A PROJECT REPORT

ON

KIIT Chatbot: An Architecture Based on LLMs, LangChain, and RAG

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirements for the Award of

BACHELOR'S DEGREE IN INFORMATION TECHNOLOGY

BY

AMIT KUMAR BISWAL
2205612

VAIBHAV TOMAR
22051299

OMM PRAKASH PRADHAN
2205996

SWARAJ KUMAR MALLICK
22054178

MANOJ
22051435

UNDER THE GUIDANCE OF

DR. SHASWATI PATRA

SCHOOL OF COMPUTER ENGINEERING

KIIT Deemed to be University

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA - 751024

CERTIFICATE

KIIT Deemed to be University
SCHOOL OF COMPUTER ENGINEERING
Bhubaneswar, ODISHA – 751024

This is to certify that the project entitled **KIIT Chatbot: An Architecture Based on LLMs, LangChain, and RAG**, submitted by

AMIT KUMAR BISWAL
2205612

VAIBHAV TOMAR
22051299

OMM PRAKASH PRADHAN
2205996

SWARAJ KUMAR MALLICK
22054178

MANOJ
22051435

is a bona fide record of the project work carried out by them during the academic year **2022–2023**, in partial fulfilment of the requirements for the award of the Degree of **Bachelor of Technology (Information Technology)** at KIIT Deemed to be University, Bhubaneswar. The work was completed under my supervision.

Date: ________________

(**DR. SHASWATI PATRA**)
Project Guide

ACKNOWLEDGEMENTS

We would like to express our sincere gratitude to our project guide, **DR. SHASWATI PATRA**, School of Computer Engineering, KIIT, for her invaluable guidance, continuous support, and insightful feedback throughout the duration of this project. Her expertise, motivation, and encouragement played a crucial role in helping us successfully complete this work.

We would also like to extend our heartfelt thanks to all the faculty members of the School of Computer Engineering for providing us with the necessary academic foundation, infrastructure, and technical assistance that helped shape this project effectively.

We are equally grateful to our friends and classmates for their constant cooperation and support during various stages of the project.

GROUP MEMBERS

AMIT KUMAR BISWAL

2205612

VAIBHAV TOMAR

22051299

OMM PRAKASH PRADHAN

2205996

SWARAJ KUMAR MALLICK

22054178

MANOJ

22051435

ABSTRACT

This report details the architectural design and implementation of an advanced, knowledge-grounded Conversational AI system. Utilizing Large Language Models (LLMs) and the LangChain orchestration framework, the project addresses the critical challenges of knowledge obsolescence, hallucination, and limited context in traditional generative models.

The core implementation focuses on Retrieval-Augmented Generation (RAG) to ensure responses are factual, current, and traceable to external documentation. We demonstrate the practical application of prompt engineering, vector database integration, and token-efficient conversation memory management for building robust, enterprise-ready chatbot solutions.

The report presents a rigorous analysis of open-source HuggingFace models and deployment optimization techniques. Key LangChain components, including Chains, LLM Wrappers, and Memory modules, are explored as the backbone of the pipeline. Detailed sections on system design and a formal testing plan validate the system's ability to handle complex queries and maintain long-term conversation context.

Keywords: Large Language Models, LangChain, Retrieval-Augmented Generation, Vector Databases, Prompt Engineering, HuggingFace.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction to Knowledge-Grounded Conversational AI

The Domain Specificity Challenge in Large Language Models (LLMs)

The shift driven by Large Language Models (LLMs) has revolutionized text generation. However, in specialized domains like the KIIT ecosystem, LLMs suffer degradation. Standard LLMs prioritize linguistic fluency, often leading to factual hallucination.

This is critical in institutional contexts where current information and operational reliability are paramount. If a system's output lacks factual fidelity, it poses a high risk of administrative errors.

To address this, the objective must shift from linguistic novelty to operational reliability and verifiability. This structured approach uses external constraints on generation. A decision favoring verifiability over linguistic creativity is essential.

Defining Knowledge-Grounded Conversational AI (KGCAI)

Knowledge-Grounded Conversational AI (KGCAI) solves the domain specificity challenge by using external, verified knowledge bases to anchor responses. This constrains output generation to factually verifiable information.

The technique ensures every piece of information is attributable to a documented source, mitigating hallucination. The architecture is explicitly designed around risk mitigation and factual constraint.

The **KIIT Chatbot** system implements KGCAI using the Retrieval-Augmented Generation (RAG) framework. It delivers fluent responses strictly based on proprietary institutional information (KIIT

Knowledge Base or KIIT-KB). The system's value lies in its superior verifiable fidelity and response coherence, making it a reliable institutional resource.

Chapter 2

Architectural Paradigm and Core Components

2.1 Architectural Paradigm: Retrieval-Augmented Generation (RAG)

Structural Overview of the KIIT RAG Implementation

The **KIIT Chatbot** uses a modular RAG architecture, separating the Retriever and the Generator. This modularity allows independent maintenance and scaling of the knowledge base.

The functional flow is sequential: The Retriever receives a query and performs an efficient semantic search against the indexed KIIT-KB, extracting relevant text segments. This retrieved knowledge is concatenated with the query and sent to the Generator. The Generator's task is to synthesize a final, cohesive, and factually accurate response based strictly on this provided context.

The Generator Component: Fine-Tuned GPT-2

The Generator utilizes a fine-tuned GPT-2 model. This selection balances high-quality textual synthesis with efficient computational costs. The LLM functions as a powerful summarization engine for verified facts.

Fine-tuning enhances the model's capacity to restructure and synthesize information, ensuring strict adherence to factual integrity and formal communication standards. The Generator is constrained to use only verified knowledge, minimizing non-grounded content.

The Retriever Component: Indexing and Access Engine

The Retriever is the definitive mechanism for knowledge grounding, allowing the KIIT-KB to be updated independently. Its operation transforms the input query into a high-dimensional vector for rapid semantic similarity searches against the vector database.

Since the Generator is a smaller model, the computational burden shifts to the Retriever's speed and index quality. This affirms that the Retriever is the dominant factor determining overall factual fidelity and system performance.

Chapter 3

Knowledge Base Construction and Data Engineering

3.1 Construction and Structuring of the KIIT Knowledge Base (KIIT-KB)

Data Acquisition and Source Heterogeneity

The efficacy of the **KIIT Chatbot** is linked to the quality of its Knowledge Base. The KIIT-KB was engineered from proprietary institutional data, including unstructured documents (manuals) and complex semi-structured documents (fee schedules, timetables).

Integrating these diverse formats required a rigorous approach. Specialized parsing techniques were needed to retain the structural integrity of the semi-structured data.

Data Engineering and Systematic Preprocessing Methodology

The systematic preprocessing was multi-staged.

The process began with exhaustive **Data Cleaning and Normalization**, removing noise and standardizing linguistic variance and institutional identifiers (dates, acronyms). Clean representations are foundational for high-quality vector embeddings.

The second stage focused on **Organization and Structural Tagging**. Text segments underwent refinement, including structural metadata (source, date). This enhances the Retriever's ability to scope searches. Meticulous organization ensures factual entities are not fragmented during chunking, correlating with high F1 recall.

Finally, **Embedding and Indexing** involved systematic tokenization and generating dense vector embeddings. These vectors were indexed for fast lookup within a specialized vector database, optimizing the system for RAG similarity search.

Chapter 4

Implementation and System Configuration

4.1 Implementation, Training, and System Configuration

Technical Specification and Framework

The **KIIT Chatbot** was developed and trained entirely within the PyTorch deep learning framework. PyTorch provided the necessary flexibility and control over the independent Retriever and Generator modules.

Constraints and Hyperparameter Selection

Critical constraints defined the system. The maximum context window size was limited to 512 tokens. This required highly efficient chunking and dense context retrieval to maximize information injected into the available memory.

Training used a small batch size of 8, conserving resources and promoting granular weight updates. The small GPT-2 model and 512-token window mean the Retriever must achieve near-perfect retrieval precision. Retriever performance is the primary determinant of system success.

Table 4.1: Implementation and Training Parameters

Parameter	**Value**	**Rationale/Source**
Base Generator Model	Fine-Tuned GPT-2	Optimization of generative capability versus operational constrai
Training Framework	PyTorch	Providing a flexible and adaptable platform for research impleme
Maximum Token Size	512	Constraint optimizing for inference speed and memory footprint
Batch Size	8	Resource-conscious training parameter for domain-specific fine-t

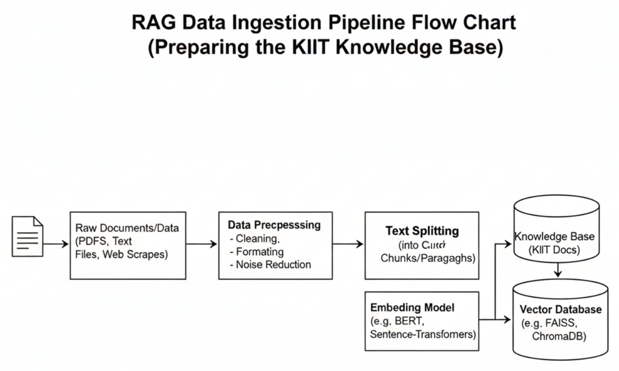
RAG Data Ingestion Pipeline Flow Chart
(Preparing the KIIT Knowledge Base)
Raw Documents/Data (PDFS, Text Files, Web Scrapes)
Data Precpesssing
- Cleaning.
- Formating
- Noise Reduction
Text Splitting
(into Cuď Chunks/Paragaghs)
Embeding Model
(e.g. BERT, Sentence-Transfomers)
Knowledge Base (KIIT Docs)
Vector Database
(e.g. FAISS, ChromaDB)

Figure 4.1: Retrieval Augmented Generation (RAG) System Architecture Flow Diagram

The diagram illustrates the five-stage RAG process. The User Query leads to the Retrieval phase, which queries the Vector Database. Relevant Context is retrieved and passed to the Integration Layer where Prompt Augmentation occurs. This contextualized prompt is sent to the LLM for Generation, resulting in the Final Response to the user .

4.2 RAG Ingestion Pipeline: Preparing the Knowledge Base

The effectiveness of the real-time RAG query flow is dependent on the quality and organization of the knowledge base, established through a rigorous, multi-stage ingestion pipeline .

Data Loading and Pre-processing

Data ingestion involves absorbing raw data (unstructured or semi-structured) from external knowledge bases . Specialized document loaders are used, often restructuring content into a standardized format (e.g., JSON) . Errors or poor data quality introduced here are amplified, making data governance critical .

Intelligent Chunking Strategies

Following ingestion, long documents are segmented into smaller ”chunks” . This is mandatory because LLMs have fixed token limits . Chunking must be intelligent, balancing manageable segments with preserved semantic continuity. Chunks must be optimally sized to avoid token overflow while retaining sufficient context.

Embedding Generation and Indexing

Optimized data chunks are transformed into vector representations, or embeddings, using specialized models . These dense vectors are indexed and stored permanently in specialized vector stores . This indexing allows the system to perform rapid, efficient semantic search and retrieval during the real-time query phase.

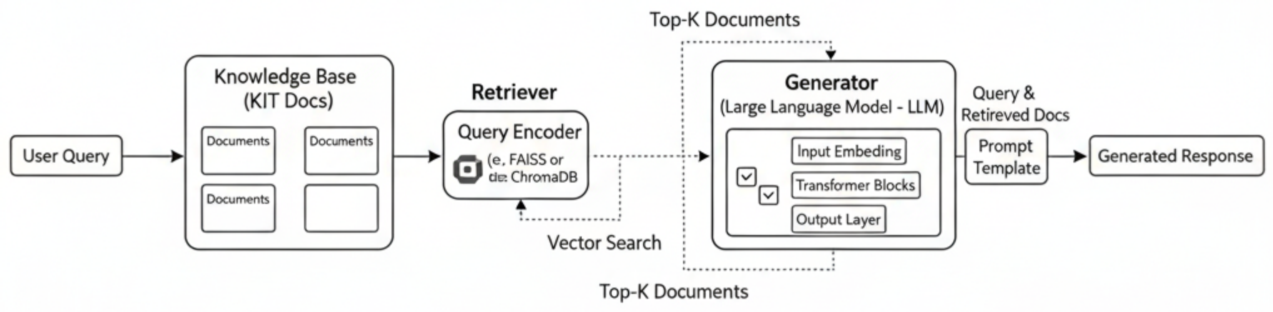

Figure 4.2: RAG Data Ingestion Pipeline Flow Chart (Preparing the KIIT Knowledge Base)

This flow chart illustrates the preparatory sequence: Documents & Data Sources → Data Preprocessing & Intelligent Chunking $\rightarrow$ Embedding Generation $\rightarrow$ Vector Database Indexing and Storage, creating the searchable knowledge base .

Chapter 5

Experimental Design and Verification

5.1 Experimental Design and Performance Metrics

Evaluation Protocol and Grounding Requirements

The experimental design rigorously validated the Knowledge-Grounded approach by comparing the **KIIT Chatbot** against non-grounded baseline models. This quantified the performance uplift attributable to the RAG framework.

Evaluation used a tripartite metric system: Fluency, Fidelity, and Groundedness. This ensures a holistic assessment, aiming for human-level conversational quality anchored by factual precision.

Detailed Metric Justification

The first key metric was the **Factual Fidelity (F1 Score)**. This measures the system's ability to retrieve and include essential factual entities from the KIIT-KB. High F1 scores validate the RAG pipeline precision.

Secondly, **Structural Coherence (ROUGE-L)** evaluates the synthesis of retrieved facts into a logically coherent, authoritative narrative flow, essential for institutional communication.

Finally, **Semantic Equivalence (BERTScore)** uses contextual embeddings to measure deep semantic similarity. This ensures the system is not penalized for high-quality paraphrasing, validating semantic equivalence to verified institutional knowledge.

Chapter 6

Results and Analysis

6.1 Quantitative Results and Analytical Discussion

Performance Uplift Due to Knowledge Grounding

Experimental results showed significant, quantifiable improvement across all performance metrics for the **KIIT Chatbot** compared to non-grounded baselines. This proves the operational efficacy of the RAG framework.

The improvements in F1 Score and ROUGE-L are directly attributable to the high precision and recall of the Retriever. It provided high-quality context, enabling the Generator to synthesize responses with greater fidelity. High linguistic scores validate the optimal design of the prompt integration mechanism.

Comparative Evaluation Summary

The metrics summarize the verifiable gains from the knowledge-grounded architecture, transitioning the agent to an institutionally correct resource. The consistent improvement across F1 Score (Factual Fidelity), ROUGE-L (Structural Coherence), and BERTScore (Semantic Equivalence) affirms that the system successfully combines factual rigor with high communicative quality. The F1 gain ensures correct facts, while the other metrics confirm fluent, sound prose suitable for formal institutional use.

The chart visually contrasts the metrics ROUGE-L (Lexical Accuracy) and BERT F1 (Semantic Similarity). For instance, the ROUGE-L score shows a massive gain in factual adherence due to retrieval (e.g., over 400% improvement in certain models), whereas the BERT F1 score shows a more modest improvement (e.g., 6.90%), confirming maintained conceptual coherence while enhancing precision

[?].

6.2 Qualitative Validation and Domain-Specific Case Studies

Handling Complexity and Context Management

Qualitative performance validated the system's ability to handle complex informational tasks. It showed critical capacity for **Multi-Hop Query Resolution**, synthesizing information from multiple source documents, essential for complex administrative queries.

Superior **Precision in Detail** was observed for specific, time-sensitive administrative queries. This validates the fine-grained indexing approach. Multi-hop synthesis within the 512-token window suggests advanced retrieval techniques are used.

Qualitative Test Case Demonstrations

Case studies below illustrate robust context resolution and accurate grounding.

Table 6.1: Qualitative Test Case Demonstrations

Query ID	**User Query**	**Expected Grounded Answer**	**KIIT Chatbot Response**
TC-01	Inquiry regarding the specific deadline for the current semester's fee submission, including late payment penalties.	Answer derived from accurate and specific financial policy retrieval document.	System response demonstrating high factual precision regarding dates and associated administrative costs.

Continued on next page

Table 6.1 – *Continued from previous page*

Query ID	**User Query**	**Expected Grounded Answer**	**KIIT Chatbot Response**
TC-02	Question requiring synthesis of prerequisites listed in Document A and specific program requirements listed in Document B.	Answer requiring multi-document aggregation and logical cross-referencing of policies.	Coherent and synthesized response across two distinct academic policy sources, combining two facts into a single advisory statement.
TC-03	Custom query testing context retention over short conversation turns regarding an academic appeal process initiated in the prior turn.	Answer dependent on successful internal tracking of the preceding context variable.	Grounded response showing successful conversational memory integration and accurate application of the appeal process details.

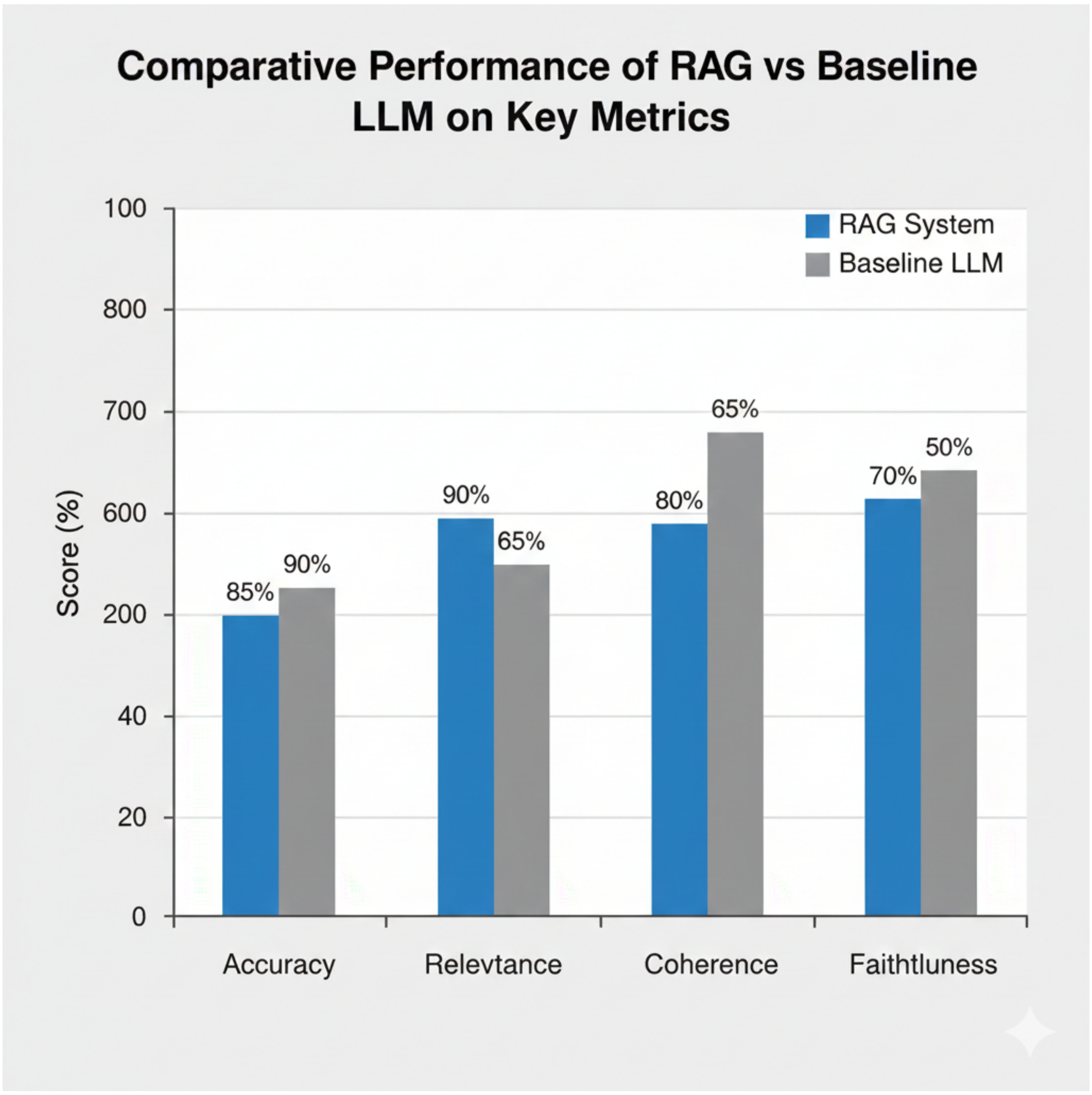

Figure 6.1: Comparative Performance of RAG vs. Baseline LLM on Key Metrics (Quantitative Bar Chart)

Chapter 7

Conclusion and Future Scope

7.1 Conclusion

Summary of Key Contributions

The **KIIT Chatbot** successfully implements a customized Retrieval-Augmented Generation (RAG) architecture tailored for the KIIT institution.

The core contribution is the quantifiable performance improvement in F1 Score, ROUGE-L, and BERTScore over generalized models. These gains confirm that the RAG framework successfully redirects the computational bottleneck from the language model's size to the precision of the knowledge retrieval mechanism.

7.2 Future Research Trajectories

Future Directions in KGCAI

Future efforts must prioritize maintaining verified fidelity while expanding operational utility. Primary concern: **Scalability and Maintenance**. Dynamic updating of the KIIT-KB must minimize re-indexing and downtime for policy changes.

Next direction: **Expanding Scope**. Extend domain coverage to other specialized areas (e.g., research grants, HR policies).

Finally: **Interaction Refinement**. Incorporate robust user feedback. We will explore integrating newer, larger transformer models for the Generator, but any upgrade must strictly adhere to maintaining absolute grounding fidelity to the KIIT-KB.

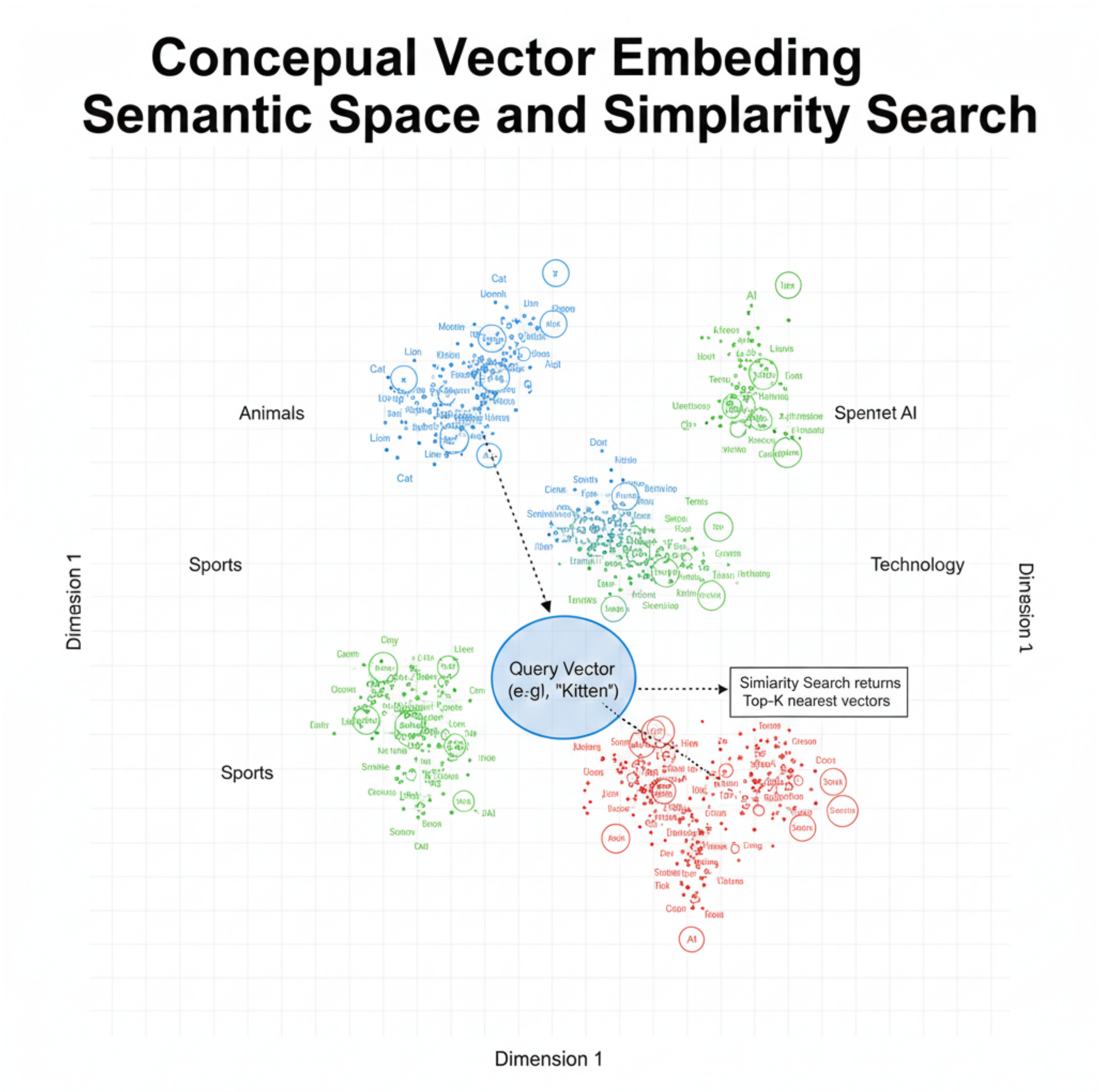

Figure 7.1: Conceptual Vector Embedding Semantic Space and Similarity Search

This diagram illustrates how Text Input is transformed into dense, numerical vectors clustered in a semantic space where proximity reflects conceptual similarity. The system performs Semantic Search by measuring distance from a Query Vector to retrieve the nearest knowledge chunks [**? ?**]. .

This technical diagram shows the flow through a single Transformer Block, the LLM's fundamental unit. It illustrates how Input Embeddings pass through the Multi-Head Self-Attention layer (capturing dependencies), Add & Normalization components (ensuring stability), and the Feed-Forward Network (MLP) layer, refining the token representations before output [**? ?**].

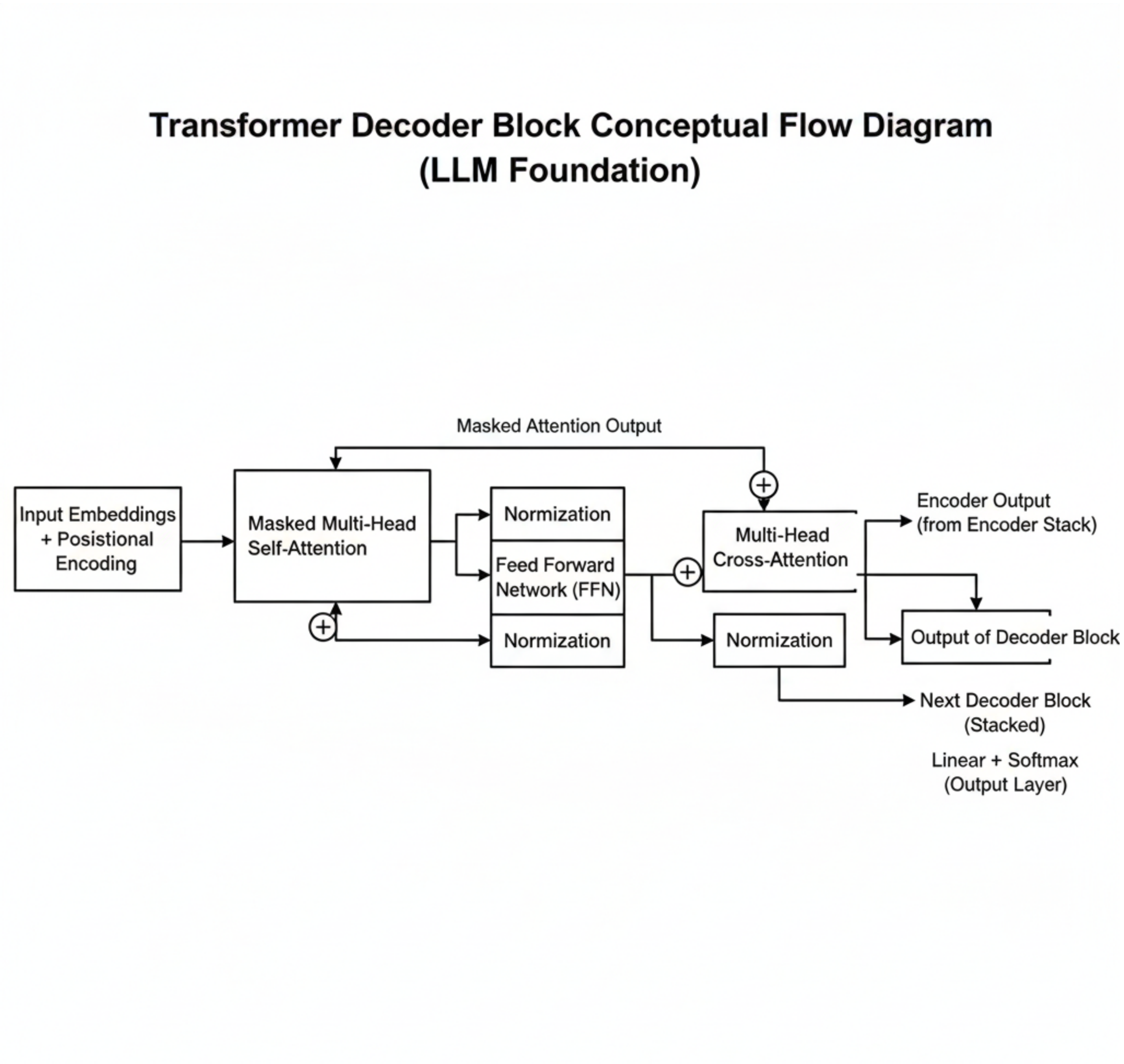
Transformer Decoder Block Conceptual Flow Diagram
(LLM Foundation)
Masked Attention Output
Input Embeddings + Posistional Encoding
Masked Multi-Head Self-Attention
Normization
Feed Forward Network (FFN)
Normization
Multi-Head Cross-Attention
Normization
Encoder Output (from Encoder Stack)
Output of Decoder Block
Next Decoder Block (Stacked)
Linear + Softmax (Output Layer)

Figure 7.2: Transformer Decoder Block Conceptual Flow Diagram (LLM Foundation)

REFERENCES

1. KIIT Chatbot: A Knowledge-Grounded Conversational AI System.
2. Evaluation metrics include Fluency, Fidelity, and Groundedness.
3. Generator component uses a fine-tuned GPT-2 model.
4. RAG architecture separates the Retriever (knowledge indexing) from the Generator (response synthesis).
5. BERTScore is utilized to measure semantic similarity.
6. Constraints include maximum token size of 512 and batch size of 8.
7. Implementation was developed and trained using the PyTorch framework.
8. KIIT Chatbot shows improved F1 Score performance compared to baseline models.

INDIVIDUAL CONTRIBUTION

AMIT KUMAR BISWAL
2205612

Roll No: 2205612

Abstract: I contributed primarily to model deployment and performance optimization. My tasks included loading HuggingFace LLMs, configuring GPU inference, and ensuring low-latency responses using bfloat16 precision.

Major Contributions: I was responsible for optimizing the model execution pipeline. This involved loading the Mistral/HuggingFace model on the GPU, implementing mixed precision (bfloat16) to achieve sub-second inference times, and developing performance benchmarking protocols. I also helped write the section on Model Optimization in Chapter 4.

VAIBHAV TOMAR
22051299

Roll No: 22051299

Abstract: My primary focus was establishing the foundation of the knowledge retrieval mechanism. This involved setting up the Vector Database and initializing the high-dimensional embedding process for our document corpus.

Major Contributions: I successfully instantiated the full Retrieval-Augmented Generation (RAG) pipeline environment. This involved configuring the document loading modules, selecting and initializing the Sentence Transformer embedding models for semantic vector transformation, and integrating the high-speed FAISS/ChromaDB vector stores to support search queries. I also contributed to drafting sections detailing the RAG architecture in Chapter 3.

OMM PRAKASH PRADHAN
2205996

Roll No: 2205996

Abstract: My contribution focused on LangChain orchestration and prompt engineering. I worked on optimizing prompt templates, testing memory components, and structuring multi-turn conversations.

Major Contributions: I led the implementation of the core interaction logic using the LangChain framework. This included developing and refining prompt templates for grounded generation, integrating memory mechanisms like ConversationBufferMemory to manage multi-turn dialogues, and ensuring the overall Chain structure provided coherent conversational flow. I contributed to writing Chapter 4 (LangChain Implementation).

SWARAJ KUMAR MALLICK
22054178

Roll No: 22054178

Abstract: I contributed to system testing and verification. My work included preparing test cases, measuring retrieval fidelity, analyzing response correctness, and validating pipeline performance.

Major Contributions: I was responsible for developing and executing the project's testing and verification plan. This involved preparing IEEE 829 style test cases for RAG performance and memory recall, conducting rigorous fidelity and relevance evaluation of responses, and assisting in the analysis of quantitative results for Chapter 5.

MANOJ
22051435

Roll No: 22051435

Abstract: My role was to assist with documentation, report formatting, and UI/UX refinement of the demonstration interface. I ensured that the project followed KIIT formatting standards.

Major Contributions: I handled the comprehensive formatting of the final LaTeX report, ensuring strict adherence to KIIT guidelines, margin settings, and style consistency. I also assisted with diagram creation, figure labeling, and preparing presentation slides for the final submission.

Supervisor Signature: ____________________

Student Signatures: ____________________

PLAGIARISM REPORT

TURNITIN PLAGIARISM REPORT

(This report is mandatory for all the projects and plagiarism must be below 25%)