AI Research Assistant Agent

Capstone Project Proposal

Group Name: Solo Project

Team Members: Alhassane Samassekou

Date: April 7, 2025

Problem Statement

In the age of information overload, students, researchers, and professionals struggle to efficiently gather and organize information from reliable sources. Traditional research is time-consuming, and current tools often lack the ability to synthesize data meaningfully. This project proposes the development of an AI agent that helps users research topics, gather information from multiple sources, evaluate credibility, and present findings in a structured, citation-aware format. The agent will adapt its behavior over time using reinforcement learning principles to enhance performance and user satisfaction.

Project Option

Selected Option: Option 1 – Research Assistant Agent

This project will create an intelligent assistant capable of guiding users through complex research processes, making information retrieval smarter and more user-adaptive.

Agent Design

Agent Architecture:

- Input Processing: Natural language understanding to parse user questions.
- Memory System: Local session-based memory and vector database (e.g., FAISS).
- Reasoning Component: Chain-of-Thought reasoning pattern with planning-then-execution.
- Output Generation: Summarized and cited reports via natural language generation.

Reinforcement Learning Concepts:

- Feedback Mechanism: User responses provide feedback to improve performance.
- Reward System: Success scoring based on source credibility, user rating, and relevance.
- Policy Improvement: Updates to planning heuristics and citation formatting logic over time.

Safety and Security Measures:

- Input Validation: Prompt filtering for harmful/inappropriate content.
- Boundary Enforcement: Defined restrictions on agent responses.
- Fallback Strategies: Offers alternatives if unable to retrieve data.
- Transparency: Clearly explains actions, tools used, and limitations.

Tool Selection

Primary Tools:

- Google Search API / SerpAPI: Real-time information retrieval.
- GPT-3.5 / OpenAI API: Summarization, explanation, and citation formatting.
- FAISS (Vector Database): Stores embeddings for continuity and learning.

Tool Usage Flow:
- Trigger logic based on query type.
- Error Handling for API limits or failures.
- Interpretation and scoring of results before summarization.

Development Plan

Milestones:
- April 10: Finalize architecture & tools
- April 17: Implement core modules
- April 22: Integrate reinforcement logic
- April 25: Add safety mechanisms
- April 28: Internal testing & improvements
- May 3: Prepare final deliverables
- May 5: Final Submission

Evaluation Strategy

The agent's effectiveness will be evaluated using:
- Qualitative Feedback: User ratings on result usefulness.
- Quantitative Metrics: Accuracy, fallback rate, and summary compression.
- Adaptability: Learning across sessions.
- Safety Logs: Tracking input violations and fallback triggers.

Resource Requirements

Development Environment: Google Colab with Python 3.
Libraries: requests, transformers, faiss-cpu, langchain, openai, nltk.
Compute Needs: Occasional GPU access for NLP tasks (Colab free or Pro tier).

Risk Assessment

Risk: API Limits / Downtime → Mitigation: Caching, retries.
Risk: Output inaccuracies → Mitigation: Multi-source validation.
Risk: Safety filter bypass → Mitigation: Audit and update filters.
Risk: Feature creep → Mitigation: Freeze scope early.
Risk: Limited RL integration → Mitigation: Use heuristic-based learning loops.