Application of Multi-Agentic LLM's to Recommendation System

Mounika Atmakuri [EJ66944], EJ66944@umbc.edu

Sridhar Reddy Maram [LA32371], LA32371@umbc.edu

Tauseef Taufiq K [YJ06499], K327@umbc.edu

Surya Teja Avvaru [BO06455], suryata1@umbc.edu

ABSTRACT:

This paper proposes a Recommendation System built using a multi-agent LLM framework with LangChain consisting of five specialized agents: Flights, Tourist Locations, Location History, Hotel, and Weather. Recommender systems have experienced exponential growth in various fields, especially in the tourism sector, improving tourism activities' accuracy, personalization, and experience. Generally, in a multi-agent system, there are autonomous, independent LLM-powered agents that interact towards achieving a goal. Each of the agents has its own input for the model, its own tools, and its custom code. The Weather Agent displays real-time updates about the weather, the Hotel Agent recommends hotels, while the Flight Agent finds the customized flight options. Where the History Agent responds to facts about tourist location history, the Tourist Locations Agent suggests the best places to visit. Each of the agents acts independently using LLM functionalities, coordinated through the architecture of LangChain, to enable efficient travel planning. It describes methods for data processing, agent interaction, and the analysis of their performance. The results show that the framework is promising for comprehensive travel assistance.

Keywords: *Recommendation system, Multi-Agent LLM, Large Language Models, LangChain, Travel Assistance, Autonomous Agents*

1.INTRODUCTION:

Planning a trip is one of the most stressful experiences, as it takes a lot into consideration: what to see, where to stay, and how not to burst the budget. Nowadays, there is a great demand for intelligent systems that could simplify this process while making it as personalized as possible, since interest in tourism and travel became so high [3]. The project has focused on the implementation of the Multi-Agent LLM Framework using LangChain serving as a smart tour guide system [1][2].

This virtual travel assistant is an elaborate on real-time economic flight options, weather, budget-friendly hotel options, history about the place, and recommended tourist spots one should not miss, based on the destination and travel date. Therefore, the system will behave much like a human travel agent by harnessing the power of artificial intelligence in providing personalized, precise travel assistance [4][6].

Traditional Approach:

Early automatized travel planning solutions were rule-based. In such systems, lists of recommendations had to be prepared in advance, based on predefined sets of rules, which were quite good for structured and straightforward queries but fell short in scenarios where personalized or real-time updates were needed [3][5]. Due to this fact, the traveler would have to invest considerable efforts to manually adapt his or her plan to dynamic conditions or special preferences [7].

Advances with AI:

These days, travel planning has gone through a sea change with the introduction of AI and ML. The use of AI-powered systems developed personalization through the analysis of user behavior and preference. Various studies, such as Travel Agent, have been able to show how itinerary generation that is not only logical but agile to meet the needs of individuals could be done by large language models like GPT-4 [5]. These systems also have their drawbacks in the matter of handling real-time data and giving full-fledged travel planning [2][3].

Large Language Models:

Large language models, like GPT-4, have been pathbreaking both in text comprehension and in generating text that is no less human [4]. Research has proved beyond doubt that these models can support various domains, which include customer support and generating content [1]. For travel, LLMs are a great boon in the sense that they can analyze a large bulk of information and then develop coherent and relevant responses [7]. In most of these cases, however, the models are not capable of handling real-time information, a crucial component in any feasible travel planning [3].

2. LITERATURE REVIEW:

The concepts have gone a long way from the very idea of using technology to help facilitate travel planning. Early travel systems were largely static, with many actions really on the user's manual drive for input. Sites allow one to book flights and hotels but still require travelers to do a lot of the work themselves: comparing options, planning their itineraries. These are suitable for simple needs of travel; they are inadequate for advanced purposes of making personalized recommendations or offering dynamic advice [3][7].

3. RESEARCH PROBLEM:

Our main question is: Is it possible to design an AI tour guide capable of providing personalized and accurate travel recommendations considering realistic scenarios such as budget flights and hotels, weather conditions, tourist locations and their history.

4. DATA PROCESSING:

Input Parsing: The system uses natural language processing techniques to extract crucial information that has been put across by the user. Requests include destination, airline and hotel budget with respect to ratings and amenities desired, weather conditions, and travel dates. This will ensure the agents capture the correct intent by the user and provide relevant information [1].

API Calls: After parsing the input, each agent sends a structured API call to collect information about tourist places, hotels, weather, flights, and historical details. Data formatting is performed to unify the data coming from different sources so that it can be provided uniformly to the user for travel information [5][6].

5. AGENT DESIGN AND IMPLEMENTATION:

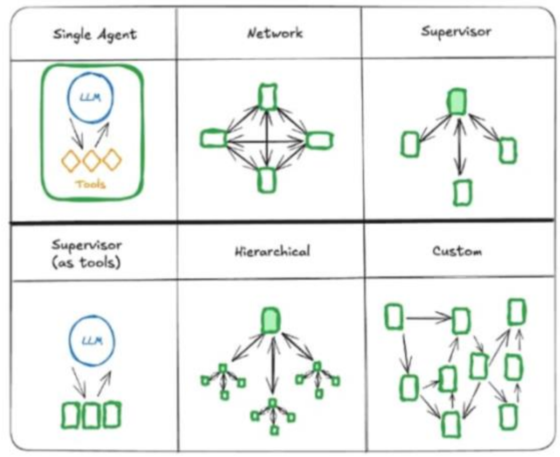

Fig 1: Types of Agents [Source: Langchain]

Flight Agent: This agent checks for customized airline travel details. Here, the agent makes use of the APIs provided by the flight aggregators and further filters the results to ensure that the flights fall within a preferred time of travel and budget.

Hotel Agent: The Hotel Agent was supposed to suggest hotels around tourist locations according to user preferences. It interfaces with hotel booking API through parsing against the budget constraints provided by the user and uses LLM-based query optimization for ranking affordability against business listing ratings. Certain challenges in parsing and filtering out the cost were addressed via Python scripting inside the LangChain framework [1].

Location-History Agent: History agent provides information about each of the tourist places in the destination. In this module, the agent receives historical facts through the processing of queries over Wikipedia. Results are then presented in a format and style easily understandable. We used the natural language summarization capability of LLMs and relevance checking against the user's query [1].

Tourist Locations Agent: This agent updates about the famous places a tourist may want to visit at the destination. Since in it, the integration of tourism data

APIs and user review analytics are in place, it makes recommendations based on personal preferences. Integration manages queries and summarizes data effectively for better results for the user [3][7].

Weather Agent: We integrated the Weather Agent, fetching real-time weather updates of any given destination. It makes use of open weather API and then processes through user input queries regarding location and time of interest to format a response using LangChain. Further code has been written to parse and interpret API data to provide an accurate forecast based on the requirements of the users' needs [2].

Tools in Agents:

These tools will make those agents more functional and effective in multi-agentic LLMs. This enables them to interface with systems outside the agent world-for example, APIs that fetch results in real time, compute elaborate calculations, and harness functionality from its functions [1]. In that way, each agent will have more ability to perform its specific tasks, be it fetching weather updates, comparing flight prices, or summarizing historical information [5]. This is enabled through tools whereby agents can extend beyond the standard language processing, resulting in much adaptable and useful ways against the diverse dynamic nature of user queries [7].

6. WORKFLOW IN MULTI-AGENTIC LLM SETUP:

- ***Setup Libraries:*** Import all the necessary libraries needed to connect language models and process data such as `LangChain`, `OpenAI`, `requests`, 'regex'[1].
- ***API Key Configuration***: Store the API key securely and set up the API keys for respective external services used, like OpenAI, serpAPI, Amadeus API, open weather API, google search API, google maps API [4].
- ***API Request Function***: Assign functions for the API calls to be made on the outside and format inputs internally in pulling data from JSON files [2].
- ***Tool Initialization***: Declare what tools each of these agents is going to use to fetch data from their respective API's.
- ***Agent Setup***: Set up what each of the agents does and what its role is, along with the tools and functionalities it would require carrying out this role, using LLM models [1].
- ***Inter-Agent Communication:*** Establish protocols of information exchange that will enable agents to effectively cooperate [2].
- ***Web Interface***: Create an intuitive web interface for specifying input query using streamlit [7].
- ***User Input Processing***: It is at this point that the Conversational Agent will process in detail what the user says or types in and extract information usable to perform a specific task [6].
- ***Execution:*** This is where the agents perform the tasks and share data, compile the personalized travel plan that is displayed to the user [1][5].

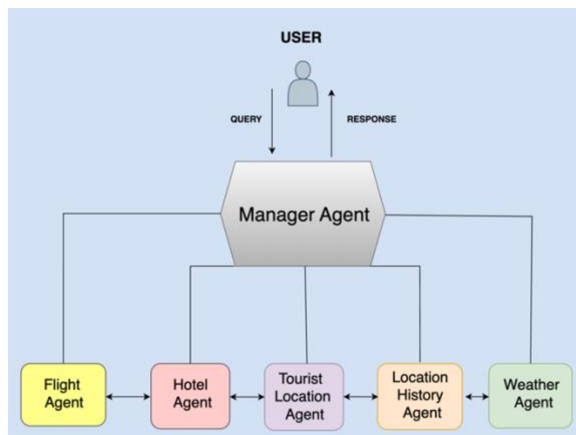

Fig 2: Multi-Agentic LLM Workflow

7. LIMITATIONS AND CHALLENGES:

• ***Latency Issues:*** One was latency, whereby some of the agents-for examples, the Flight Agent-operating responses took too much time. Even as the use of the mechanisms enhancing speed was very well utilized in the system, another issue adding to the reasons of slowness has something to do with the quantity of data they have to process.

• ***Expensive APIs***: Most of the application of the LLM-based APIs had to be translated into real financial costs, commanding many tokens per query. For agents like Hotel and Flight Agents, needing the analysis of much information, the rates are growing high, hence pressing us to seek ways whereby we could reduce the usage of tokens without changing the performance.

• ***Informational Integration and Co-ordination:***. Each agent had to share the information in an efficient manner and process it further. The miscommunication amongst each other would delay the output for the user input.

8. DESCRIPTION OF THE OUTCOMES:

Flight Agent Outcome:

This is an integrated trip-planning project that provides results personalized to include destination location and dates of travel, such as dates of departure and return. This serves a wide array of preferences efficiently by providing recommendations within a given budget range. One can get flight options within his or her budgetary allowance and further narrow down the results with favorite airline, say Emirates, for more customized results [5][6].

The system shall be able to support both domestic and international trips, including round-trip options, so it falls within the user's budget. The system then takes into consideration how much time they can spend so that a sensible itinerary may be optimally suggested and fitted to the available time of the user [6]. This makes the system, with the above qualities, an AI-friendly travel agent that can carry out trip planning and modification to specifications for budget-conscious or fixed preference travelers with airline and trip duration preferences [4][5].

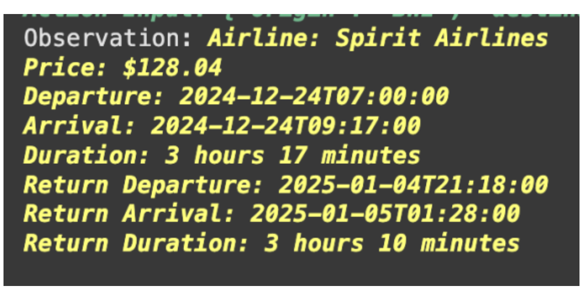

Fig 3: Flight Agent sample output

Hotel Agent Outcome:

The Hotel Agent makes personalized suggestions regarding the hotel based on predefined criteria like price, facilities available, ratings, or within budget [3]. It is considerate of the user's budget constraint and filters the option of hotels in order that the booking is matched according to financial expectations [2]. Also, the agent considers desired amenities such as free Wi-Fi, breakfast, or fitness center with specific hotel ratings for personalized suitable accommodations. In that way, suggestions will be made to keep as close to the budget as possible but at an acceptable level of comfort and convenience [6].

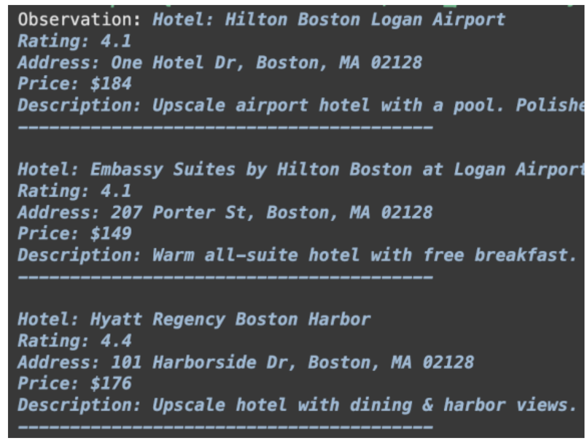

Fig 4: Hotel Agent sample output

Tourist Locations Agent Outcome:

The Tourist Locations Agent suggests links to interesting places to visit in the destination city [1]. The agent shares the ratings and popularity of various sites to recommend the best places to visit [8]. Because it is based on the feedback and reviews provided by users, the agent makes sure that the attractions recommended provide a worthwhile experience for the user who is interested in historical landmarks, natural wonders, or hotspots representing culture. In this way, it helps the users realize the worth of a trip by pointing out highly rated tourist spots related to their interests [7].

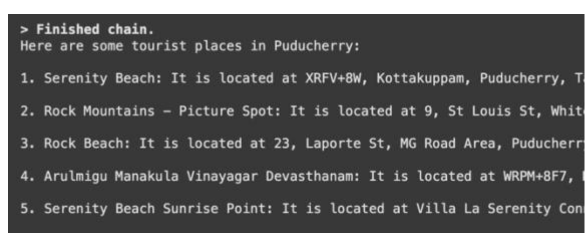

Fig 5: Tourist Location Agent sample output

History Agent Outcome:

The History Agent then summarizes information from each tourist location recommended by the Tourist Locations Agent [1]. It gives concise, interesting summaries of each attraction, ranging from basic historical facts to cultural significance and tidbits. This is so that the traveler will better understand the importance and history behind what they may see [4]. Where the agent makes the difference is in making this information easily digestible, adding meaning and enriching every visit [3].

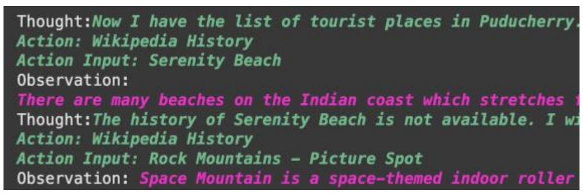

Fig 6: Tourist Location History Agent sample output

Weather Agent Outcome:

This module of Weather Agent will furnish exhaustive details regarding the specific destination's weather forecast chosen, based on the travel dates selected by the user [6]. It displays real-time weather conditions with temperatures, any kind of precipitation, and general weather for setting up travelers accordingly for their journey [1]. In this case, by providing users with a forecast of the departure date and a forecast of the return date, the agent will have equipped the user with the necessary knowledge to contemplate weather-related contingencies that would make the whole travel experience smooth and less burdensome in terms of packing and scheduling activities [5].

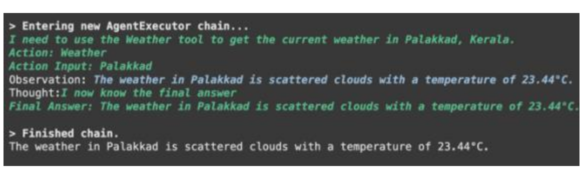

Fig 7: Weather Agent sample output

9. CONCLUSION:

The present paper provides the basic architecture for LLM multi-agent using LangChain, with the design offering a smart tour guide system in such a manner that development may give rise to personalized and effective travel planning executed by autonomous agents: weather, hotel, flight, history, and tourist recommendations.

Additionally, it has personalized travel plans quite effectively. The historical contents are good to look at, and the recommendations have updates in real time, hence increasing user satisfaction. However, the system depends on APIs provided externally. Since this would have a great effect on data quality and availability, the system is yet to be upgraded for complex handling of user preferences.

We feel that this work takes an important step toward AI-powered travel planning, significantly easing the user experience. Our future work will be directed at improving preference modeling and optimization of data sources, besides experimenting with adaptive methods for better performance and reliability - a seedbed for even more intelligent and creative travel assistants.

10. REFERENCES:

[1] Vijaykumar Kartha. 2023. "Multiple AI Agents: Creating Multi-Agent Workflows Using LangGraph and LangChain." Medium.

[2] LangChain. 2023. "LangGraph: Multi-Agent Workflows." LangChain Blog.

[3] ResearchGate. 2023. "Recommender Systems Applied to the Tourism Industry: A Literature Review."

[4] OpenAI. 2023. "GPT-4 Technical Report." arXiv preprint arXiv:2303.08774 (2023).

[5] Jianxun Lian, et al. 2024. "RecAI: Leveraging Large Language Models for Next-Generation Recommender Systems." Companion Proceedings of the ACM on Web Conference 2024.

[6] Xu Huang, et al. 2023. "Recommender AI Agent: Integrating Large Language Models for Interactive Recommendations." arXiv preprint arXiv:2308.16505 (2023).

[7] Yan Zhang, et al. 2023. "Personalized Travel Recommendation System Based on User Preferences and Social Tags." Journal of Travel Research, 62(1), 45-58.