Short Approach Description

0) I generated the camp description using LLM (Language Model): capm_text.txt

1) Upon receiving a user's query, I consult the LLM to determine if the user intends to enroll a child in the camp.

a) If the response is "YES," I transition to collecting essential details (parent's full name, phone number, email, and child's age).

b) If the response is "NO," I forward both the user's question and the camp description to the LLM, requesting an answer based on the provided description.

2) During the final stage of the information collection process, I engage the LLM to structure the information into a JSON format with specific organization.

Note: I maintain a complete record of the conversation history, which I append to the current request made to the LLM.

Open Questions

How would you optimize the process if you had more time?

1) I am currently utilizing a Language Model (LLM) to discern whether a parent intends to inquire about the camp or enroll their child in it. However, I believe that employing conventional Natural Language Processing (NLP) techniques could enhance efficiency in both time and cost aspects. My approach involves embedding the user's question and subsequently employing a vector similarity metric to ascertain whether the user's intention aligns with registering their child for the camp or not.

   By integrating established NLP techniques, I anticipate a more streamlined and resource-effective method for discerning user intent. This involves transforming the user's input into a vector representation and then measuring its similarity against predefined vectors associated with queries related to camp inquiries and child enrollment. This approach is likely to yield quicker results and potentially reduce computational costs compared to using a full-fledged Language Model.

2) I now retain all the previous questions and answers within the context. However, we have the option to eliminate the questions that have already been addressed (and

answered) from the context.

3) Currently, I consistently utilize the complete summer camp description as context. However, it's possible for me to extract only the sections that pertain to the specific question posed by the user. For instance, I can remove portions related to activity descriptions if the inquiry is about the age range for children. There are multiple methods to achieve this, such as adding annotations to the camp description or employing a rule-based approach to identify paragraphs directly linked to the question.

4) I believe it's necessary to limit the functionality of the LLM when it's employed as a conversational AI assistant for a particular task. For instance, I prefer the assistant not to respond to general questions such as "Who is the creator of Hamlet?" but rather guide the user back to the designated topic of the assistant's expertise.

5) Add Personalization, for example use a user’s name for a conversation; use ‘son’ or ‘daughter’ instead of ‘a kid’ etc.

6) Lastly, we have the option to fine-tune the model to customize it for a particular domain

How would you test the prompts' performance?

1. Regarding the Application Prompt, we can automate the simulation of user responses by employing a predefined set of "users." This set would encompass parent names, contact information, and the ages of their children. Technically, this involves inserting user-specific details into predefined contexts, inputting these sentences into the LLM, and subsequently comparing the user details generated by the LLM with the information we possess for a given user. Metrics such as Precision, Recall, F1, and Accuracy can be utilized to evaluate this process.

2. For Router Prompt the easiest way to test is a manual QA of “YES” (= user wants to sign a kid to the camp) and “NO” (user asks about some information)

3. For Question Prompt I would split testing to manually and automatic parts.

   Manual tests
   1) System behavior using various questions that parents might ask about the topic. Include various levels of complexity and formats (e.g., short queries, longer descriptions).

   2) Accurate Responses: Evaluate if the model provides accurate and relevant responses to each question. Check if the information it offers is aligned with the context of the query.

   3) Unstructured Questions: Include unstructured or ambiguous questions to evaluate the model's ability to clarify and provide meaningful responses.

4) Handling Specifics (dates, location, values etc): Evaluate the model's performance when questions involve specific details like dates, locations, or policies. Check if it provides accurate and specific answers.

5) Negative Responses: Test the model's ability to respond appropriately when it doesn't have information for a particular question.

Automatic
We can compile a reference set of questions along with their accurate answers, subsequently comparing the LLM's responses against the correct ones. This can be achieved through vector similarity or the BLEU metric, which is commonly used in automatic translation assessments.

Regarding precise elements like numerical values and distances, it's necessary to verify not only the similarity between the LLM's response and the reference truth but also ensure that the numbers themselves match accurately.

What edge cases do you think are not handled currently that you would add?

1) Relative dates. Dealing with relative dates can be challenging for LLMs. We must address questions like "Can my kid start in two weeks?" in a specific manner. One approach is to preprocess such questions by extracting the relative date and converting it to the current date.

2) Information was not covered in the provided description. Typically, LLMs cannot respond with "I don't know, please contact the camp staff." For instance, when asked about the Staff-to-Child Ratio at the camp, the LLM might respond with something like "The ratio is maintained at an optimal level."