Dissertation on

"A Cost-Effective, Proactive Hallucination Routing System for LLMs"

Submitted in partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology
in
Computer Science & Engineering (Artificial Intelligence and Machine Learning)

UE23AM320A – Capstone Project Phase - 1

Submitted by:

Sourabh S Mahindrakar	**PES1UG23AM313**
Chandan R	**PES1UG23AM917**
Sreephaneesha k	**PES1UG23AM314**
Sri Charan D A	**PES1UG23AM315**

Under the guidance of

Dr. Ravi Gorripati
Associate Professor
PES University

August - December 2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100 feet Ring road, BSK 3rd stage, Hosakerehalli, Bengaluru – 560085

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled

'A Cost-Effective, Proactive Hallucination Routing System for LLMs'

is a bonafide work carried out by

Sourabh S Mahindrakar	**PES1UG23AM313**
Chandan R	**PES1UG23AM917**
Sreephaneesha k	**PES1UG23AM314**
Sri Charan D A	**PES1UG23AM315**

In partial fulfilment for the completion of Fifth-semester Capstone Project Phase - 1 (UE23AM320A) in the Program of Study -Bachelor of Technology in Computer Science and Engineering (Artificial Intelligence and Machine Learning) under rules and regulations of PES University, Bengaluru during the period Aug. 2025 – Dec. 2025. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 5th-semester academic requirements in respect of project work.

Signature **Dr. Ravi Gorripati** Associate Professor	Signature Dr.Jayashree R Chairperson	Signature Dr. K S Sridhar Dean of Faculty and Registrar

External Viva

Name of the Examiners	**Signature with Date**
1. _____	_____
2. _____	_____

DECLARATION

We hereby declare that the Capstone Project Phase - 1 entitled **"A Cost-Effective, Proactive Hallucination Routing System for LLMs"** has been carried out by us under the guidance of **Dr. Ravi Gorripati, Associate Professor** and submitted in partial fulfilment of the course requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering (Artificial Intelligence and Machine Learning)** of **PES University, Bengaluru** during the academic semester Aug – Dec 2025. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

PES1UG23AM313	**Sourabh S M**	_____
PES1UG23AM917	**Chandan R**	_____
PES1UG23AM314	**Sreephaneesha K**	_____
PES1UG23AM315	**Sri Charan D A**	_____

ACKNOWLEDGEMENT

I would like to express my gratitude to **Dr. Ravi Gorripati,** Department of Computer Science & Engineering (Artificial Intelligence and Machine Learning), PES University, for his continuous guidance, assistance, and encouragement throughout the development of UE23AM320A- Capstone Project Phase – 1.

I am grateful to Capstone Project Coordinators, **Dr. Chetana Srinivas** for organizing, managing, and helping with the entire process.

I take this opportunity to thank **Dr. Jayashree R,** Professor & Chairperson, Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning), PES University, for all the knowledge and support I have received from the department. I would like to thank **Dr. K S Sridhar** Dean of Faculty and Registrar, PES University for his help.

I am deeply grateful to **Prof. Jawahar Doreswamy**, **Chancellor, PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University,** for providing me with various opportunities and enlightenment every step of the way. Finally, Phase 1 of the project could not have been completed without the continual support and encouragement I have received from my family and friends.

ABSTRACT

Large Language Models are powerful, but they also love to make things up—confidently. Most existing solutions try to fix these hallucinations after the model has already produced a bad answer, which wastes compute and doesn't stop the damage. This paper takes a different route: instead of reacting after the fact, it judges before the model even begins generating whether the answer is likely to be trustworthy.

The proposed system estimates the model's confidence using three signals:

- How closely its internal representations match reliable reference embeddings,
- How steadily its reasoning progresses through layers, and
- A learned predictor trained directly on activation patterns.

These signals are combined into one confidence score. Based on that score, the system routes the question to one of several paths letting the small model answer if it's confident, using retrieval when certainty dips, escalating to a bigger model when needed, and falling back to a human only when the model is truly lost.

Across standard knowledge-heavy QA benchmarks, this proactive approach catches hallucinations far better than older methods while using significantly less compute.

Instead of treating hallucinations like an afterthought, this framework brings early awareness into LLM pipelines—making them faster, safer, and more reliable for real-world use

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 The Problem: LLMs Sound Smart but Often Guess Wrong

Large Language Models are incredible at producing fluent, convincing text—but that fluency hides a dangerous flaw: they routinely generate statements that *sound true* but are actually wrong. These hallucinations are not rare, and in high-stakes settings like medical advice or legal reasoning, they can cause real harm.

1.2 Why Current Fixes Aren't Enough

Most existing systems try to catch hallucinations *after* the model has already generated an answer. Retrieval-augmented generation, consistency checks, or post-hoc verification help, but they waste compute and fail to prevent the model from producing false content in the first place. In other words, the model has already "spoken," and now you're scrambling to clean up the mess.

1.3 A Shift in Strategy: Stop the Hallucination Before It Starts

This project flips the usual pipeline. Instead of waiting for the model to hallucinate, we analyse the model's internal signals *before* generation begins. The goal is simple: determine whether the model is likely to answer reliably. If not, the system automatically reroutes the query to a safer option—retrieval, a larger model, or a human.

1.4 How We Estimate Confidence

The confidence estimator draws from three powerful internal clues:

1.4.1. **Semantic Alignment**

We compare the model's internal representation of the query with embeddings from a trusted reference model. If alignment is weak, the model probably doesn't "understand" the query well.

1.4.2. **Internal Convergence**

We look at how stable the hidden layers are as the model processes the input. Poorly converging layers usually signal uncertainty or confusion.

1.4.3. **Learned Confidence**

A dedicated neural predictor is trained to read internal activations and estimate reliability directly. These three signals are combined into a single confidence score.

1.5 Smart Routing Based on Confidence

Once the score is computed, the system chooses one of four paths:

- **High confidence** → small local model responds
- **Medium confidence** → use retrieval to ground the answer
- **Low confidence** → escalate to a larger, stronger model
- **Very low confidence** → hand off to a human reviewer

This makes the system faster, cheaper, and safer than post-generation fixes.

1.6 What This Achieves

Across multiple QA benchmarks, this method sharply improves hallucination detection while cutting computational cost by around 40%. The model gets better at knowing *when it doesn't know*, and that self-awareness leads to fewer mistakes and tighter control over reliability.

Feature	Reactive Standard (Current Industry)	Proactive Approach (Proposed System)
Timing of Detection	Checks reliability **after** the text is generated.	Estimates confidence **before** generation begins.
Computation Cost	High: Wastes compute generating bad answers that are later discarded.	Low: Routes simple queries to small models and prevents wastage.
Methodology	Relies on consistency checks (SelfCheckGPT) or post-hoc verification.	Uses internal signals (Alignment, Convergence) and external risk checks.
Goal	"Fix the mess" after the model has spoken.	"Stop the hallucination" before it starts.

Table 1.1 Reactive vs Proactive

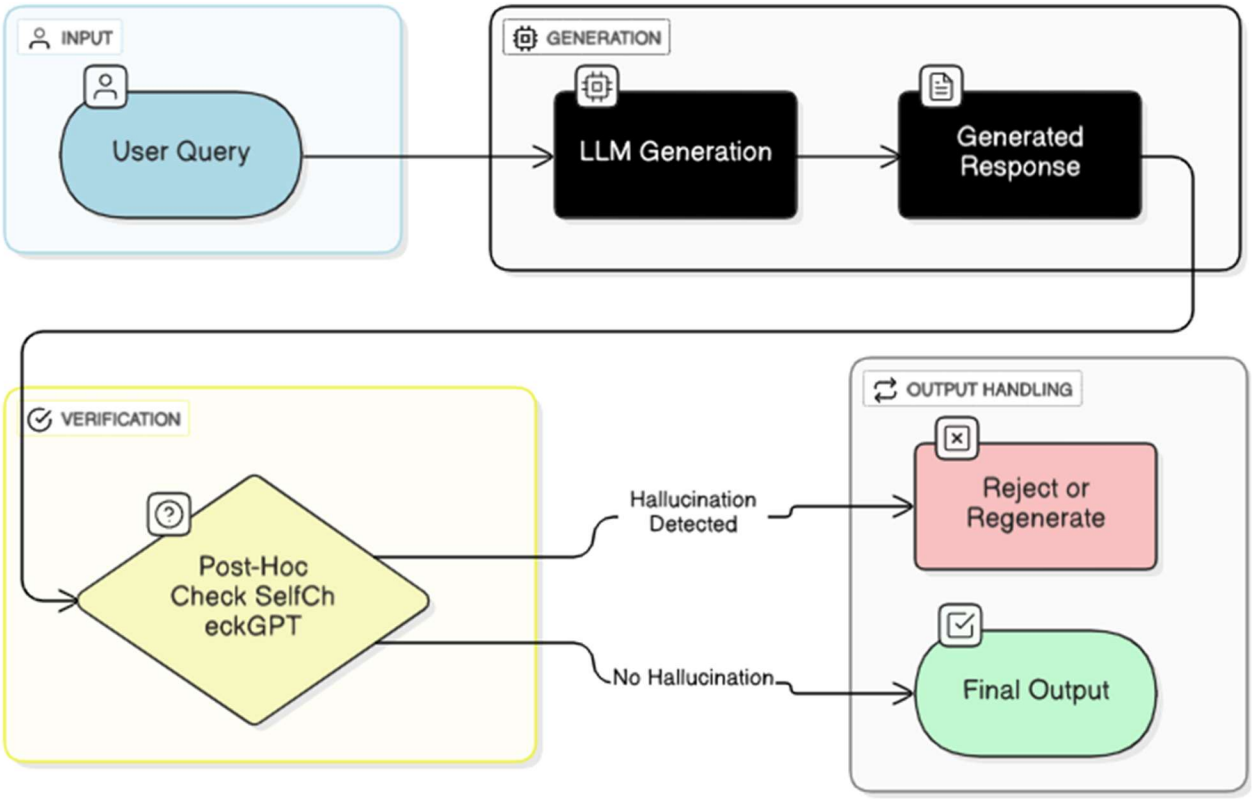

Figure 1.1 Traditional Reactive Hallucination Mitigation Pipeline

CHAPTER 2

PROBLEM DEFINITION

Large Language Models often produce answers that sound confident but are actually incorrect. These hallucinations happen because the model lacks self-awareness—it cannot reliably judge when it doesn't know enough to answer. Most existing solutions try to fix the problem *after* the model has already generated an unreliable response, which wastes computation and still allows incorrect information to appear.

To address this fundamental weakness, the project focuses on building a system that can proactively assess the model's confidence before generation and route the query safely based on that assessment.

2.1 Objectives

- **Estimate Confidence Before Generation**

  Develop a mechanism that predicts whether the model is likely to hallucinate *before* it produces an output.

- **Use Multi-Signal Confidence Assessment**

  Combine semantic alignment, internal processing stability, and a learned confidence predictor into one reliable score.

- **Implement Intelligent Routing**

  Route queries to the appropriate path—local generation, retrieval augmentation, larger model, or human review—based on confidence thresholds.

- **Reduce Computational Overhead**

  Avoid unnecessary use of large models or post-hoc checks, improving overall efficiency.

- **Enhance Reliability and Accuracy**

  Improve hallucination detection rates, reduce false positives, and significantly boost F1 performance over baseline methods.

CHAPTER 3

LITERATURE SURVEY

3.1. 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.

3.2. 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.

3.3. 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.

3.4. The "Proactive" Shift: Confidence-Aware Routing

- **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).
    - **Our Contribution (The Extension):** The base paper is “Self-Aware” (internal signals only). Our 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.

Study / Author	Approach Type	Key Methodology	Limitation (The Gap)
Ji et al. (2023); Huang et al. (2023)	Survey / Taxonomy	Defined hallucinations (Intrinsic vs. Extrinsic, Factuality vs. Faithfulness) and causes.	Identifies the problem but offers a definition rather than a solution.
SelfCheckGPT (Manakul et al., 2023)	Reactive (post-Hoc)	Samples multiple stochastic responses to check for consistency; if samples diverge, it's a hallucination.	**High Latency & Cost:** Roughly 4.2x more expensive than standard generation; unsuitable for real-time apps.
RAG (Lewis et al., 2020)	Preventative	Retrieves external documents to ground the model before answering.	**"Always-On" Inefficiency:** Retrieves data even for simple queries, adding unnecessary overhead.
Confidence-Aware Routing (N. M, 2025)	Proactive (Self-Aware)	Predicts reliability *before* generation using internal signals like semantic alignment and convergence[7777].	**Lacks Context Awareness:** Only looks at internal signals, missing cases where the model is confident but misled by bad context[8].

Table 3.1 Summary of Literature Survey

CHAPTER 4

RESEARCH / TECHNOLOGY GAPS AND CHALLENGES

Despite rapid progress in large language models, several critical gaps limit their reliability—especially when handling knowledge-intensive or safety-sensitive tasks. This project directly addresses these shortcomings, but the broader challenges remain important to highlight.

4.1. Lack of Pre-Generation Uncertainty Awareness

Most existing systems detect hallucinations *after* the model has already generated an answer. There is no reliable mechanism for an LLM to assess its uncertainty **before** responding, leading to confident but incorrect outputs. This absence of proactive evaluation is a major gap in current LLM pipelines.

4.2. Overreliance on Post-Hoc Corrections

Techniques such as RAG, self-consistency checks, and output verification are computationally heavy and often too late—they only react after the hallucination has occurred. This creates unnecessary compute costs and fails to prevent misinformation.

4.3. Limited Use of Internal Model Signals

LLMs generate rich internal activations, but most hallucination-detection approaches barely use them. Important indicators such as semantic alignment drift, unstable layer progression, or inconsistent hidden-state behaviour are **under-explored**, leaving a large gap in reliable uncertainty quantification.

4.4. Absence of Unified Confidence Scoring

Existing uncertainty-estimation techniques tend to rely on a single method—entropy, sampling, embeddings, or external classifiers. None provide a **combined, multi-signal confidence score** that captures semantic, structural, and learned aspects of reliability. This lack of integration reduces accuracy and increases false positives.

4.5. Inefficient Routing of Queries

Current LLM systems do not make smart decisions about *where* a query should be handled. Heavy models are often used unnecessarily, while low-confidence queries are not escalated properly. A major gap is the absence of **deterministic routing mechanisms** tied to confidence estimation.

4.6. Bias and Domain Sensitivity Challenges

Embedding-based alignment and internal confidence predictors depend heavily on reference models and training data. This introduces risks such as:

- domain-specific inaccuracies
- biased confidence estimation
- mis-calibration in unfamiliar contexts

4.7. Threshold Generalization Issues

Static confidence thresholds can fail across domains or user contexts. Without adaptive thresholding, routing decisions may become inconsistent or unreliable—especially with varied query types.

4.8. Limited Evaluation on Larger Models

Much of the experimentation uses relatively small models (e.g., 360M parameters), which may not fully reflect how confidence signals behave at scale. This creates a research gap in validating the approach on larger, real-world LLMs.

CHAPTER 5

OBJECTIVE

5.1 Primary Goal

The primary goal of this project is to architect and implement an intelligent, multi-stage query processing system that proactively assesses hallucination risks in real-time. Unlike traditional static pipelines, this system utilizes a dynamic routing mechanism to direct user queries to the most appropriate response pathway—optimizing for a balance between computational efficiency, response latency, and factual reliability. The system is specifically designed to function within the constraints of a "Small Local Model" environment (approx. 1B parameters), ensuring accessibility and feasibility on standard hardware.

5.2 Specific Technical Objective

5.2.1. Objective 1: Development of the Internal Confidence Module

The first technical objective is to engineer a robust Internal Confidence Module that serves as the initial "gatekeeper" for the system. This module operates strictly on the local model's internal states to evaluate uncertainty without incurring the latency of external retrieval. It computes a confidence score based on three distinct signal layers:

- **Semantic Alignment:** This component evaluates the consistency of the generated text by analysing semantic variations across multiple sampled outputs. It detects instances where the model vacillates between different meanings, indicating low confidence.
- **Internal Convergence:** This metric measures the statistical agreement of the model's internal probability distributions (logits). High convergence suggests the model is certain about its next-token predictions, whereas low convergence indicates ambiguity.

- **Learned Confidence:** This involves a trained regression layer or calibration mechanism that outputs a scalar confidence score derived directly from the model's training metrics and hidden states.

Risk Factor	Definition	Detection Method (Phase 2 Plan)
Context Scarcity	When the retrieved information is too sparse to answer the query.	Algorithms to measure information density in retrieved context.
Context Conflict	When source documents contradict one another, confusing the model.	Integration of Natural Language Inference (NLI) to detect contradictions.
Domain Mismatch	When the retrieved documents are irrelevant to the user's specific query.	Semantic similarity checks between query embeddings and document embeddings.

Table 5.1 Types of Signals

5.2.2. Objective 2: Implementation of a Conditional External Reality Check

The second objective is to implement a secondary validation layer known as the External Reality Check. Crucially, this module is designed to be conditional—it is triggered only when the Internal Confidence Module returns an ambiguous ("Medium") score. This architectural decision minimizes unnecessary API calls and latency. When activated, this module assesses the quality of retrieved external data against three key risk factors:

Context Scarcity: Detects scenarios where the retrieved information is too sparse or insufficient to definitively answer the user's query.

Context Conflict: Utilizes Natural Language Inference (NLI) techniques to identify situations where source documents contradict one another, which typically leads to model confusion.

Domain Mismatch: Performs semantic similarity checks to flag instances where retrieved documents are thematically irrelevant to the specific user query.

5.2.3. Objective 3: Engineering a Dynamic Routing & Decision Logic

The third and central objective is to develop the Dynamic Routing System, a logic-driven engine that orchestrates the flow of data based on the scores derived from the previous two objectives. The routing logic is defined by the following decision matrix:

Pathway A: Direct Generation (High Internal Score) If the Internal Confidence Module returns a High Score, the system determines that the local model possesses sufficient internalized knowledge. The query is routed immediately to the Small Local Model for generation. This is the most efficient pathway, offering the lowest latency.

Pathway B: Cloud Escalation (Low Internal OR Low External Score) This pathway acts as a failsafe mechanism. It is triggered in two specific scenarios:

Immediate Fail: If the initial Internal Score is Low**.**

Secondary Fail: If the Internal Score was Medium, but the subsequent External Reality Check yielded a Low Score (indicating poor retrieval quality). In both cases, the query is deemed too high-risk for the local system and is escalated to a Large Cloud Model (e.g., GPT-4 or similar) to ensure response quality.

Pathway C: RAG Generation (Verified Augmentation) This pathway is reserved for queries where the local model is uncertain (Medium Internal Score) but high-quality external data is available (High External Score). The system proceeds with Retrieval-Augmented Generation (RAG), grounding the response in the verified external context.

5.2.4. Objective 4: Creation of a Custom Hallucination Benchmark

The fourth objective focuses on data infrastructure. To rigorously train and validate the risk assessment modules, a custom labelled dataset will be constructed. This process involves:

- Aggregating prompts from established QA benchmarks such as Natural Questions and Trivia QA.
- Generating responses using the target small language models.
- Manually or automatically labelling these responses for specific hallucination types (e.g., factual fabrication, reasoning errors) to establish a "ground truth" for the confidence modules.

5.2.5. Objective 5: System Integration and Performance Benchmarking

The final objective is the holistic integration of the Internal Module, External Reality Check, and Routing Logic into a unified API or application pipeline. The system will be benchmarked against standard static RAG pipelines to demonstrate improvements in:

Accuracy: Reduction in hallucination rates.

Latency: Average time-to-response (proving the efficiency of the "Direct Generation" path).

Computational Cost: Reduced reliance on expensive cloud model tokens.

5.3. Project Scope

The scope of this research is strictly defined within the domain of text-based, knowledge-intensive Question Answering (QA). The architecture focuses on optimizing the performance of moderately-sized open-source Large Language Models (LLMs), specifically verifying their utility in resource-constrained environments where full-scale cloud dependence is not feasible.

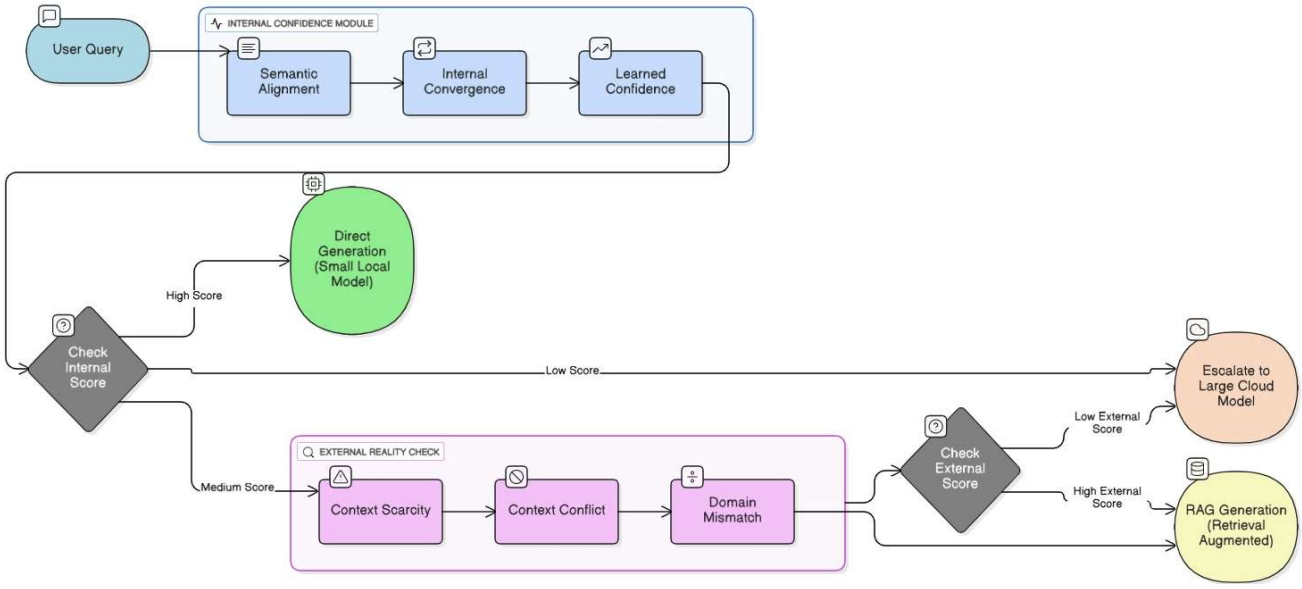

Figure 4.1 Conditional Hybrid Confidence System (3-Interval)

CHAPTER 6

CONCLUSION OF CAPSTONE PROJECT PHASE - 1

In this first phase of the project, we have successfully defined the scope and methodology for building a **Cost-Effective, Proactive Hallucination Routing System**. Our work to date has established the following:

- **Problem Validation:** Through an extensive literature survey, we validated that current hallucination mitigation strategies are largely reactive and computationally expensive, often attempting to fix errors after they occur.
- **Proposed Innovation:** We have designed a solution that improves upon the "self-aware" base paper by making it "context-aware." We introduced a novel **External Reality Check** layer to assess risks such as Context Scarcity, Context Conflict, and Domain Mismatch.
- **Feasibility Confirmation:** We conducted a feasibility study confirming that the system is computationally viable. By utilizing smaller LLMs (360M parameters) and open-source tools, the project can be implemented using standard university hardware.
- **Strategic Roadmap:** We have outlined a clear path for Phase 2, which includes creating a custom labelled dataset from QA benchmarks and training the confidence module to dynamically route queries between Direct Generation, RAG, and Human Review.

Final Summary: This project bridges the gap between model efficiency and reliability. By shifting from post-hoc correction to pre-generation risk assessment, we aim to create a system that allows LLMs to be safely deployed in high-stakes domains like healthcare and finance without the inefficiencies of "always-on" RAG.

CHAPTER 7

PLAN OF WORK FOR CAPSTONE PROJECT PHASE - 2

7.1. Dataset Preparation & Pre-processing

- **Objective:** Create the ground truth required to train the risk assessment model.
- **Action Items:**
    - **Acquire Benchmarks:** Download and set up existing QA benchmarks such as Natural Questions and Trivia QA.
    - **Generate Responses:** Run these questions through the selected small LLM (approx. 360M parameters) to generate initial responses[2].
    - **Labelling:** Create a custom dataset by labelling these generated responses for hallucinations. This involves categorizing answers as factual, hallucinated, or ambiguous to serve as training data.

7.2. Development of the Confidence Module

- **Objective:** Build the hybrid system that analyses both internal model states and external data quality.
- **Action Items:**
    - **Internal Signal Extraction:** Implement the logic to extract the three internal signals defined in the base paper: Semantic Alignment, Internal Convergence, and Learned Confidence.
    - **External Reality Check Implementation:** Develop the new "Context-Aware" layer to detect your three specific risk factors:
        - **Context Scarcity:** Algorithms to measure if retrieved context is too sparse.
        - **Context Conflict:** Integration of Natural Language Inference (NLI) to detect contradictions in source documents.

- **Domain Mismatch:** Semantic similarity checks to verify relevance between query and documents.

7.3. Routing System Logic & Integration

- **Objective:** Develop the dynamic decision-making engine.
- **Action Items:**
    - **Score Aggregation:** Create a formula or classifier that combines the Internal and External scores into a single Confidence Score.
    - **Thresholding:** Define the thresholds that determine which path a query takes:
        - High Confidence --> Direct Generation
        - Medium Confidence/Need Grounding --> RAG
        - High Risk/Conflict -->Large Cloud Model.

7.4. System Evaluation & Benchmarking

- **Objective:** Prove the effectiveness of the system against established baselines.
- **Action Items:**
    - **Baseline Comparison:** Compare your "Proactive" system against "Reactive" methods (like SelfCheckGPT) and "Always-on RAG" in terms of computational cost and accuracy.
    - **Performance Metrics:** Measure:
        - Accuracy: Reduction in hallucination rates.
        - Efficiency: Time taken per query and computational overhead.
        - Routing Precision: How accurately the system identifies when to use RAG vs. Direct Generation.

7.5. Documentation & Final Presentation

- **Objective:** Compile findings into a research paper and final defence.

- **Action Items:**
    - Draft the final report including the methodology, experimental setup, and results.
    - Prepare the Phase 2 Final Presentation.
    - (Optional Goal) Prepare the work for publication as mentioned in your project scope references.

Phase / Module	Objective	Key Action Items
Dataset Prep	Create ground truth for training the risk model	1. Acquire benchmarks (Natural Questions, Trivia QA) 2. Generate responses using 360M param model. 3. Label responses (Factual vs. Hallucinated)
Confidence Module	Build the hybrid internal/external analysis system	1. Extract internal signals (Convergence, Alignment) 2. Develop "Context-Aware" layer for external risks
Routing Logic	Develop the dynamic decision engine.	1. Create formula to aggregate Internal + External scores 2. Define thresholds for Direct Gen vs. RAG vs. Human Review
Evaluation	Validate system against baselines.	1. Compare against SelfCheckGPT and Always-on RAG 2. Measure Accuracy, Efficiency, and Routing Precision

Table 7.1 Plan of Work

CHAPTER 8

REFERENCES/BIBLIOGRAPHY

Primary References

- N. M, "Confidence-Aware Routing for Large Language Model Reliability Enhancement: A Multi-Signal Approach to Pre-Generation Hallucination Mitigation," *arXiv preprint arXiv:2510.01237*, 2025.
- P. Manakul, M. Gales, and A. Prom-on, "SelfCheck GPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models," in Proc. EMNLP, 2023, pp. 9004-9017.

Additional Key Works Cited in Literature Survey

- **RAG Foundation:** Lewis et al., 2020, "Retrieval-Augmented Generation (RAG)".
- **Hallucination Surveys:** Ji et al., 2023; Huang et al., 2023.

APPENDIX A DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

1. Acronyms and Abbreviations

- ESA: End Semester Assessment.
- ISA: In-Semester Assessment.
- LLM: Large Language Model.
- NLI: Natural Language Inference.
- NLP: Natural Language Processing.
- QA: Question Answering.
- RAG: Retrieval-Augmented Generation.
- SRN: Student Registration Number.

2. Definitions of Key Terms

- **Context Conflict:** A risk factor where contradictions exist between different source documents, potentially confusing the model.
- **Context Scarcity:** A condition where the retrieved context is too sparse or brief, forcing the LLM to guess and fill in blanks from its internal memory.
- **Domain Mismatch:** A situation where retrieved documents are not semantically relevant to the user's specific query.
- **Hallucination:** A phenomenon where Large Language Models produce content that appears plausible but is factually incorrect, undermining reliability.
- **Proactive System:** A proposed system designed to efficiently predict the risk of hallucination *before* generating a response, rather than correcting it afterward.
- **Reactive Methods:** Current solutions that attempt to detect or fix hallucinations only after the text generation process is complete.

- **Retrieval-Augmented Generation (RAG):** A technique that grounds LLM responses in external documents to improve factuality**.**
- **Routing System:** An intelligent mechanism that assesses risk and dynamically directs queries to the most appropriate pathway (e.g., direct generation vs. RAG vs. human review).
- **SelfCheck GPT:** A specific reactive method that checks for consistency across multiple generated answers to detect hallucinations.