., "What is the capital of France?"), adding unnecessary latency and computational overhead.
        - **Context Hallucination:** If the retrieved documents are irrelevant or conflicting, the model can still hallucinate by misinterpreting the context, a failure mode your project explicitly targets with its "Context Conflict" check.

4. The "Proactive" Shift: Confidence-Aware Routing (Your Base Paper)

Your project directly extends this recently proposed framework which attempts to solve the inefficiencies of the above methods.

- **N. M (2025) – *Confidence-Aware Routing for LLM Reliability***

- **Core Philosophy:** This paper proposes shifting from "reactive correction" to "proactive assessment." It predicts reliability *before* generation to decide if the model needs help.
- **Methodology:** It calculates a confidence score using three internal signals:
    1. **Semantic Alignment:** Measuring the distance between the model's internal representations and reference embeddings.
    2. **Internal Convergence:** Analysing how stable the model's hidden states are across its layers.
    3. **Learned Confidence:** A dedicated estimation of the model's own certainty.
- **Routing:** Based on this score, it routes queries to: Direct Generation (High confidence), RAG (Medium confidence), or Human Review (Low confidence).
- **Your Contribution (The Extension):** The base paper is "Self-Aware" (internal signals only). Your project makes it "Context-Aware" by adding an External Reality Check (Context Scarcity, Conflict, and Domain Mismatch) to catch cases where the model feels confident but is being misled by bad data.