GenPerfectTrip

Final Report

Yiwei He, Junhong Chen

December 12, 2023
word count: 1860

1 Introduction

Trip preparation can sometimes be extremely tedious and time-consuming, especially when planning to travel to a place that you had no previous experience with. GenPerfectTrip is a project that aims to generate, and construct detailed, personalized trip plans based on users' input of their trip preferences. The users can also interact with the system to build onto the preliminary plan to further improve the generated plan.
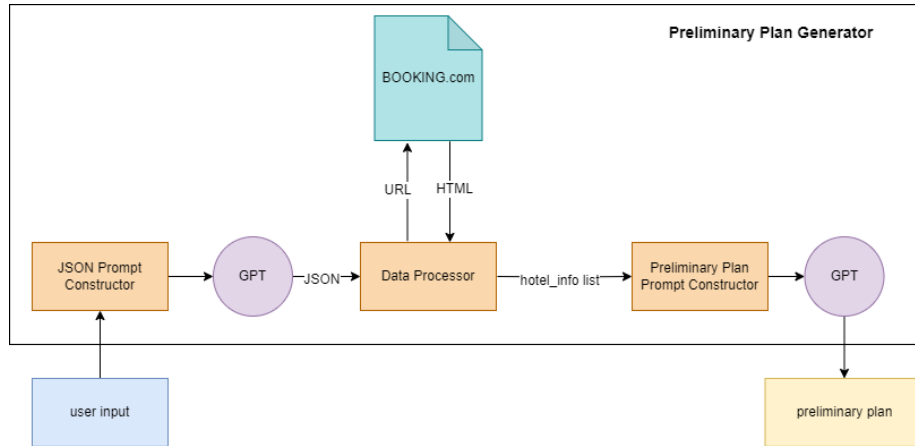
2 Illustration

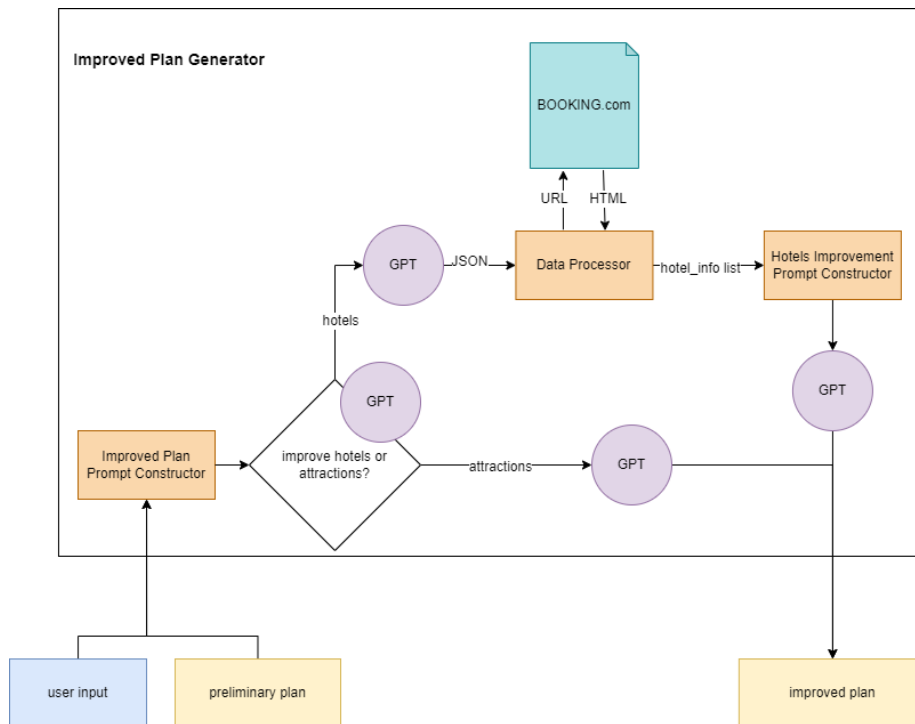Figure 1: Architecture for the Preliminary Plan Generator

Figure 2: Architecture for the Improved Plan Generator

3 Background and Related Work

Prompt engineering is a technique of crafting inputs to elicit desired outputs from generative models. It helps tune the models to give more accurate outputs and improve the overall output quality for specific complex tasks, which therefore is widely used in the real of applications of natural language processing. Prompt engineering is not just redesigning and reconstructing prompts, it also contains a wide range of skills and patterns to be able to guide the models to give better outputs. White (2023) and his colleagues came up with multiple prompt engineering techniques presented in pattern form that can be used to generate better prompts to successfully improve the outputs of LLM conversations, allowing the model to able to focus on different keywords in different circumstances and hence produces desired outputs. There are different approaches regards to different patterns. The Persona Pattern conveys the idea that it's easier for the models to produce outputs from a certain point of view when telling them to act as persona X. GenPerfectTrip used several prompt engineering techniques to instruct the GPT model to give the optimal outputs during the trip plan generation process, which include the Persona Pattern mentioned above.

4 Data and Data Processing

4.1 Data Collection

1. Collection of 20 sets of input trip preferences, each individual input may contain information about destinations, trip duration, and the budget for the trip. The testing data set ranges from very descriptive trip preferences to very unclear and general preferences. The following are two of the examples:

   (a) Descriptive trip preference: I have a break for 3 days and wanting to travel to Vancouver. I don't mind spending money for a comfier and more high-end hotel during my trip. Could you please help me come up with a plan?

   (b) Ambiguous trip preference: I want to go skiing for a week in Canada, help me find a destination and hotels.

2. Human collection of hotel and attractions information based on the trip preferences for comparison with the assistance of GPT Expedia plug-in.

4.2 Data Processing

1. Pre-web scraping processing involves with retrieving information from user input to be used for web scraping.

2. Post-web scraping processing involves with taking the hotel information from web scraping, selecting several top-rated hotels, and storing them in a list to be used for plan generation. The hotel information contains hotel name, price per night, location of the hotel.

5 Architecture and Software

The software comprises primarily two components: generating the preliminary plan, and improving the preliminary plan. The architectures of these two components are outlined in the illustration section.

5.1 Architecture for Preliminary Plan Generator

The JSON prompt constructor's purpose is to guide the GPT in converting user input into JSON format which contains "destination", "price_range", " number_of_nithgts" keys and correlative values. It constructs the system prompt instructing GPT to extract information such as destination, trip duration, and price range, from the user input (if available) and return a JSON. There are some specific rules for constructing the JSON in this system prompt. This module sends a request to GPT where the user prompt is the user input and the system prompt refers to Appendix (13.1).

The data processor's role is to utilize the information provided in JSON for web scraping. It parses the JSON, extracts relevant information, and uses it to form a URL that can access an available web page on BOOKING.com for search results for hotels. Upon sending a request to this URL, the data processor retrieves the HTML content, which contains the details about the listed hotels. It then parses HTML content to extract information such as name, location, price, rating, and distance to the location. Ultimately, it outputs a list of hotels information to the Preliminary Plan Prompt Constructor.

The preliminary plan prompt constructor creates a system prompt instructing the model in a two-step plan generation process. Firstly, it guides the model to select hotels from the provided list, shaping the accommodation aspect of the plan. Secondly, it directs the model to generate a detailed itinerary for each day of the trip. This module sends a request to GPT where the user prompt is the user input and the system prompt refers to Appendix (13.2).

5.2 Architecture for Improved Plan Generator

The improved plan generator shares a similar structure with the preliminary plan generator but diverges in terms of logic and content flow.

The improved plan prompt constructor is responsible for generating a system prompt that directs the GPT model to produce either a JSON or an improved plan based on the user's preference to modify hotels or activities. Please refer to Appendix (13.3) for the example.

Unitizing the Persona pattern mentioned above, the prompt tells the model to act as a trip planner so that it is facilitated to improve the plan from a trip planner's perspective. The prompt then directs the model to decide whether to generate a JSON or improve the plan by replacing the current activities with the activities the user is interested in. The preliminary plan is seamlessly integrated into the system prompt, allowing the model to make any improvements

to the activities before returning the improved plan to the user.

If the GPT determines the user's intention to improve the hotels, it considers the user's new requirements and generates a JSON using the same rule as the JSON prompt constructor. In this scenario, the data processor executes the same process as in the preliminary plan generator, producing a list of hotels information. This list is then delivered to the hotel improvement prompt constructor, which constructs a system prompt incorporating the provided hotel list, the preliminary plan, and the instructions for the model to change the hotels only. Please refer to Appendix (13.4) for the example.

6   Baseline Model and Comparison

1. To evaluate the performance of the system, a set of accuracy criteria was created, and human evaluations were utilized to check for each individual criterion. These criteria are used to evaluate both the preliminary and improved plan.

    (a) Destination accuracy: To verify the system did not randomly choose a destination that is not related to the user input.

    (b) Duration of Stay accuracy: To verify the system complied with the user input.

    (c) Price Range accuracy: To verify the system did not suggest a hotel that is over budge or under budget. This one is a bit tricky, as sometimes the user preference can be objective, such as what does fancy mean in terms of price per night. In this case, it is needed to compare the selected hotel's price with other hotels in that destination and verify that it fits the user preferences.

2. For the improved plan, there is a strict requirement of only changing things the user explicitly asked to modify. This is to make sure that the improved plan is not just a newly generated plan, but rather based on previous plan and an actual improvement over the preliminary plan. This is done through comparing the detail in the two plans such as hotel information and various activities.

3. Each generated preliminary plan is tested with attempts to change either activities and hotel information. The following is the guild line used:

    (a) For modifying hotel information, user input should only contain the new preference for hotel. For example, "I want to stay in cheaper hotels"

    (b) For modifying activities, user input should only contain the new preference for activities. For example, "Instead of shopping, I want to explore the city more"

4. To further evaluate the system' generated plan, it is compared against the GPT Expedia Plug-in with the same user preferences.

7 Quantitative Result 100

Results				
Plans	Destination accuracy	Price accuracy	Duration accuracy	Addition requirement
Preliminary Plan	100%	100%	100%	N/A
Improved Plan	100%	92%	100%	90%

8 Qualitative Result

Most of the inaccuracy in improved plan came from inputs that ask the system to find cheaper hotels, if the preliminary plan already has cheap hotels, it might fail to find cheaper hotel and generate a plan with more expansive hotels. In the case of the additional requirement, the system struggles when the user wants to modify the duration of stay where the generated output would modify both activity and hotel.

8.1 Positive Example

Figure 3 shows an example output for a fancy 3 day trip to Vancouver. This generated plan was able to convert descriptive words such as fancy into a reasonable numerical value that represents the target price of the hotels. It also selected the correct destination and generated an itinerary over 3 days.

Figure 3: Sample Output for the Preliminary Plan

Figure 4 shows an example that was considered as a positive output for the improved plan. The user input for this particular sample was that the user wants to stay in cheaper hotels. The system was able to keep the previous itinerary and select cheaper hotels for the user to stay in.

Sure, I can recommend some more affordable options for your stay. Here's your updated itinerary:

Day 1:
- Check-in at the "Sonder at Revival". It's a well-rated hotel located in West End, Vancouver and is priced at CAD 556. The hotel has a good rating of 8.9 and is just 1.4 km from the center.
- After settling in, start your day with a visit to the Vancouver Art Gallery which is a short drive away from the hotel.
- Have lunch at one of the many restaurants in the downtown area.
- In the afternoon, take a stroll in the nearby Stanley Park, enjoy the nature and the views of the city.
- Have dinner at a restaurant in the park.
- Return to the hotel and rest for the day.

Day 2:
- Check-out from the "Sonder at Revival" and check-in at the "Sheraton Vancouver Wall Centre". It's a comfortable hotel located in Downtown Vancouver and is priced at CAD 596. The hotel has a good rating of 8.4 and is just 500 m from the center.
- Visit the Science World at TELUS World of Science. It's a great place to explore and learn about various scientific phenomena.
- Have lunch at a nearby restaurant.
- In the afternoon, visit the Vancouver Aquarium located in Stanley Park.
- In the evening, enjoy a dinner cruise around the Vancouver Harbor.
- Return to the hotel and rest for the day.

Day 3:
- Check-out from the "Sheraton Vancouver Wall Centre" and check-in at the "Sonder The Sunset". It's a well-priced hotel located in West End, Vancouver and is priced at CAD 570. The hotel has a good rating of 8.4 and is just 0.9 km from the center.
- Visit the Granville Island Public Market in the morning. It's a bustling and vibrant area filled with a variety of food vendors, artisans, and performers.
- Have lunch at one of the many eateries on the island.
- In the afternoon, take a ferry to North Vancouver and visit the Capilano Suspension Bridge Park.
- Return to downtown Vancouver and have dinner at a local restaurant.
- Return to the hotel and rest for the night.

Day 4:
- Check-out from the hotel and return home.

This plan includes a variety of activities and stays at more affordable hotels, ensuring you have a comfortable and enjoyable trip to Vancouver.

Figure 4: Sample of Positive Output for the Improved plan

8.2 Negative Example

8.2.1 Failure in Fulfilling Addition Requirement

Figures 5 and 6 show an example that was considered as a negative sample for the improved plan. In this example, it was asked to change the hotel information, however in the improved plan, both hotels and activities were changed.

Figure 5: Preliminary plan

Figure 6: Improved plan

8.2.2 Failure in Price Assessment

Figures 7 and 8 show an example that was considered as a negative sample for the improved plan in the price evaluation. In this example, it was asked to find cheaper hotels, but the hotels in the improved plan were more expensive.

Figure 7: Preliminary plan

Figure 8: Improved plan

8.3 Comparison with GPT plug-in

Compared to the GPT plug-in, which only outputs either the hotel or activities, GenPerfectTrip is capable of considering the hotel location and suggesting activities along with the selected hotels.

Figure 9: System Generated Plan

Figure 10: GPT plug-in Generated Plan

9 Discussion and Learning

GenPerfectTrip demonstrated notable success in generating comprehensive and personalized trip plans for users, encompassing hotel selections and detailed trip itineraries, which perfectly align with their requirements and preferences

such as the destination, trip duration, budget constraints, and specific attractions of interest. One notable aspect is the application's ability to seamlessly combine both hotel selection and itinerary planning in a single solution, which set GenPerfectTrip apart, especially in comparison to the existing Expedia's plugin. Through the development of GenPerfectTrip, the most significant learning was the critical role of prompt engineering and the boundless possibilities that GPT offers. When starting a similar project in the future, the development team would build it on the top of this project and utilize prompt engineering and GPT to implement more detailed trip arrangements such as hotel booking, ticket purchasing, UBER requests, and restaurant reservations that come with the plan generated, enabling users to have all aspects of their trip meticulously prepared with just a single tap.

10   Individual Contribution

Junhong Chen is responsible for:

- constructing the system prompt for generating the preliminary plan.

- setting up the Django framework

- back-end logic and API integration

- the dynamic web scraping by utilizing the Beautiful Soup library, which included JSON parsing, URL construction, HTML parsing, and data extraction.

- the responsive front-end web interface.

- integrating JQuery into Django to handle front-end side logic and dynamic content updates.

Yiwei He is Responsible for:

- collection of the testing data set.

- constructing the system prompt for converting user inputs into JSON, and the two system prompts for plan improvement.

- testing and debugging the generation of preliminary and improved plan

- implementing the evaluation metrics

- designing the architecture and general workflow

- evaluation of system performance

11   Reference

White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., ... & Schmidt, D. C. (2023). A prompt pattern catalog to enhance prompt engineering with chatgpt. arXiv preprint arXiv:2302.11382. https://arxiv.org/abs/2302.11382

12 Permission

- permission to post video: wait till see video.

- permission to post final report: YES.

- permission to post source code: YES.

13 Appendix

13.1 Prompt for JSON prompt constructor

You are a trip planner that will help the users to come up with a trip plan. The user would give you ideas about their trip preference, you should extract information about destination, number of nights and price_range and output ONLY a json type. Price_range should be in numerical value for hotels to stay per night. This value should be related to the budget and number of nights. If one of the three information is not given, please give a value based on other information. For example, if the user doesn't know how long the trip will be, you should determine based on the destination and price_range. Price_range = budget/number of nights. The output destination should be a city to visit. All value must not be null. Price_range should take form like 100-200

13.2 Prompt for preliminary plan prompt constructor

Your job is to help the users generate a plan for their trip based on their requirements and hotel information.
This list stores the information for 10 hotels that the user can choose, and you need to do the following things:
step 1, help the user choose hotels the user should stay in and provide detailed information about the hotels. You can choose different hotels based on the itinerary every day.
step 2, generate a trip plan for the user. Recommend where they should go and what they should do. Don't ask anything just generate the entire trip plan for the user.
list:{hotel_info}

13.3 Prompt for improved plan prompt constructor

You are a trip planner that will take the provided plan and try to improve the plan according to the user's requirement. You should only change details based on user's requirement. If the user wants to improve on the activities, do not change anything about hotel, if the user wants to make change to the hotel information, output ONLY a json file based the new requirement in the form of destination, number_of_nights and prince range.Price_range should be in numerical value for hotels to stay per night. Price_range should take form like 100-200. When forming the output, you must take in account of the original plan. For example, if original hotel price is 500 and user wants cheaper hotel, it must be lower than 500.
Provided Plan:{original_plan}

13.4   Prompt for hotel improvement prompt constructor

You are a trip planner that will take the provided plan and try to improve the plan according to the hotel list and user input. You should not change the activities and only change the hotels to stay.
Provided Plan:{original_plan}
hotels list:{hotel_info}