Map GPT Playground: Smart Locations and Routes with GPT

Chiqun Zhang
chizhang@microsoft.com
Microsoft
Mountain View, CA, USA

Antonios Karatzoglou
ankaratz@microsoft.com
Microsoft
Mountain View, CA, USA

Helen Craig
hecraig@microsoft.com
Microsoft
Mountain View, CA, USA

Dragomir Yankov
dragoy@microsoft.com
Microsoft
Mountain View, CA, USA

ABSTRACT

People often ask questions for which the answer contains multiple locations or locations with additional context. The questions can be very natural and easy to understand by other people, yet if formulated as map queries, today's map search engines struggle to understand them. Here we look into three categories of such map queries: 1) queries with multiple explicitly stated locations, e.g. *'Show me directions from A to B through C, D, and E'*; 2) queries where the locations need to be inferred, e.g. *'Show me on the map all locations which James Bond visited in Casino Royale'*; 3) queries of locations where we request additional geographical, historical or other context, e.g. *'Show me a map of wildlife in Australia'*. We build a prototype system, called *Map GPT Playground*, and demonstrate with it how such queries can be seamlessly answered by combining the power of Large Language Models (LLM) with foundational maps services, such as geocoding, routing, etc. We describe the architecture of the system and reason over the abstractions needed for the system to be able to generalize across complex query intents and invoke suitable chains of services to fulfil them. Lastly, we demonstrate that in resolving complex location search queries, novel considerations emerge without prior analog, namely the set of returned locations needs to be spatially consistent and often to satisfy some inferred from the query temporal order.

CCS CONCEPTS

• **Information systems** → **Information retrieval query processing**; **Users and interactive retrieval**; **Retrieval tasks and goals**.

KEYWORDS

Maps, Route Planning, Large Language Models, GPT models, Bing Maps

SIGSPATIAL '23, November 13–16, 2023, Hamburg, Germany

ACM ISBN 979-8-4007-0168-9/23/11…$15.00
https://doi.org/10.1145/3589132.3625595

ACM Reference Format:
Chiqun Zhang, Antonios Karatzoglou, Helen Craig, and Dragomir Yankov. 2023. Map GPT Playground: Smart Locations and Routes with GPT. In *The 31st ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL '23), November 13–16, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3589132.3625595

1 INTRODUCTION

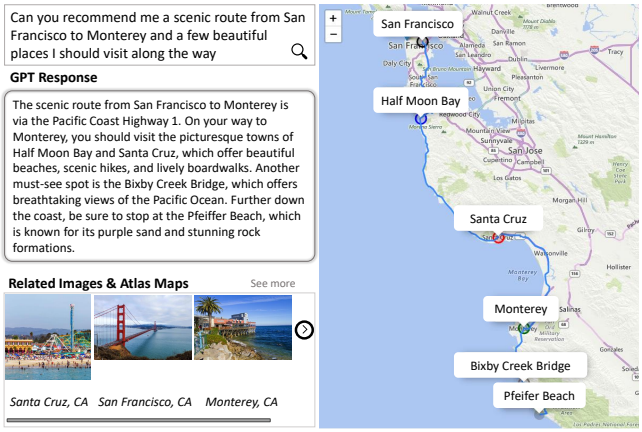

Figure 1: Map GPT Playground demonstration. Given a query, the system can identify multiple relevant geolocations, provide textual information and visual context about them, e.g. images and interactive map with optimized routes.

Consider the query *'Can you recommend me a scenic route from San Francisco to Monterey and a few beautiful places I should visit along the way'* in Figure 1. It uses a very natural way of expressing road trip intent, yet we tried this and all other examples in this paper, with multiple major map search engines and in-car navigation assistants and at present they all fail to resolve any of them. The reason is that map search engines can understand one [2] or two [4] explicit location within a query, but are not equipped to identify multiple explicitly or implicitly stated locations. Here we demonstrate *Map GPT Playground* - a system that uses GPT [6] to understand queries, such as the above and many more. In particular, in our demo system we focus on three broad classes of queries:

- **Explicit (multi) location queries**, where the locations are stated by users within the query, e.g. *'San Francisco to San*

Diego through Santa Barbara, Pasadena, and Irvine'. Users might be interested in simply seeing the locations on a map or have routes through them.

- **Implicit location queries**, where the locations are not specifically stated and the GPT agent generates them for us. These are queries such as the one in Figure 1, or *'Show me the locations which James Bond visited in Casino Royale'*.
- **Atlas queries**, where users ask for locations and certain thematic maps content that relates to them - geographic, historic, etc. E.g. *'Show me a map of wild life in Australia'*, or *'Show me a map of the main economic centers in Japan'*.

To achieve generality in understanding and answering the above queries, the system implements an architecture which applies GPT in several stages [7]. First, GPT is used in a generic *Intent Understanding Layer* where one or more intents are being derived, e.g. which of the above three categories the query falls into, whether users need just a map of the locations or also routes, etc. Second, GPT is used to identify whether it should invoke certain *foundation services (models)*, such as geocoding [1], routing [3], etc. Lastly, we use GPT to ensure two novel aspects that emerge when resolving multi-location queries, namely **spatial** and **temporal consistency**. Spatial consistency means that the locations we return are correct as a set, e.g. for a user in Pasadena, Texas returning it as one of the places for the explicit multi-location query *'San Francisco to San Diego through Santa Barbara, Pasadena, and Irvine'* would be wrong regardless of its geospatial proximity to the user. Within the set of all locations in the query Pasadena, California is more relevant. Temporal consistency is needed to satisfy the semantic order of the locations, if such is implied by certain words in the query. For instance, in the same query the term *'through'* suggests that San Diego should be visited last after the other cities.

2 ARCHITECTURE DESIGN AND IMPLEMENTATION

Figure 2 shows the proposed workflow. There are two major components: intent understanding and foundation models. Intent understanding detects whether the query is a map query and then labels it with a set of possible intent classes. Then relevant spatial entities are extracted by GPT. Based on the assigned labels, a handful of foundation models are invoked to provide useful information and enriched response.

2.1 Intent Understanding

2.1.1 Intent Detection. The first step of the proposed workflow is to understand the query intent. In our demo, we define four intents, which can easily be extended further. The intents cover the query classes mentioned in Introduction, as well as whether additional context is required.

- Implicit Locations Intent: Differentiates between the implicit and explicit query classes. Implicit queries require some or all locations to be further inferred by GPT, e.g. Figure 1.
- Atlas Map Intent: Detects whether the query is of the third class defined in Introduction.
- Routing Intent: The query is requesting a driving route with possible preferences or constraints. For example, *'I want to go from Seattle to San Francisco through Portland'*.

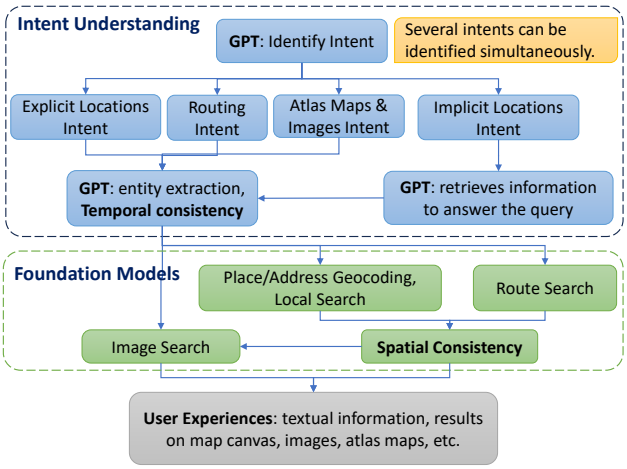

Figure 2: System overview. GPT is used in multiple reasoning steps to understand queries and extract entities. Various services act as foundation models providing detailed information.

- Location Image Intent: The result will be richer and lead to better user experience if we show images for locations derived from the query.

Given a query, we ask GPT to detect whether the query contains any of the above intents, represented with a binary label in the output (see prompt below). Detection for each intent is independent so a mixed combination of multiple intents is possible. In this work, we have iterated over multiple versions of the GPT prompt to identify what works best for the intent task. We found out that providing sufficient and diverse examples is paramount in achieving good intent understanding. One simplified version of the prompt including the output format is shown as follows:

You are an assistant designed to extract intent from a query:
Example 1:
User input: What locations did the characters pass through in 'The Grapes of Wrath'?
Output: Implicit:1 | Routing:1 | Atlas:0 | Image:0
Example 2:
User input: I need to go to the office at 1 Microsoft Way and then to SeaTac airport.
Output: Implicit:0 | Routing:1 | Atlas:0 | Image:0
Example 3:
User input: I want to go from Seattle to Sunnyvale through Portland
Output: Implicit:0 | Routing:1 | Atlas:0 | Image:1

2.1.2 Entity Extraction. Following intent understanding, if the query has implicit intent GPT is called with it to generate a model response. Then we use GPT again to extract all geo-entities from either the query itself (explicit query) or the model response. In this step, GPT is asked to complete the following sub-tasks.

- Extract relevant location entities, including source and destination locations. If source is not provided, output the user location as source.
- Label each extracted entity as Place, Business, or Address.

- **Temporal consistency:** Indicate whether the extracted entities should be visited following a temporal order (denoted as <T>).

The following demonstrates the prompt template used in this step. The prompt contains one example only for illustration.

You are an assistant designed to extract geospatial entities from text. Here are your tasks:

- You need to extract geospatial entities from the input text.

- You also need to label each entity into one of these categories: Address, Place, or Business.

- You need to understand if user plans to visit the entities following temporal order, using words such as 'before', 'after', etc. If so, output the entities ordered and add <T> at the end, otherwise, add <O>.

- You need to identify the start location from the input text. If the start location is not one of the extracted entities, put 'UserLocation' at the beginning of your output.

Here are some examples.

Example 1:

User input: I need to go to the office at 1 Microsoft Way and then to SeaTac airport

Entities: UserLocation(Address); 1 microsoft way (Address); SeaTac airport(Business) <T>

2.2 Foundation Models

2.2.1 Location Search. Once the entities are extracted, we utilize Bing Maps Location APIs to either geocode an address or a place, or search for a nearby business. Depending on the map service providers, different APIs could be used for different entity categories to achieve best result. For example, based on [5], *Location By Query API* is good for getting latitude and longitude coordinates that correspond to location information provided as a query string which supports both place queries or address queries (which include road queries too), *Local Search API* returns a list of business entities matching the textual query and centered around a location.

2.2.2 Image Search. When *Atlas Map* or *Location Image* intents are detected by Intent Understanding (Section 2.1), Bing Image Search API is called to retrieve images or photos relevant to the query. If the query intent is *Atlas* we call image search with the raw query as it already describes the thematic content of the requested atlas map, e.g. *'Historical map of 13-th century Spain'* or *'Map of wild life in Australia'*. In case of *Image* intent, we request travel related images for the detected locations. E.g., we augment the entity *Santa Cruz, CA* into an image search query *'travel images of Santa Cruz, CA'*.

2.2.3 Route Search. Given a source location, a destination location, and a set of stops, Bing Maps Route API is used to find the optimal route and estimated travel time. When the temporal order has been inferred by GPT (Section 2.1.2), the detected stops are visited following that order. Otherwise, Route API is asked to optimize the stops visiting order to minimize the route distance.

2.2.4 Spatial Consistency. One limitation of LLM responses is lack of spatial consistency. We study two types of spatial consistency problems. The first type happens when the location search is ambiguous. For example, whether *Pasadena* should refer to *Pasadena, Texas* or *Pasadena, California*. To ensure such spatial consistency, we request from the Location Search foundation models to return top 3 results for each location. For each set of possible locations, we create a convex hull. The set with smallest convex hull area is picked as the final set of locations. The second type of spacial consistency relates to routing proximity of locations inferred in *Implicit Intent* queries. For example, although Death Valley National Park is a famous place to visit in Southern California, it may not be a good stop when driving from Los Angeles to Las Vegas, even if suggested by GPT. We propose a post-processing method to filter the suggestions from GPT by leveraging Bing Route APIs. As shown in Algorithm 1, given source, destination and a set of potential stops, we first get the travel time between the source and destination without any stop ($\hat{t}$). Then we add each recommended stop i as a waypoint and get the new travel time with the API (t_i). If $t_i < \alpha\hat{t}$, where α is predefined maximum detour allowance larger than 1, we add that stop to the final suggestions list.

Algorithm 1: Steps to implement spatial route consistency.

Input: A list of stops **W**, source location **s**, destination location **d**, and scalar α - max detour allowance.
Result: An updated list of stops $\hat{W}$ satisfying proximity consistency.
Get travel time $\hat{t}$ between source **s** and destination **d** without any stop.
for *Each stop* w^i *in* $\boldsymbol{W}$ **do**
  Get travel time t^i between **s** and **d** through stop w^i.
  if $t^i < \alpha\hat{t}$ **then**
    $\hat{W}$ += $[w^i]$
  end
end

3 DEMONSTRATION

In this section, we present results reflecting the three query classes defined in Introduction. All examples in this section are "complex" queries that today's search engines cannot resolve, and figures in this section are snapshots from the actual demo system we have built.

3.1 Explicit Multi-location Queries

Figure 3 shows three queries with explicit multiple locations. Such queries fail when issued in today's map search engines, yet our demo system resolves them seamlessly with the help of GPT. On the left figure, the explicit intent and all locations are correctly identified from the query. The spatial consistency component ensures that the route goes through the correct locations, instead of similar named cities in remote locations. GPT also interprets the word *'through'* correctly, ensuring the right order for temporal consistency. The queries on the right have small, yet essential difference - *'then'* and *'but before that'* require different ordering for temporal consistency, which GPT infers correctly again. Spatial consistency is also ensured - there are multiple pizza places from the same chain, yet our system chooses the one closest to the library.

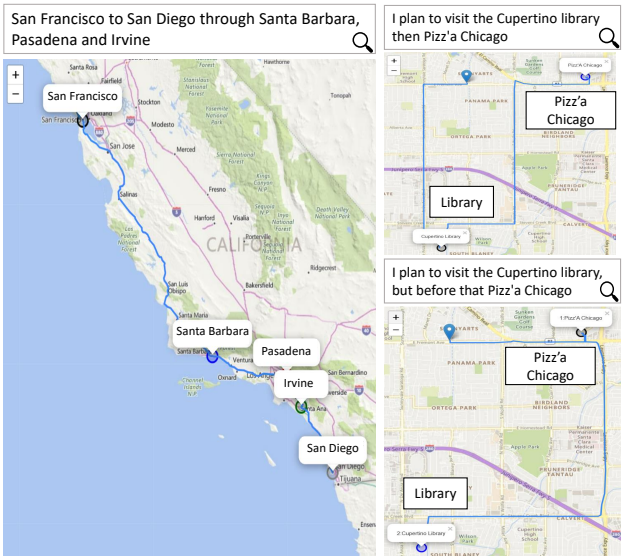

Figure 3: Explicit multi-location queries. Left: routing query with multiple spatially consistent stops. Right: the model is able to detect correct temporal order from words, such as *'then'* or *'before'* in the query.

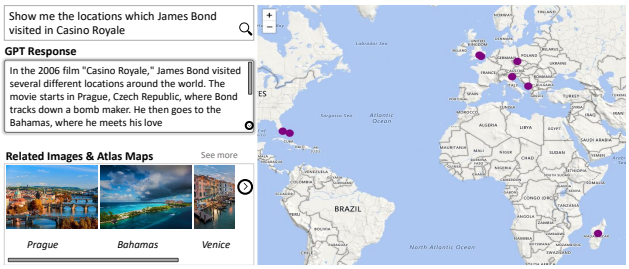

Figure 4: Implicit location queries. The proposed system is able to answer the query, show relevant places on map canvas, and present iconic photos for each place.

3.2 Implicit Location Queries

In Figure 1, we presented an example of implicit trip suggestions query. Also in this category fall queries inquiring about locations that are mentioned in books, movies, historical reference, etc. For instance *'Where did Martin Luther King Jr. deliver his speech: I have a dream?'*, or *'Show me the locations which James Bond visited in Casino Royale'*, Figure 4. As seen from the figure, the system first understands the implicit intent of the query, then GPT generates a response with detailed information. Finally, all relevant places are extracted and geocoded on the map canvas.

3.3 Atlas Queries

Often users are interested in certain thematic content related to a map of a location - flora or fauna, resources, historical view etc. Two examples are shown in Figure 5 - *'Show me a map of wildlife in Australia'* (left) and *'Map of main economic centers in Japan'*. Map search engines apply models that are capable of understanding

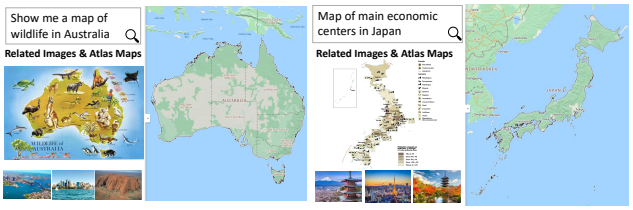

Figure 5: Atlas queries. Current map experiences can be enriched with different atlas maps and other images.

locations [2] but not the additional thematic context. Having atlas images in map search can serve information needs in a very convenient way. Users can have overall understanding of the context from the atlas images, which however often do not have much geographic details. Such details can be obtained by exploring the interactive map canvas for the locations. For instance, we can zoom into a particular location to see what cities might be close to an economic center in Japan. Such capability opens the proposed map system to applications beyond pure utility, letting people also explore and create emotional connections with the world.

4 CONCLUSION

In this paper, we present a novel prototype system, *Map GPT Playground*, combining GPT with foundational maps services. The system is able to understand complex query intents, extract critical entities and requirements, resolve and aggregate detailed information, and present enriched response to the user. Through queries from multiple intents including explicit multiple location queries, complex implicit queries, and atlas queries, we have demonstrated that *Map GPT Playground* not only is able to resolve complex location searches, but also follows spatial consistency as well as temporal order. The proposed system lays out a promising direction for future map services which go beyond pure utility applications towards emotional connection and memory creation.

REFERENCES

[1] Pavel Berkhin, Michael R. Evans, Florin Teodorescu, Wei Wu, and Dragomir Yankov. 2015. A New Approach to Geocoding: BingGC. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '15)*.

[2] Helen Craig, Dragomir Yankov, Renzhong Wang, Pavel Berkhin, and Wei Wu. 2019. Scaling Address Parsing Sequence Models through Active Learning. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '19)*.

[3] Daniel Delling, Andrew V Goldberg, Thomas Pajor, and Renato F Werneck. 2012. Customizable route planning in road networks. In *Proceedings of the International Symposium on Combinatorial Search*.

[4] Jasper Huang, Chiqun Zhang, Dragomir Yankov, Maryam Mousaarab Najafabadi, and Tsheko Mutungu. 2022. Active Learning for Transformer Models in Direction Query Tagging. In *Proceedings of the 30th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '22)*.

[5] Microsoft. [n. d.]. *Bing Maps REST Services.* https://learn.microsoft.com/en-us/bingmaps/rest-services/

[6] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. arXiv:2203.02155 [cs.CL]

[7] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face. arXiv:2303.17580 [cs.CL]