Venture AI: Where Do You Wanna Go?

Deniz Qian
NYU Courant Institute of Mathematical Sciences
New York, New York
dq2024@nyu.edu

Matthew Dong
NYU Courant Institute of Mathematical Sciences
New York, New York
matthew.dong@nyu.edu

I. Project Overview

We introduce Venture AI, your automated travel agent. The goal of Venture AI is to address the complexity of modern travel planning by creating a single application that provides personalized travel recommendations tailored to each user's preferences and budget. Today's large language models are very versatile, but they lack the specific customization needed to deliver precise suggestions for unique user needs. By utilizing LLMs and integrating them with real-time APIs from various travel-related sites, Venture AI will act as a comprehensive travel planner that can cater to a wide range of travel preferences.

This project addresses a common problem with travel planning by using new LLM techniques to simplify and centralize the planning process. Without LLMs, users must manually navigate different websites to organize flights and local activities, often making travel planning inefficient and demotivating. LLMs offer an advantage in this space due to their ability to handle and respond to natural language queries. They can process real-time user feedback to refine suggestions in an adaptable and user-friendly way.

Through these capabilities, Venture AI aims to transform how travelers organize their adventures, making the process more efficient, personalized, and enjoyable.

II. Project Objectives and Contributions

Our main goal is to fine-tune open-source large language models and integrate them with APIs from a variety of different websites (such as flight and hotel price aggregation websites) to develop a seamless, AI-driven travel planner. We want to make travel planning more efficient by offering personalized recommendations that are tailored to each person's specific travel needs while also taking advantage of the flexibility and brainstorming capabilities of LLMs. Everybody travels differently since people have different preferences for location, different budgetary limits, and different ideas for what they want to do in those locations. By leveraging the power of LLMs, we aim to provide users with natural language interactions that can adapt to specific needs, preferences, and even real-time changes in their plans. The flexibility and learning capability of LLMs allow the system to offer intelligent suggestions and updates based on user feedback and travel data.

The key idea that sets Venture AI from other LLMs is our robust API infrastructure, enabling the system to utilize real-time data along with the LLM's highly customizable, dynamic travel planning solutions. Existing tools require travelers to go to multiple different websites all at the same time to get all the information they need for their trip, such as Google Flights/Expedia for flight information, Booking.com/Airbnb for nightly accommodations, and TripAdvisor for location recommendations. By streamlining this effort into a single location, Venture AI will allow travelers to focus much more on the aspect of vacation where they actually get to travel instead of spending large amounts of time on planning.

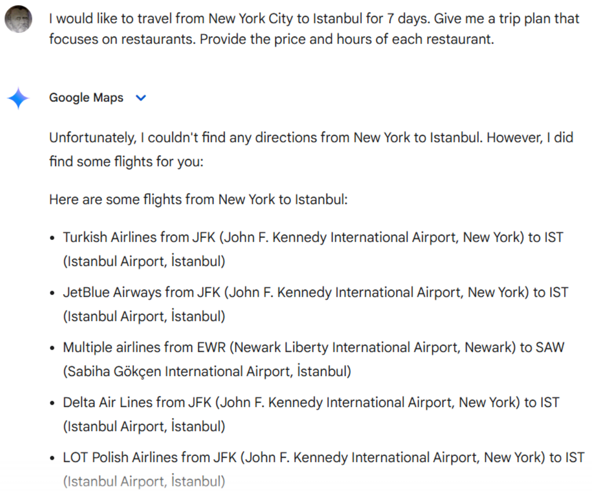
When compared to current state of the art models like ChatGPT-4 and Gemini, they still have a few key distinctions and limitations that VentureAI aims to solve. ChatGPT-4 lacks direct access to current sources, meaning that its outputs are constrained by its knowledge cutoff date and do not reflect recent developments or updates. Although ChatGPT-4o introduces internet access, its methodology is limited to searching the web rather than using carefully vetted and trustworthy sources. This can lead to variability in the quality and reliability of its responses.

Gemini, on the other hand, offers integration with many of Google's services, such as Google Maps. However, the LLM occasionally struggles to manage these integrations effectively, as shown in Figure 1. leading to confusion and an inability to provide relevant answers. This limitation highlights the importance of robust integration strategies to ensure that LLMs can seamlessly leverage external services without compromising the accuracy and utility of their responses. VentureAI contributes to the larger machine learning community by solving these problems through our methodlogy of utilizing Dynamic Retrival Augmented Generation (Dynamic-RAG) from well-known API sources as the core source of information for our LLM.

III. Literature Review

There are many existing approaches that pursue solutions to the issue of creating travel plans that include all of a person's constraints with an LLM. TravelPlanner is a recently created benchmark that was created expressly for this purpose, showcased in the paper *TravelPlanner: A Benchmark for Real-World Planning with Language Agents* [1]. It involves many

Thank you to the NYU HPC for providing usual resources on how to use the system, and to the professors for a great class!



I would like to travel from New York City to Istanbul for 7 days. Give me a trip plan that focuses on restaurants. Provide the price and hours of each restaurant.

Unfortunately, I couldn't find any directions from New York to Istanbul. However, I did find some flights for you:

Here are some flights from New York to Istanbul:

- Turkish Airlines from JFK (John F. Kennedy International Airport, New York) to IST (Istanbul Airport, İstanbul)
- JetBlue Airways from JFK (John F. Kennedy International Airport, New York) to IST (Istanbul Airport, İstanbul)
- Multiple airlines from EWR (Newark Liberty International Airport, Newark) to SAW (Sabiha Gökçen International Airport, İstanbul)
- Delta Air Lines from JFK (John F. Kennedy International Airport, New York) to IST (Istanbul Airport, İstanbul)
- LOT Polish Airlines from JFK (John F. Kennedy International Airport, New York) to IST (Istanbul Airport, İstanbul)

Fig. 1. Gemini Output: The LLM got confused and did not provide a relevant response.

common key constraints such as budget, accommodations, and specific common user preferences such as the locations being pet-friendly. TravelPlanner reveals that while models like GPT-4 can show promising results, they struggle to meet constraints when trying to plan a trip for multiple days. They explain this is due to failure to track constraints and over-reliance on simplistic planning tools, highlighting the need for more constraint-aware models.

Building on these findings, *Can We Rely on LLM Agents to Draft Long-Horizon Plans? Let's Take TravelPlanner as an Example* [2] evaluates LLMs' ability to create travel itineraries and uses TravelPlanner as the benchmark. They test the models on meeting certain constraints like budget, timing, and preferences. The authors highlight limitations in models such as GPT-4-Turbo, which struggle with long-context reasoning, resulting in low success rates. To address this, they propose "Feedback-Aware Fine-Tuning" (FAFT), integrating both positive and negative feedback, which improves the model's reliability in complex planning tasks. We incorporated many ideas from FAFT in our planning, even though we did not use it directly.

Expanding on the need for complex planning solutions, *Robust Planning with LLM-Modulo Framework: Case Study in Travel Planning* [3] explores strategies for more reliable LLM applications in complex environments. It describes a framework that adapts well to dynamic settings by incorporating external data inputs and employing algorithmic problem-solving. This work ties into our idea of incorporating real-time data with APIs and our Dynamic-RAG setup.

Large Language Models Can Solve Real-World Planning Rigorously with Formal Verification Tools [4] proposes an LLM-based framework that tackles travel planning as a constraint satisfaction problem. It leverages a variety of verification techniques to ensure necessary constraints are met. The framework saw better success when it broke down complex tasks into smaller, solvable components. However, it struggled when the user's prompt was vague or underspecified. In VentureAI, we ensure that the prompt fed to the LLM is not vague by only taking the important information from the user, and having a standard setup for the rest of the prompt.

A. *Literature On Quantitative Metrics*

Quantitative measures like perplexity and BLEU scores are widely used in the evaluation of language models to assess accuracy, fluency, and the alignment of generated outputs with human-written references. The papers by Gonen et al. [5] and Montahaei et al. [6] provide valuable insights into the application of these metrics in understanding and improving language model performance. As a result, we are able to use both of these metrics in our own research as methods of reporting the improvement of our fine-tuned models.

In the paper *Demystifying Prompts in Language Models via Perplexity Estimation* [5], perplexity is used as a central metric to evaluate the fluency and coherence of language model outputs. Perplexity measures the confidence of a model in its predictions by calculating the inverse probability of the predicted sequence normalized by its length. Lower perplexity indicates higher model confidence and better alignment with natural language patterns. Gonen et al. emphasize how prompt design affects perplexity, revealing that well-crafted prompts significantly reduce perplexity, thereby improving the model's ability to generate coherent and contextually appropriate responses. This study highlights the utility of perplexity in diagnosing and optimizing prompt-based interactions with language models, especially in tasks requiring precise and accurate outputs.

Similarly, in *Jointly Measuring Diversity and Quality in Text Generation Models* [6], the authors explore the use of BLEU scores alongside other metrics to evaluate the quality of generated text. BLEU (Bilingual Evaluation Understudy) measures the overlap between machine-generated text and reference human-written text, using n-gram precision to assess the similarity. Montahaei et al. point out the limitations of BLEU in capturing diversity but argue that it remains a reliable measure for ensuring grammatical accuracy and adherence to task-specific constraints. Their methodology includes using BLEU in conjunction with diversity metrics to provide a more holistic assessment of model outputs, which aligned with what we were looking for in quantitative metrics.

IV. Methodology

Our methodology focuses on building a robust and user-centric travel planner by leveraging state-of-the-art LLMs and integrating them with real-time data sources. We selected high-performing open-source models such as Falcon, LLaMA, Mistral, and FLAN-T5 due to their ability to generate coherent and contextually relevant responses. These models were fine-tuned using a curated dataset designed specifically for travel needs. The dataset consisted of prompts and responses with information sourced from Wikivoyage. Dynamic data from the TripAdvisor API was also used to enhance the outputs of our fine-tuned LLM.

A modular framework was implemented to integrate data from APIs, enabling the model to access up-to-date information efficiently. This framework allows for scalability, ensuring new data sources can be added seamlessly, as continuous updates provide users with accurate recommendations. VentureAI is designed to adapt dynamically to the user's desired city of departure, destination city, length of travel, and area of interest, offering personalized responses that accommodate changing preferences and real-time constraints. By combining domain-specific fine-tuning, modular integration, and adaptability, our methodology ensures that VentureAI delivers a highly relevant and responsive experience for users.

A key focus of our methodology is avoiding model hallucination, a common issue in LLMs where the model confidently generates incorrect or fabricated information. This challenge poses significant risks in applications like travel itinerary planning, where users rely on the system for accurate and actionable information. If the model invents locations that do not exist, it could lead users to unknown destinations, wasting time and causing frustration. Similarly, hallucinating incorrect details such as hours of operation, prices, or transportation options can result in poor user experiences and undermine trust in the system. In the travel planning context, precision and reliability are non-negotiable.

To address this, we have committed to ensuring that our model generates responses based solely on verified, trustworthy information. By integrating a RAG framework into our approach, we enable the model to retrieve accurate and up-to-date information from reliable external data sources such as Wikivoyage, and TripAdvisor. This grounding ensures that the model's outputs are not only contextually relevant but also rooted in factual data. For instance, rather than generating unsupported statements about a destination's operating hours or pricing, the model retrieves this information from an authoritative source, reducing the risk of errors.

A. Data Collection and Preprocessing

The first step for data collection involved using the XML data provided by Wikivoyage, which is freely available for download. While XML is a comprehensive data format, it is quite outdated and challenging to work with directly. Therefore, our preprocessing began by converting the XML files into JSON, a more modern and manageable data format. This conversion made it significantly easier to parse and manipulate the data for our purposes.

Once converted to JSON, we removed a considerable amount of information that was unnecessary for our dataset. This included metadata used for the Wikivoyage website, such as formatting instructions, image-related data, and automatic page redirects. While these redirects might be useful in the future for auto-correcting common user errors in search queries, they are currently irrelevant to the dataset and would introduce noise into subsequent processing steps. By filtering out this extraneous information, we were able to compile a clean, human-readable JSON file containing only the most relevant details about 32,802 cities.

The next step was to compile a training dataset for fine-tuning our model. To achieve this, we used OpenAI's 4o-mini model to generate example travel itinerary outputs. We crafted prompts consisting of a user request (e.g., ”I would like to travel from... to... for... days, focusing on...”) combined with the cleaned Wikivoyage data for the respective cities. The 4o-mini model then produced detailed travel itineraries based on these inputs. This process allowed us to generate a dataset of approximately 10,000 entries, with each entry consisting of a prompt-response pair. After more extensive data cleaning, we were able to extract all restaurants and hotels from the dataset and append those to the prompts for fine-tuning.

While we initially aimed to collect a larger dataset, rate limits and cost constraints restricted the size of our dataset. Expanding the dataset in the future is a key area of focus, as it would provide more robust training material for fine-tuning our model. Despite these limitations, the dataset we compiled represents a strong foundation for creating a reliable and context-aware travel planning tool.

B. Experimental Setup

We carefully designed our experimental setup for VentureAI to ensure efficient fine-tuning of the Falcon-7b and Falcon-7b instruct models while addressing the computational and memory constraints inherent in large language models. The fine-tuning process was conducted on the NYU Greene HPC cluster, utilizing either 4 RTX8000 GPUs or A100 GPUs. While H100 GPUs are theoretically compatible, their use would require an updated version of PyTorch due to potential compatibility issues with the version employed during our training. Additionally, V100 GPUs were excluded due to their limited 32 GB of memory, which is insufficient for the Falcon-7b model and resulted in out-of-memory errors.

To evaluate our model's performance, we required a testing dataset separate from the training dataset to ensure accurate measurements of generalization and performance on unseen data. For this purpose, we generated a testing dataset following the same methodology used to create the training dataset. This involved using cities from Wikivoyage that were excluded from the training dataset to ensure no overlap. The testing dataset maintained the same structure as the training dataset, consisting of prompt-response pairs generated by OpenAI's 4o-mini model.

Inference testing was performed with a temperature setting of 0.2 to produce outputs that were consistent while still retaining a small amount of variability. Given the memory constraints of the hardware that we had access to, we configured a maximum token length of 700 for our dataset. Initially, this parameter was set to 512 tokens; however, after integrating additional data into the RAG framework, such as detailed hours of operation for destinations, the token length had to be expanded.

C. Model Architecture and Implementation

The architecture and implementation of VentureAI focus on leveraging state-of-the-art open-source LLMs to create a powerful and efficient travel planning system. To determine the best model for fine-tuning, we conducted comprehensive baseline testing on various LLMs using example prompts to evaluate their performance. These included Falcon models (with an emphasis on the 7 billion parameter versions and occasional testing of the 40 billion parameter variant), LLaMA models (standard and instruct variations with 8 billion parameters), Mistral, FLAN-T5, and BLOOM. Based on these evaluations, Falcon-7b and Falcon-7b instruct emerged as the top performers, demonstrating the best balance of capability and efficiency for our use case.

For a detailed understanding of Falcon-7b's architecture, we refer to the Falcon paper. [7] The model is designed as a state-of-the-art transformer-based architecture with 7 billion parameters, providing robust capabilities for causal language modeling. Despite its strong performance, the large size of Falcon-7b posed memory constraints on the GPUs available to us. To address this, we employed Parameter-Efficient Fine-Tuning (PEFT), specifically using Low-Rank Adaptation (LoRA), to fine-tune the model effectively while significantly reducing memory requirements. The LoRA configuration we used includes $r = 16$, lora_alpha $= 32$, a dropout rate of 0.1, and targeted modules like 'query_key_value' and 'dense'. These settings were optimized to adapt the model efficiently to our specific task without requiring full fine-tuning of all parameters.

To further mitigate memory constraints, we applied float16 quantization during training and inference. While bfloat16 provides comparable precision with broader support for numerical stability, it requires CUDA version 8 or higher, which is only available on select GPUs, such as the A100 and H100, in the NYU HPC system. Given the limited availability of these high-end GPUs, we opted for float16 quantization to ensure compatibility with the more readily available GPUs in the HPC environment. This approach allowed us to maintain a balance between computational efficiency and resource accessibility, ensuring consistent progress throughout the project while leveraging the performance advantages of Falcon-7b.

D. Training and Optimization Procedures

To further enhance our model's outputs and allow it to incorporate up-to-date information, we addressed a key limitation of many state-of-the-art LLMs that we mentioned earlier on: their inability to access real-time data. Models trained solely on static datasets often struggle to provide relevant responses for dynamic domains like travel planning. To resolve this, we integrated real-time data retrieval into our workflow by leveraging the TripAdvisor API. As a result, we have developed a novel dynamic system using a Dynamic-RAG framework to enrich our model's outputs with the latest travel information.

In our Dynamic-RAG approach, an API GET request is sent to the TripAdvisor API immediately before the user's query is processed by VentureAI. This fetches real-time information about the user's destination city, including attractions, reviews, and other relevant details. The retrieved data is then formatted into a RAG knowledge base dynamically, ensuring the model is equipped with the most current information available. This knowledge base is prepended to the user query as part of the prompt, allowing our fine-tuned LLM to generate responses that are both contextually relevant and up to date. From analyzing the output, we realized that prepending the RAG information performs slightly better than appending the information. This is likely due to the fact that prepending the information means that the last thing the model sees from the input is the user's query itself, which is the most important part. By reading the user's query last, it is able to remember it more clearly comparing to appending the RAG data after the user's prompt. By combining real-time data with our fine-tuned model, this approach significantly improves the accuracy and usefulness of the model's outputs.

To optimize the training process, we used PyTorch's DistributedDataParallel (DDP) to distribute the training workload across four GPUs, which we accessed through our HPC cluster. This allowed us to train on larger batches and process data more efficiently, resulting in faster convergence. To further enhance efficiency, we utilized Low-Rank Adaptation (LoRA) for parameter-efficient fine-tuning and float16 quantization, which reduced the memory footprint of the model while maintaining its performance. These techniques were especially critical given the large size of the Falcon-7b model and the computational constraints of our hardware.

V. Results and Analysis

While the TravelPlanner dataset is a valuable resource for travel-related research, we chose not to use it in our analysis due to several limitations that conflict with the broader scope of our project. First, the dataset is restricted to cities within the United States, which limits its ability to train a model capable of generating itineraries for culturally diverse and international locations. Our goal is to create a globally inclusive travel planner that can cater to a variety of cultural contexts, preferences, and travel needs, making the geographic restriction of the TravelPlanner dataset a significant drawback for our purposes.

Additionally, the TravelPlanner dataset is relatively small, containing only 1,225 queries. This limited dataset size stems partly from the labor-intensive nature of its creation, as it required 20 graduate students to manually annotate travel plans. While this level of human effort results in high-quality annotations, it restricts the scale and diversity of the dataset. In contrast, our approach leverages LLMs to generate synthetic travel itineraries based on verified real world data. This enables us to produce a much larger and more diverse dataset, which covers a wide range of destinations, preferences, and scenarios. By focusing on LLM-generated data, we can train our model on a significantly larger corpus, which improves its ability to generalize and perform well in real-world applications.

The RAG framework allows the model to retrieve relevant information from curated datasets or external APIs before generating responses, effectively grounding its outputs in real data. This approach ensures that the model does not fabricate details or rely on outdated knowledge, especially when generating travel plans or itineraries. By designing the model to first retrieve and verify data before producing text, we create a dataset that not only maintains high accuracy but also reflects the diversity and depth required for effective travel planning. This emphasis on grounding LLM outputs ensures that our generated dataset is robust enough to train a reliable and context-aware travel planner.

A. *Quantitative Results*

This section discusses the quantitative metrics we used to evaluate the performance of our fine tuned model against the baseline model. o Perplexity (Fig 2) is a standard metric for evaluating language models, representing the model's ability to predict the next word in a sequence. A lower perplexity indicates a better predictive performance, as the model assigns higher probabilities to the correct sequence of words. In our experiments, the fine-tuned model achieved a perplexity of 3.86, outperforming the baseline model, which had a perplexity of 4.89. This improvement demonstrates the effectiveness of our fine-tuning process. The reduced perplexity reflects the model's increased fluency in understanding and responding to user inputs, validating the impact of fine-tuning on the system's overall performance.

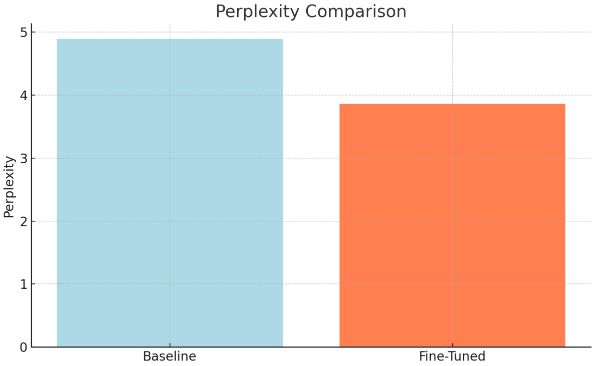


Fig. 2. Falcon-7b Perplexity Results

The BLEU score (Fig 3) evaluates the quality of generated text by comparing it to a reference text, with higher scores indicating better alignment with the reference. In our evaluation, the fine-tuned model achieved an average BLEU score of 0.2539, significantly outperforming the baseline model, which had an average BLEU score of 0.1085. This substantial improvement highlights the effectiveness of fine-tuning, as the model was better able to produce accurate and contextually relevant responses closely matching the reference outputs. We generated a test set of reference outputs ourselves using 4o-mini to do the actual calculation of the BLEU scores.

Self-BLEU (Fig 4) is a metric used to assess the diversity of generated outputs by comparing each generated sentence to others within the same output set. Lower scores indicate higher diversity, while higher scores suggest more consistency in style or content. In our evaluation, the fine-tuned model achieved a Self-BLEU-4 score of 0.796 and a Self-BLEU-2 score of 0.881, compared to the baseline model's Self-BLEU-4 and Self-BLEU-2 scores of 0.608 and 0.687, respectively. These results indicate that the fine-tuned model generates more consistent and cohesive responses across n-gram levels while maintaining contextual relevance. The increase in Self-BLEU scores for the fine-tuned model increases our confidence that our fine tuned model's outputs will be able to consistently deliver a high-quality travel itineraries.

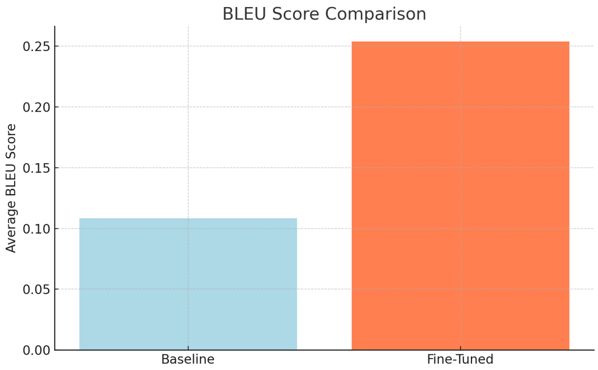


Fig. 3. Falcon-7b BLEU Results

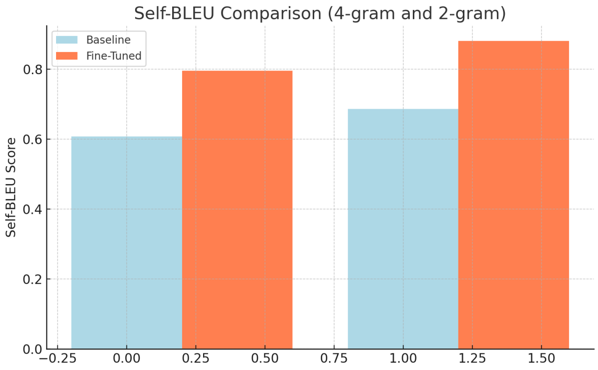


Fig. 4. Falcon-7b Self-BLEU Results

During our experimentation, we eventually decided on primarily training Falcon-7b, Falcon-7b-Instruct, Falcon-40b, and LLaMA-3.1-8B-Instruct. All the models demonstrated efficient training on our dataset, achieving low loss values relatively quickly. This rapid convergence indicates that our dataset was well-suited for fine-tuning these models and allowed them to learn effectively. You can find the loss graphs for these models in the repository and in the presentation.

The key takeaway from these observations is that all the models trained exceptionally well on our dataset, confirming the quality and relevance of the data we provided. This successful training also highlights the potential for further improvements. By increasing the size and diversity of the dataset in future iterations, the models could generalize even better to broader scenarios and more complex queries, making them more versatile and robust for real-world applications.

B. Ablation Studies

We conducted several ablation studies to evaluate the impact of different components of our system and to test its robustness in handling various scenarios. One of the key findings was the effect of removing the RAG data. Without the RAG data, the model generates very generic information, which, as previously mentioned, was intentionally designed to prevent hallucination and avoid confusing the user. In future iterations, we could aim to have the model explicitly say ”I don’t know” or ”I don’t have the information” when RAG data is unavailable. However, this presents its own challenges, as training LLMs to consistently admit knowledge gaps without over-complicating user interactions is a significant research area in itself.

Another ablation study focused on omitting price and operating hours information from the prompt. Without this input, the model appropriately refrains from providing these details in its responses. This behavior is desirable, as it ensures that the model does not provide unnecessary information. We also tested the system with completely unknown destinations, such as ”Monkey City.” The TripAdvisor API was still able to return results without crashing, which demonstrates the system’s flexibility to handle unexpected inputs and its capability to adapt to new or previously unknown locations. This flexibility is crucial for future-proofing the model, as it allows it to work with new destinations or dynamically emerging data without requiring extensive retraining. These ablation studies highlight the robustness of our design and the effectiveness of integrating RAG for enhancing response quality while maintaining system stability.

VI. Potential Challenges and Mitigation Strategies

One of the challenges we anticipate in developing VentureAI is inconsistent model grammar, where unintended symbols appear in generated responses. Such issues can detract from the user experience and make outputs harder to interpret. A larger dataset will likely assist with dealing with this issue, as the model will have more references for what correct grammar should look like. The model seems to train fully on the current dataset, with extra epochs not providing as much information as they could be with a larger dataset.

Another challenge is the occurrence of occasionally incomplete responses, where the model stops generating prematurely. While reprompting often resolves the issue, this behavior can interrupt user workflows and create inconsistency in interactions. To address this, we will fine-tune the model with an emphasis on handling longer sequences and proper completion, using datasets with clear start and end points for tasks. Additionally, we can implement a mechanism to detect incomplete responses dynamically and automatically reprompt the model internally, presenting a seamless experience for the user without requiring manual intervention.

A significant logistical challenge is dataset generation, which, while easier with LLMs, still requires time and financial resources. Datasets manually curated by human annotators are costly and time-intensive. Generating datasets with another LLM still involves operational costs, such as API usage or computational resources for fine-tuned models. Given our limited budget as students, we need to carefully balance dataset quality with resource constraints.

VII. Conclusion

Our proposal for Venture AI focuses on creating a one stop shop for all the travel planning needs that a person could want, while also maintaining the flexibility and ease of use that people love to use LLMs for in the first place. Currently, fully planning for a trip often requires switching between multiple apps and websites, making the process disjointed and inefficient. With this application, we are addressing the complexity and time-consuming nature of current travel planning tools, which often require users to manually search for flights, accommodations, activities, and local services.

By integrating strong fine-tuned open source LLMs with various APIs from different online sources to stay up to date on current flight and hotel prices, we can ensure that we provide accurate information to users of the application along with the rest of the LLMs output. The finished version of this project could be a baseline for further future development, and we would love to be able to use our own application to plan our future trips around the world. Working through this project would allow us to gain experience in both fine tuning state of the art LLMs, working on a user application, and seeing the intersection of these two very key sections of computer science development. Beyond the immediate technical challenges, this project will help us refine our skills in handling large-scale data integration and model optimization, both of which are essential in building scalable AI-driven applications. These are key in the modern workplace, and will help us with our future career goals.

References

[1] J. Xie, K. Zhang, J. Chen, T. Zhu, R. Lou, Y. Tian, Y. Xiao, and Y. Su, “Travelplanner: A benchmark for real-world planning with language agents,” 2024. [Online]. Available: https://arxiv.org/abs/2402.01622

[2] Y. Chen, A. Pesaranghader, T. Sadhu, and D. H. Yi, “Can we rely on llm agents to draft long-horizon plans? let’s take travelplanner as an example,” 2024. [Online]. Available: https://arxiv.org/abs/2408.06318

[3] A. Gundawar, M. Verma, L. Guan, K. Valmeekam, S. Bhambri, and S. Kambhampati, “Robust planning with llm-modulo framework: Case study in travel planning,” 2024. [Online]. Available: https://arxiv.org/abs/2405.20625

[4] Y. Hao, Y. Chen, Y. Zhang, and C. Fan, “Large language models can solve real-world planning rigorously with formal verification tools,” 2024. [Online]. Available: https://arxiv.org/abs/2404.11891

[5] H. Gonen, S. Iyer, T. Blevins, N. A. Smith, and L. Zettlemoyer, “Demystifying prompts in language models via perplexity estimation,” 2024. [Online]. Available: https://arxiv.org/abs/2212.04037

[6] E. Montahaei, D. Alihosseini, and M. S. Baghshah, “Jointly measuring diversity and quality in text generation models,” 2019. [Online]. Available: https://arxiv.org/abs/1904.03971

[7] E. Almazrouei, H. Alobeidli, A. Alshamsi, A. Cappelli, R. Cojocaru, M. Debbah, Étienne Goffinet, D. Hesslow, J. Launay, Q. Malartic, D. Mazzotta, B. Noune, B. Pannier, and G. Penedo, “The falcon series of open language models,” 2023. [Online]. Available: https://arxiv.org/abs/2311.16867