The Conductor in the Machine: An Analysis of Retrieval Strategies for Multi-Tool Enterprise AI Agents

Project Objective or Aim

This research aims to investigate and evaluate different strategies for routing user queries within a multi-tool enterprise AI agent. As companies increasingly adopt Large Language Models (LLMs) that can interact with various data sources (e.g., document databases, SQL databases, graph databases), the method used to select the correct data retrieval tool becomes critical for performance. This project will answer the following research question: What is the most effective and efficient strategy for routing queries in a multi-tool, Retrieval-Augmented Generation (RAG) system to maximize accuracy while minimizing latency and computational cost? This study will implement and benchmark several routing mechanisms to provide a quantitative analysis of their respective trade-offs in a simulated enterprise environment.

Project Background and Significance

The integration of Large Language Models into enterprise workflows has created a powerful new paradigm for knowledge retrieval. Systems based on Retrieval-Augmented Generation (RAG), first introduced by Lewis et al. (2020), have proven highly effective by providing LLMs with relevant, factual information from external knowledge bases before generating a response. This grounds the model's output in reality and reduces hallucinations. The foundation of these systems is a dense retrieval model, such as the Dense Passage Retriever (DPR) developed by Karpukhin et al. (2020), which excels at finding relevant text passages from large document corpora.

However, enterprise data is not monolithic; it exists in various formats across different systems. A single company may have unstructured documents (reports, emails), structured data in SQL databases (sales figures, customer info), and graph databases (organizational charts, supply chains). To address this, developers are building multi-tool AI agents that can dynamically choose the correct data source for a given query. While the underlying models for language understanding have evolved significantly since the introduction of architectures like BERT (Devlin et al., 2018), a significant gap exists in the research on the routing mechanism itself. How should an agent decide whether to query a vector database, a SQL database, or a graph database, especially when a user's query is semantically ambiguous?

This decision is non-trivial and has significant implications for the accuracy, speed, and cost of the system. An incorrect routing decision can lead to irrelevant information being fed to the LLM, resulting in a wrong answer and eroding user trust. A slow or computationally expensive routing mechanism can render the system impractical for

real-time applications and lead to prohibitive operational costs. This project is significant because it will be one of the first undergraduate research efforts to systematically analyze and compare different routing strategies. By providing a foundational framework and a quantitative cost-benefit analysis, this research will help developers build more efficient, reliable, and trustworthy enterprise AI assistants, moving the field from ad-hoc implementations to a more principled engineering approach.

Research Methods

This project will be conducted in three distinct phases over the course of a 15-week semester.

Phase 1: Baseline Agent Implementation (Weeks 1-4) The first phase will involve building a baseline multi-tool AI agent using Python and the LangChain framework. This agent will integrate three distinct retrieval tools:

1. A dense passage retriever (DPR) connected to a Qdrant vector database for unstructured documents.

2. A SQL-agent capable of querying a DuckDB database containing structured tabular data.

3. A graph-retriever connected to a Neo4j database representing relational data. For the initial baseline, the routing logic will be a simple keyword-based classifier that attempts to route the query based on predefined rules.

Phase 2: Development of Advanced Routing Strategies (Weeks 5-9) In this phase, two more sophisticated routing strategies will be developed. The first will be an "LLM-as-a-router" approach, where a call is made to a powerful LLM (like GPT-4) that is prompted to analyze the user's query and decide which tool is most appropriate. The second will be a hybrid model that first uses a fine-tuned, open-source sentence-transformer model to quickly classify the query, only escalating to the more expensive LLM-as-a-router if the initial confidence score is low.

Phase 3: Benchmarking and Analysis (Weeks 10-15) The final phase will involve a rigorous evaluation of the three routing strategies (baseline, LLM-as-a-router, and hybrid). A custom benchmark dataset of 100 enterprise-style questions will be created, with each question designed to target a specific data source. Each routing strategy will be tested against this dataset, and performance will be measured across three key metrics:

1. **Accuracy:** The percentage of queries routed to the correct tool.

2. **Latency:** The average time taken from query input to tool selection.

3. **Cost:** The estimated monetary cost per query, based on API calls and compute time. The results will be compiled, analyzed, and visualized to draw conclusions about the trade-offs of each approach.

Expected Outcome

The primary deliverable of this research will be a formal research paper, suitable for submission to an undergraduate research journal or a relevant AI conference like the AAAI Undergraduate Consortium. This paper will detail the project's methodology, present the benchmark results in tables and figures, and provide a thorough analysis of the findings. A second deliverable will be a poster presentation summarizing the project, which will be presented at UCF's Student Scholar Symposium. To ensure the work has a lasting impact, all code, the custom benchmark dataset, and the trained models will be made publicly available in a GitHub repository.

This project will generate significant new knowledge for the field of applied AI. The findings will offer the first quantitative comparison of different routing strategies, providing practical, evidence-based guidance to developers on how to design more efficient and cost-effective systems. For example, the results could help a startup decide whether to invest in an expensive LLM-based router or if a cheaper, fine-tuned model offers a better return on investment. For the UCF community, this research will showcase the university's engagement with cutting-edge AI technology. The open-source repository will serve as a valuable tool for other students and researchers, potentially being used in senior design projects, graduate-level AI courses, or other research labs on campus. By providing both a foundational analysis and a practical toolkit, this work will help position UCF as a contributor to the rapidly evolving and highly competitive field of generative AI.

Literature Review

1. Chen, D., et al. (2017). Reading Wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

2. Devlin, J., et al. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

3. Guu, K., et al. (2020). REALM: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

4. Karpukhin, V., et al. (2020). Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

5. Lewis, P., et al. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems, 33*, 9459-9474.

6. Seo, M., et al. (2016). Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Preliminary Work and Experience

I have already completed a significant amount of preliminary work for this project through the development of "QueryAgent," a full-stack prototype of a smart enterprise assistant. In this project, I successfully built a multi-tool agent using Python and LangChain that could route queries to SQL and vector databases. I also implemented a Retrieval-Augmented Generation (RAG) pipeline and used Docker to containerize the application. This prior experience means I have already overcome many of the initial technical hurdles and have a solid codebase to build upon. My coursework in data structures and algorithms, and machine learning, combined with my professional experience as an AI Data Specialist, has provided me with the necessary skills in Python, AI frameworks, and database management to successfully execute this research.

IRB/IACUC statement

This project does not involve human or animal subjects. All research will be conducted using publicly available datasets and computational models. Therefore, this project does not require IRB or IACUC approval.

Budget

The total requested budget for this project is **$1,500.00**. These funds are essential for accessing the computational resources required for the benchmarking phase of the research.

Item	Description	Cost
OpenAI API Credits	Required for the “LLM-as-a-router” strategy and for generating final answers. Estimated cost based on ~1,000 complex queries for testing and development.	$1,000
Cloud Computing Resources	For hosting the Qdrant, DuckDB, and Neo4j databases on a cloud platform (e.g., AWS) and for running the benchmarking experiments.	$500
Total		$1,500