Synergizing Spatial Optimization with Large Language Models for Open-Domain Urban Itinerary Planning

Yihong Tang*
TuTu. AI
University of Hong Kong
yihongt@connect.hku.hk

Zhaokai Wang*
Shanghai Jiao Tong Univeristy
wangzhaokai@sjtu.edu.cn

Ao Qu*
TuTu. AI
Massachusetts Institute of Technology
qua@mit.edu

Yihao Yan*
TuTu. AI
yanyihao@tutu-ai.com

Kebing Hou
TuTu. AI
houkebing@tutu-ai.com

Dingyi Zhuang
TuTu. AI
Massachusetts Institute of Technology
dingyi@mit.edu

Xiaotong Guo
Massachusetts Institute of Technology
xtguo@mit.edu

Jinhua Zhao†
Massachusetts Institute of Technology
jinhua@mit.edu

Zhan Zhao†
University of Hong Kong
zhanzhao@hku.hk

Wei Ma†
The Hong Kong Polytechnic University
wei.w.ma@polyu.edu.hk

ABSTRACT

In this paper, we introduce the novel task of Open-domain Urban Itinerary Planning (OUIP), a paradigm designed to generate personalized urban itineraries from user requests articulated in natural language. This approach is different from traditional itinerary planning, which often restricts the granularity of user inputs, thus hindering genuine personalization. To this end, we present ItiNera, an OUIP system that synergizes spatial optimization with large language models (LLMs) to provide services that customize urban itineraries based on users' needs. Upon receiving the user's itinerary request, the LLM first decomposes it into detailed components, identifying key requirements, including preferences and dislikes. Then, we use these specifics to select candidate POIs from a large-scale collection using embedding-based Preference-aware POI Retrieval. Finally, a preference score-based Cluster-aware Spatial Optimization module clusters, filters, and orders these POIs, followed by the LLM for detailed POI selection and organization to craft a personalized, spatially coherent itinerary. Moreover, we created an LLM-based pipeline to update and personalize a user-owned POI database. This ensures up-to-date POI information, supports itinerary planning, pre-trip research, POI collection, recommendations, and more. To the best of our knowledge, this study marks the first integration of LLMs to innovate itinerary planning, with potential extensions for various urban travel and exploration activities. Offline and online evaluations demonstrate the capacity of our system to deliver more responsive, personalized, and spatially coherent itineraries than current solutions. Our system, deployed on an online platform, has attracted thousands of users for their urban travel planning.

CCS CONCEPTS

• **Information systems** → **Language models**; **Location based services**; • **Computing methodologies** → **Spatial and physical reasoning**.

KEYWORDS

Itinerary Planning; Intelligent Transportation; Location-based Services; Spatial Optimizations; Large Language Models

ACM Reference Format:
Yihong Tang, Zhaokai Wang, Ao Qu, Yihao Yan, Kebing Hou, Dingyi Zhuang, Xiaotong Guo, Jinhua Zhao, Zhan Zhao, and Wei Ma. 2024. Synergizing Spatial Optimization with Large Language Models for Open-Domain Urban Itinerary Planning. In *Proceeding of the 13th International Workshop on Urban Computing (KDD UrbComp '24), August 25–29, 2024, Barcelona, Spain.* ACM, Barcelona, Spain, 9 pages.

*These authors contributed equally to this work.
†Corresponding authors.

KDD UrbComp '24, August 25–29, 2024, Barcelona, Spain

1 INTRODUCTION

As a novel form of urban travel and exploration, city tours invite travelers to wander through city streets, delve into historical sites, and immerse themselves in local culture. Hence it offers a more efficient, dynamic, and cost-effective travel experience compared to traditional tourism. However, planning a city tour is essentially an urban itinerary planning problem [8], a complex procedure that involves gathering travel-related information, selecting POIs, mapping out routes, and customization for diverse user needs.

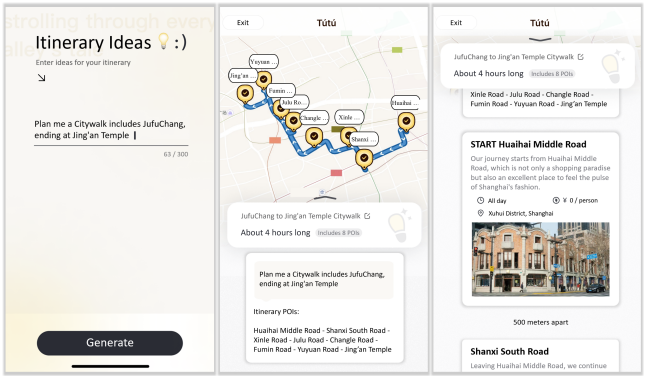

Figure 1: The OUIP problem and the deployed OUIP service.

Existing itinerary planning studies focus on traditional tourism. They typically consider coarse-grained user requirements, such as must-see POIs [26], categories[2], geographical considerations [19, 20], and time budget [10, 39] as constraints to improve the logical arrangement of sites and optimize the quality of an itinerary [3, 24]. While these spatial optimization approaches manage to maintain the quality of POIs and spatial coherency of the planned itinerary, they fail to ensure personalization and overall quality, thereby failing to satisfy the diverse demands of users. A high-quality itinerary for a city tour should transcend a mere list of POIs, and align with users' specific requirements and lifestyles.

To elaborate on this, we summarize the highlighting features of city tours that differentiate itself from traditional tourism:

- Dynamic Information: City tours involve rapidly changing POIs and requiring up-to-date information and exploration of temporary events, as contrary to traditional itineraries.
- Personalization: City tours prioritize individual preferences over universally recognized POIs in traditional tourism.
- Diverse Constraints: City tours planning take into account more complex constraints e.g. personal interest or accessibility requirements, necessitating more intricate planning.

Conventional itinerary planning approaches are challenging for city tours due to the above-discussed features. Therefore, we propose open-domain urban itinerary planning (OUIP), which involves *generating personalized urban travel itineraries based on user preferences in natural language*, as depicted in Fig. 1.

Recently, LLMs [16] demonstrate impressive applications in understand user needs and handle simple planning tasks [33]. However, the limitations of LLMs become apparent in itinerary planning [33] due to reliance on non-real-time information, incomplete knowledge, and a lack of spatial awareness. LLM-generated itineraries could be circuitous, lack of detail, and include hallucinated POIs, making them less practical for actual use in itinerary planning. Beyond the methodological gaps, evaluation of OUIP also presents difficulties due to lack of datasets and open-domain nature of the OUIP task, rendering rule-based metrics insufficient.

Given above challenges, this paper proposes ItiNera, a holistic OUIP system that synergizes spatial optimization with LLMs. Specifically, ItiNera features a *User-owned POI Collection* module to automatically scrape and extract POI features from web content, dynamically updating a user-owned, personalized POI database to ensure the customization and timeliness of POI information. ItiNera consists of five modules: *User-owned POI Collection*, *Request Decomposition*, *Preference-aware POI Retrieval*, *Cluster-aware Spatial Optimization*, and *Itinerary Generation*. All five modules are LLM-assisted, and details will be presented later. Importantly, as the OUIP is a novel task, we collect a validation dataset and develop a series of metrics for a comprehensive evaluation framework. Lastly, ItiNera has been deployed as a real-world service that attracts thousands of users within months.

Overall, the main contributions of our work are as follows:

- We introduce the Open-domain Urban Itinerary Planning (OUIP) problem, designed to provide personalized urban travel itineraries based on users' natural language inputs.
- We develop an LLM-based OUIP system, ItiNera. To the best of our knowledge, this is the first study that combines spatial optimization with LLMs to create spatially coherent, fine-grained urban itineraries tailored to users' requests. The system can be extended to various local travel and entertainment activities, such as urban explorations, multi-day travel, and more.
- We designed several rule-based and GPT-based metrics to measure the quality and personalization of generated itineraries and provide a benchmark for future research.
- Extensive experiments on real-world dataset demonstrate that ItiNera manages to create personalized, spatially coherent and high-quality urban itineraries that meet user requirements; ItiNera is also deployed in TuTu online OUIP service. User feedback validates the effectiveness of our system in real-world scenarios.

2 RELATED WORKS

2.1 LLMs in Urban Application

Since ChatGPT [16], Large Language Models (LLMs) have received widespread attention and demonstrated strong capabilities in tasks such as language processing, instruction-following, planning, and learning. Existing applications of LLMs are concentrated in agent [28, 32] and prompt [31, 37] design. However, the urban applications using LLMs remain in the early stages. Recent studies highlight the potential of LLMs in urban data processing [36] and urban planning [43]. These works reveal LLMs' capabilities in predicting human mobility patterns [15, 35], including during public events [13], and emphasize their predictive strength [30, 34]. In transportation, LLMs contribute to traffic safety analysis [40], enhance traffic forecasting [4], and automate accident report generation [42], demonstrating their wide applicability in urban transportation [41].

Leveraging LLMs for automated travel planning has recently gained public interest. Concurrently, TravelPlanner [33] proposes a sandbox environment with various tools for benchmarking LLMs on long-term (minimum 3 days) travel planning, revealing LLMs' current limitations in handling complex planning tasks. Given this, we present ItiNera, a holistic system designed for OUIP. Unlike TravelPlanner, our system focuses on fine-grained OUIP, addressing urban itinerary planning within a single day, but can be seamlessly extended to multi-day travel planning.

2.2 Itinerary Planning

Current research on IP focuses on creating itineraries based on a set of POIs. Some methods directly optimize the spatial utilities of the

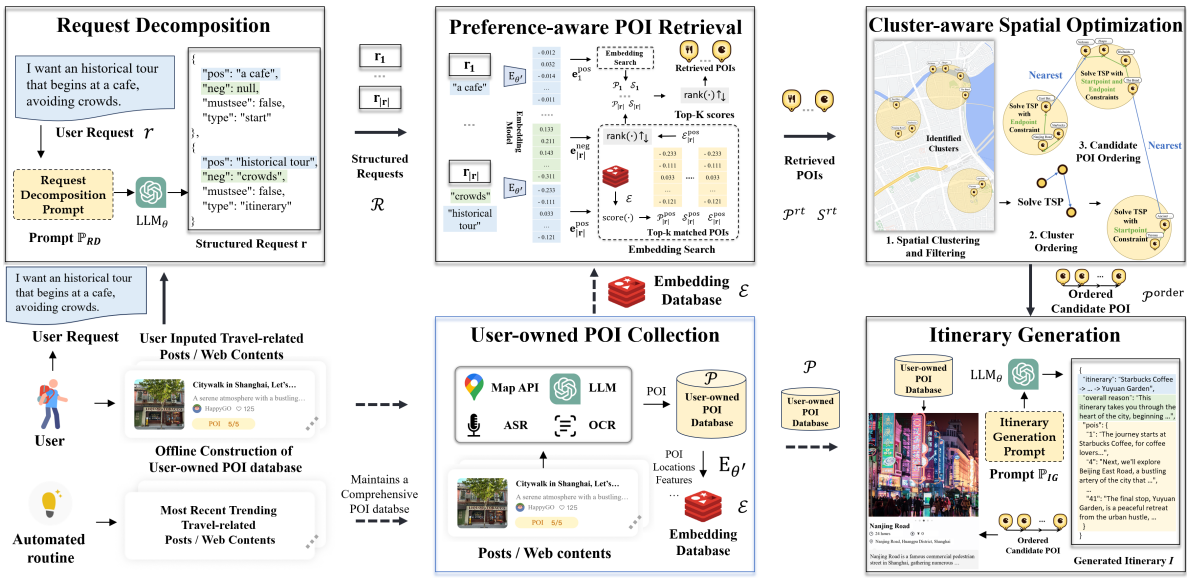

Figure 2: An overview of the ItiNera system.

itinerary to obtain spatially coherent itineraries, while others define IP as an Orienteering Problem (OP) and consider constraints that include time [10, 39], space [19, 20], must-see POIs [26], categories [2], and their combinations [6, 38], optimizing POI utilities such as location popularity [7, 27] to indirectly ensure the spatial coherence and quality of the itinerary. Although these methods can ensure the quality of POIs in the itinerary, their ability to personalize is limited. Some recommendation-based methods [9, 25] could be applied to the IP task. However, these methods often depend on historical user behavior data. Overall, existing IP methods struggle with open-domain, user natural-language inputs, failing to generate personalized itineraries, making them unsuitable for OUIP.

3 PRELIMINARIES

Definition 1 (User Request). A user request r is a text string containing the user's itinerary needs in natural language, including hard requirements (constraints) and general preferences.

Definition 2 (Point of Interest). A Point of Interest (POI) p represents a specific location, like landmarks, restaurants, or cultural sites, characterized by its geographical location, category, descriptions, and other relevant context features.

Definition 3 (OUIP). Given a user request r and the user-owned POI set $\mathcal{P} = \{p_j\}_{j=1}^{N}$, the OUIP problem aims to find an itinerary generator $\mathcal{G}_\Theta$ to select and order a subset of POIs from $\mathcal{P}$ to create a coherent travel itinerary that optimally aligns with the user's requests r while adhering to spatio-temporal considerations:

$$I \sim \mathcal{G}_\Theta(\mathcal{P}|r), \tag{1}$$

where Θ is the parameters of the itinerary generator $\mathcal{G}$, and the generated itinerary I is an ordered list of POIs.

4 METHODOLOGIES

4.1 Overview

In this section, we provide details of ItiNera, which consists of five modules as depicted in Fig. 2. We first incorporate a *User-owned POI Collection* (UPC) module to facilitate travel information gathering and user-owned POI database construction. To plan an itinerary that meets a user's request, we design a *Request Decomposition* (RD) module to accurately interpret and extract independent user preferences. Subsequently, we develop a embedding-based *Preference-aware POI Retrieval* (PPR) Module that fetches the top-K most relevant POIs to form the retrieved POI set for the itinerary. To ensure a spatially coherent itinerary, we employ a *Cluster-aware Spatial Optimization* (CSO) module to spatially filter and rank retrieved POIs by solving the hierarchical traveling salesman problem, thereby obtaining the candidate POIs. Lastly, the *Itinerary Generation* (IG) module integrated the ordered candidate POI set with multiple constraints to construct the prompt for LLM's completion. This approach enables LLM to generate travel routes that are spatial-temporally rational and align with user requests. We provide details for each module in the following subsections.

4.2 User-owned POI Collection

In OUIP, the way of travel information acquisition has undergone a transformation from conventional tourism due to technological advancements. Base on our user and market research, travelers increasingly rely on social media for pre-trip information gathering. Notably, user-generated content is more authentic and current in terms of customer experience than official sources of information.

To this end, we've developed an automated pipeline that extracts POI information from social media platforms. Our pipeline enables users to input links of travel posts. With each new post, it leverages LLMs to extract POIs and their descriptions, calls Map APIs and embedding models to obtain the locations and embeddings of these POIs, and integrates the information into the user-owned

POI $\mathcal{P}$ and embedding database $\mathcal{E}$. This feature enables users to create their personalized POI database, maintain the POI information's timeliness, and customize their travel itineraries based on it, thereby elevating the overall quality of TuTu's online services. We also execute a daily automated routine to aggregate POIs by extracting data from the most recent trending posts across multiple cities. This approach maintains and updates a comprehensive POI database, serving as a complement to the user-owned POI database when needed. Additionally, this strategy can alleviate the cold start problem for both POI acquisition and the OUIP service.

4.3 Request Decomposition

Upon receiving user requests, we leverage the natural language processing capabilities of LLMs to extract structured user requests in this module. A single user request r can be decomposed into multiple independent subrequests, each could be generally classified according to their granularity, specificity, and attitude. For granularity, we have (1) POI-level subrequests and (2) itinerary-level subrequests. In terms of specificity, there are (1) specific subrequests and (2) vague subrequests. Regarding attitude, we identify (1) positive subrequests and (2) negative subrequests. Thus, we prompt LLM to decompose the user request r base on the identified categories. Let LLM_θ denote a pre-trained LLM with parameters θ:

$$\mathcal{R} \sim \text{LLM}_\theta\left(\mathbb{P}_{RD}\left(r\right)\right), \tag{2}$$

where $\mathbb{P}_{RD}$ wraps the request r with instructions and few-shot input-output examples. A simplified version of $\mathbb{P}_{RD}$ is shown above. The result is a set of decomposed independent structured subrequests $\mathcal{R} = \{\mathbf{r}_i\}_{i=1}^{|\mathcal{R}|}$. Each subrequest $\mathbf{r}_i$ comprises four keys: 'pos', 'neg', 'mustsee', 'type', with their corresponding values $\mathbf{r}_i^{\text{pos}}$, $\mathbf{r}_i^{\text{neg}}$, $\mathbf{r}_i^{\text{must}}$, $\mathbf{r}_i^{\text{type}}$. Here, 'pos' and 'neg' are strings indicating the user's likes and dislikes contained in the subrequest, 'mustsee' is a boolean value, and 'type' indicates whether the subrequest targets the starting POI, ending POI, a specific POI, or the entire itinerary, respectively. An example of subrequests is in the upper left of Fig. 2.

Simplified prompt $\mathbb{P}_{RD}$ for Request Decomposition.

```
Please help me break down a user's requirement description into multiple independent
requirements, each including both positive and negative requirements.

### Output Format:
Return a list:
- **pos**: The positive requirement, representing what the user wants.
- **neg**: The negative requirement, generally what the user does not want, dislikes, or
refuses.
- **mustsee**: Indicates whether this is a must-visit POI, this field is `true` or `false`.
- **type**: Indicates whether the requirement is for a "place" or an "itinerary".
- Your return should be a list in the following format:
[
    {{
        "pos": "positive requirement", (excluding negative requirements)
        "neg": "negative requirement" (what's not wanted, disliked, refused),
        "mustsee": true (whether it's a must-see POI),
        "type": "place"
    }}, ...
]

### Output Guidelines
- Return a JSON list, each item in the list is a dictionary containing "pos", "neg",
"mustsee", and "type" key-value pairs.
- The list can be empty; if empty, just return a JSON list.

### User Input
{user_input}
```

For those subrequests identified with the identified 'type' as "POI" and the value of the key 'mustsee' is true, we leverage their 'pos' strings to conduct fuzzy string match over the POI names stored in the user-owned POI database to retrieve the must-see POIs $\mathcal{P}^{\text{must}} = \{p_j^{\text{must}}\}_{j=1}^{|\mathcal{P}^{\text{must}}|}$.

4.4 Preference-aware POI Retrieval

To generate itineraries that meet user requests, the initial step involves selecting POIs that match users' preferences. Although LLMs are advanced in language comprehension, their capabilities are limited by their context window size [14, 18]. Additionally, the cost of using LLMs is proportional to the number of input tokens, and their inference speed could hinder our system's response time. Considering these limitations and the large scale of POIs in the real world, we design an embedding-based preference-aware POI retrieval approach to select candidate POIs.

Specifically, consider a single subrequest $\mathbf{r}_i$, we first employ an embedding model $\text{E}_{\theta'}$ to encode both the 'pos' and 'neg' fields:

$$\mathbf{e}_i^{\text{pos}} = \text{E}_{\theta'}\left(\mathbf{r}_i^{\text{pos}}\right); \mathbf{e}_i^{\text{neg}} = \text{E}_{\theta'}\left(\mathbf{r}_i^{\text{neg}}\right), \tag{3}$$

where θ' denotes the parameters of the embedding model E, and $\mathbf{e}_i^{\text{neg}}, \mathbf{e}_i^{\text{pos}} \in \mathbb{R}^d$ are two resulted d-dimensional embeddings.

Ideally, we want the queried POIs best fit the positive subrequest while avoiding the negative subrequest. To achieve this, we first use the calculated positive embedding $\mathbf{e}_i^{\text{pos}}$ to retrieve k POIs $\mathcal{P}_i^{\text{pos}} = \{p_{i,j}^{\text{pos}}\}_{j=1}^k$ with top similarity scores $\mathcal{S}_i^{\text{pos}} = \{s_{i,j}^{\text{pos}}\}_{j=1}^k$ and embedding $\mathcal{E}_i^{\text{pos}} \in \mathbb{R}^{k\times d}$, where $p_{i,j}^{\text{pos}}$ and $s_{i,j}^{\text{pos}}$ represent the jth POI and score for ith positive sub-request, respectively. Then, to ensure avoiding negative POIs, we compute the similarity scores of $\mathcal{E}_i^{\text{pos}}$ and $\mathbf{e}_i^{\text{neg}}$ and rerank the retrieved POIs base on the gap between POI-wise positive and negative similarity scores.

Suppose we use $\mathcal{E} \in \mathbb{R}^{N\times d}$ to denote the POI embeddings precomputed over POI attributes that stored in the user-owned embedding database, the above process could be formulate as:

$$\mathcal{P}_i^{\text{pos}}, \mathcal{S}_i^{\text{pos}}, \mathcal{E}_i^{\text{pos}} = \text{score}^k\left(\mathbf{e}_i^{\text{pos}}, \mathcal{E}\right) \tag{4}$$

$$\mathcal{P}_i^{\text{neg}}, \mathcal{S}_i^{\text{neg}}, \mathcal{E}_i^{\text{neg}} = \text{score}\left(\mathbf{e}_i^{\text{neg}}, \mathcal{E}_i^{\text{pos}}\right) \tag{5}$$

$$\mathcal{P}_i, \mathcal{S}_i = \text{rank}\left(\mathcal{P}_i^{\text{pos}}, \mathcal{S}_i^{\text{pos}} - \mathcal{S}_i^{\text{neg}}\right), \tag{6}$$

where the $\text{score}(\cdot)$ function measures embedding similarities, and the superscript k indicates that the function only returns the top-k results based on the similarity scores. The $\text{rank}(\cdot)$ reorders POIs based on their similarity scores from highest to lowest.

Lastly, we select the POIs with the highest top-k summed-scores (as preference scores) from the unioned set of all retrieved POIs. This forms the final retrieved POI set $\mathcal{P}^{rt}$ for the user request r:

$$\mathcal{P}^{rt}, \mathcal{S}^{rt} = \text{rank}^k\left(\cup_{i=1}^{|\mathcal{R}|}\left(\mathcal{P}_i, \mathcal{S}_i\right)\right) \cup \left(\mathcal{P}^{\text{must}}, \mathcal{S}^{\text{must}}\right), \tag{7}$$

where the set $\mathcal{S}^{\text{must}}$ is assigned large values to ensure must-see POIs are included in the final itinerary by subsequent modules. During the union process, the scores of the same POI under different subrequests are summed to form the final preference scores.

4.5 Cluster-aware Spatial Optimization

Travelers tend to move from one cluster of POIs to another [2]. This can be attributed to the efficiency and convenience of exploring POIs within a proximal area, reducing the time and effort involved

in transit. Therefore, spatially filtering all retrieved POIs and sequencing their visitation order is essential in order to inform large language model, which has exhibited limited capabilities for understanding layout and optimizing routes in 2D space. To achieve this, we compute spatial clusters of the retrieved POIs and then select candidate POIs based on geographical proximity and preference scores. We address cluster-aware spatial optimization by solving a hierarchical traveling salesman problem [11].

4.5.1 Obtain Candidate POI via Spatial Clustering and Filtering. Given the set of retrieved POIs, $\mathcal{P}^{rt}$, we create a spatial proximity graph G using a distance matrix D. In this graph, each node denotes a POI, and edges connect nodes (POIs) that are closer than a specific distance threshold τ. A community detection algorithm is applied to G so that all the nodes are divided into multiple mutually exclusiveclusters. Next, we iteratively select the clusters with the largest summed preference scores derived from the PPR module until the total number of selected POIs reach a predefined threshold N^c, and the POIs in these clusters form the candidate POIs $\mathcal{P}^c$ for the user request r. The above process is detailed in Algo. 1.

Algorithm 1 Spatial Clustering and Selection of Candidate POIs

Input: Retrieved POI set $\mathcal{P}^{rt}$ with scores $\mathcal{S}^{rt}$, Distance threshold τ, Candidate POIs num threshold N^c
Output: Spatial Clusters C, Candidate POIs $\mathcal{P}^c$

// Spatial Clustering
$G \leftarrow (V, E)$ with $V \leftarrow \mathcal{P}^{rt}, E \leftarrow \emptyset$; $C \leftarrow \emptyset$; $\mathcal{P}^c \leftarrow \emptyset$
for $p_a^{rt}, p_b^{rt} \in V$ with $a \neq b$ **do**
  if $distance(p_a^{rt}, p_b^{rt}) < \tau$ **then**
    $E \leftarrow E \cup \{(p_a^{rt}, p_b^{rt})\}$
  end if
end for
while $V \neq \emptyset$ **do**
  $c \leftarrow$ largest clique in G
  $C \leftarrow C \cup \{c\}$; $V \leftarrow V \setminus c$
end while
// Selection of Candidate POIs
for each cluster $c \in C$ **do**
  $s_c^{rt} \leftarrow \sum_{p_j \in c} s_j^{rt}$
end for
Sort C in descending order of $\mathcal{S}^c = \{s_c^{rt}\}_{c=1}^{C}$
while $|\mathcal{P}^c| < N^c$ **do**
  $c_{\max} \leftarrow \arg\max_{c \in C} s_c^{rt}$
  $\mathcal{P}^c \leftarrow \mathcal{P}^c \cup c_{\max}$; $C \leftarrow C \setminus \{c_{\max}\}$
end while
return $C, \mathcal{P}^c$

4.5.2 POI Ordering via Solving a Hierarchical TSP. After obtaining the spatial clusters C, we further optimize the visitation order of the candidate POIs to ensure a spatially coherent itinerary. This process first involves determining the access order of each cluster. Then within each cluster, we determine the visitation order of POIs by solving TSPs with starting and ending POI constraints. Specifically, the start and end points for each cluster are selected based on the proximity of POIs in adjacent clusters as described in Fig. 2. We provide an Algo. 2 below that illustrates the procedure.

Algorithm 2 Hierarchical TSP for POI Ordering

Input: Spatial clusters C, candidate POIs $\mathcal{P}^c$, distance matrix D
Output: Ordered list of candidate POIs $\mathcal{P}^{\text{order}}$

// POI Ordering
$C^{\text{order}} \leftarrow \text{SolveTSP}(C, D)$; $\mathcal{P}^{\text{order}} \leftarrow \emptyset$
for each cluster c in C^{order} **do**
  $p^c_{\text{start}}, p^c_{\text{end}} \leftarrow \text{GetClusterEndpoints}(c, C^{\text{order}}, D)$
  $c_{\text{path}} \leftarrow \text{SolveConstraintTSP}(c, p^c_{\text{start}}, p^c_{\text{end}}, D)$
  $\mathcal{P}^{\text{order}} \leftarrow \mathcal{P}^{\text{order}} \cup c_{\text{path}}$
end for
// Start POI Selection and POI Reordering
$p_{\text{start}} \leftarrow \text{Select}(\mathcal{P}^{\text{order}})$
$\mathcal{P}^{\text{order}} \leftarrow \text{Reorder}(\mathcal{P}^{\text{order}}, p^{\text{order}}_{\text{start}})$
return $\mathcal{P}^{\text{order}}$

The described process facilitates an optimized and coherent traversal among the selected POIs. Moreover, employing a hierarchical approach to solve the TSP with a large number of POIs. 'SolveTSP' and 'SolveTSPWithEndpoints' represent procedures to solve the standard and constrained TSP, respectively. The function 'GetClusterEndpoints' determines the start and end points for each cluster. Additionally, the ordering of POIs is influenced by the choice of starting point. Hence, 'Select' identifies the itinerary's starting point p_{start} from decomposed subrequests $\mathcal{R}$ or, if unavailable, by prompting an LLM with $\mathcal{P}^c$ and r to select p_{start}. Then, 'Reorder' function ensures that the POIs follow the original order of precedence in $\mathcal{P}^{\text{order}}$ to form the final $\mathcal{P}^{\text{order}}$ starts from p_{start}.

4.6 Itinerary Generation

Simplified prompt $\mathbb{P}_{IG}$ for final itinerary generation.

```
Please consider carefully and use the provided "Candidate POIs" list to craft a one-day
itinerary in the form of an engaging and realistic travel story.

## Itinerary Information:
- **Candidate POIs**: {p_order}
1: "Time Cafe, Dongcheng District. Located in the bustling Nanluoguxiang, this cafe is a
popular spot for photo enthusiasts and also a cat cafe. Surrounded by beautiful roses, it's
perfect for capturing memories."
2: . . .
- **Must-see POIs**: {r_must}
- **Keyword Requirements**: {r_pos}
- **User's Original Request**: {r}
- **Start POI**: {p_start}
- **End POI**: {p_end}

## Constraints:
- Only choose POI from the **Candidate POI** list and in ascending order.
- Bars should be at the end of the itinerary, while coffee shops should not be the last POI.

## Task Description:
1. You have approximately {hours} hours. Select suitable POIs  from the **Candidate POIs**
list in ascending order, ensuring your itinerary is selectively filtered and does not include
all POIs from the list.
2. All POIs added to the itinerary must follow the ascending order of **Candidate POIs**.
3. Generate a JSON file containing all selected POIs.

## Output Format:
{{
    "itinerary": "List of POIs, separated by '->'"
    "Overall reason": "Overall recommendation reason for the designed day trip itinerary",
    "pois": {{
        "n": "Description and recommendation reason for each POI", ...
    }}
}}
```

Selecting any subset from $\mathcal{P}^{\text{order}}$ guarantees a spatially coherent itinerary, but a high-quality itinerary should also comply with additional constraints, such as matching the user's time availability and ensuring practicality. For example, it should avoid consecutive

Method	Shanghai							Qingdao						
	Rule-based Metrics				GPT-based Metrics ↑ (%)			Rule-based Metrics				GPT-based Metrics ↑ (%)		
	RR ↑ (%)	AM ↓ (meters)	OL ↓	FR ↓ (%)	PQ	IQ	Match	RR ↑ (%)	AM ↓ (meters)	OL ↓	FR ↓ (%)	PQ	IQ	Match
IP [7]	6.4	1573.3	2.96	/	30.3	26.2	17.8	7.6	4134.3	2.86	/	23.6	16.8	20.2
Ernie-Bot 4.0 [23]	15.7	513.5	0.91	15.2	42.1	46.5	42.5	27.2	776.2	0.78	21.4	43.4	38.2	33.3
GPT-3.5 [16]	16.6	422.4	0.83	13.5	40.4	43.1	45.4	25.5	691.5	0.55	22.0	33.4	39.0	46.6
GPT-4 [17]	18.0	267.2	0.56	8.2	45.0	48.2	46.9	27.3	569.4	0.49	19.6	46.6	48.7	48.4
GPT-4 CoT [31]	18.4	258.3	0.49	7.5	/	/	/	30.2	542.6	0.43	17.8	/	/	/
Ours	**31.4**	**86.0**	**0.42**	/	**69.8**	**64.6**	**72.0**	**35.4**	**225.8**	**0.26**	/	**71.2**	**68.2**	**67.8**
Method	Beijing							Hangzhou						
	Rule-based Metrics				GPT-based Metrics ↑ (%)			Rule-based Metrics				GPT-based Metrics ↑ (%)		
	RR ↑ (%)	AM ↓ (meters)	OL ↓	FR ↓ (%)	PQ	IQ	Match	RR ↑ (%)	AM ↓ (meters)	OL ↓	FR ↓ (%)	PQ	IQ	Match
IP [7]	3.3	3034.2	2.26	/	27.8	18.2	20.4	1.8	1744.4	1.52	/	34.8	31.4	22.5
Ernie-Bot 4.0 [23]	18.8	379.4	0.74	12.8	31.2	34.8	32.1	12.9	605.2	1.17	24.4	43.6	34.3	38.2
GPT-3.5 [16]	19.7	347.8	0.58	14.3	29.2	40.5	43.8	14.4	665.4	1.16	19.8	41.2	40.8	32.8
GPT-4 [17]	20.6	342.6	0.52	11.1	45.4	43.6	45.2	14.8	746.1	1.09	23.2	46.2	39.6	39.4
GPT-4 CoT [31]	21.0	327.7	0.54	10.2	/	/	/	15.5	455.0	1.09	18.6	/	/	/
Ours	**28.4**	**79.2**	**0.46**	/	**59.2**	**67.6**	**75.2**	**21.4**	**30.5**	**0.12**	/	**61.6**	**65.4**	**68.3**

Table 1: Overall results. GPT-based metrics represent the win rate against GPT-4 CoT.

restaurant visits and ensure venues are visited at appropriate times, such as not planning bars in the morning or coffee shops late at night. Given the complex nature of these constraints, traditional optimization-based algorithms can become excessively complicated and lack variability [26, 38], hindering system deployment and itinerary diversity. To address this, considering the advanced reasoning and planning capabilities of LLMs, we leverage LLMs to generate final itineraries that satisfy these diverse constraints.

Specifically, the primary objective of this module is to effectively utilize LLM to select an optimal subset from $\mathcal{P}^{\text{order}}$, which closely aligns with user requests while adhering to various constraints. This process can be formally defined as follows:

$$\mathcal{I} \sim \text{LLM}_\theta \left(\mathbb{P}_{IG} \left(\mathbf{r}, \mathcal{P}^{\text{order}}, \mathcal{L} \right) \right), \tag{8}$$

where $\mathcal{L}$ indicates extra natural language input that improves the language quality of the generated itinerary.

The prompt $\mathbb{P}_{IG}$ for generating the final itinerary consists of the following parts: (1) User request information; (2) Candidate POIs and their context; (3) Task description; (4) Specific constraints; (5) Language style guide; (5) Output format description.

Utilizing LLMs to generate the final travel itinerary allows for a balance among various considerations. Additionally, leveraging the reasoning capabilities of LLM, we can create a time-appropriate travel itinerary tailored to the user requests. Specifically, we leverage a simple time indication prompt $\mathbb{P}_{TI}$ to indicate the travel time (in hours) of the itinerary, and the results can be integrated into the "Task Description" part of the prompt $\mathbb{P}_{IG}$.

5 EXPERIMENTS

We conduct extensive experiments to answer the following research questions: **RQ1**, why does directly using LLM to generate travel itineraries not guarantee a good user experience, and what advantages can ItiNera offer in comparison? **RQ2**, how do the various components of ItiNera enhance the quality of the generated itineraries? **RQ3**, how well does ItiNera serve the OUIP service at TuTu in an online production environment? Sec. 5.3 provides results to answer RQ1. Sec. 5.4 provides an ablation study to answer RQ2. Sec. 5.6 provides online performance results to answer RQ3.

5.1 Experimental Setup

5.1.1 Data Description. We conduct experiments on an urban travel itinerary dataset from four Chinese cities, collected in collaboration with a leading travel agency specializing in single-day city tours. The dataset comprises top-rated 1233 urban itineraries and 7578 POIs. Each data sample contains a user request, the corresponding urban itinerary plan, and detailed POI data.

5.1.2 Implementation Details. We utilize two LLMs through their APIs: GPT-3.5 (gpt-3.5-turbo version) and GPT-4 (gpt-4-turbo version). GPT-4 is used for the final itinerary generation ($\mathbb{P}_{IG}$) to guarantee the overall quality of the itinerary, and GPT-3.5 is employed for other LLM interactions to ensure the response speed of our OUIP service. We also incorporate the text-embedding-ada-002 model for embedding purposes. The spatial coherence of itineraries is optimized through an open-source TSP solver[1]. The integration of POI data encompasses various attributes, including geographical coordinates, user ratings, categorizations, and physical addresses, facilitated through the Amap API[2]. Notably, the original languages of our service and data are simplified Chinese, and the translated version is used for demonstration purposes in this paper.

5.2 Evaluation Metrics

According to the concept of the OUIP task, a satisfactory itinerary must possess the following characteristics: spatial coherence and alignment with the user's needs. To this end, we have designed several objective metrics to evaluate the generated itineraries.

5.2.1 Rule-based Metrics. We first design some rule-based metrics: (1) **Recall Rate (RR)**: the recall rate of POIs in the ground truth

[1]https://github.com/fillipe-gsm/python-tsp
[2]https://lbs.amap.com/

Variants	UPC	RD	PPR	CSO	IG	Rule-based Metrics			GPT-based Metrics ↑ (%)		
						RR ↑ (%)	AM ↓ (meters)	OL ↓	PQ	IQ	Match
GPT-4 CoT [31]	×	×	×	×	✓	18.4	258.3	0.49	/	/	/
GPT-4 CoT + UPC	✓	×	✓	×	✓	34.2	240.2	0.52	65.5	61.8	70.6
Ours w/o RD	✓	×	✓	✓	✓	22.6	35.4	0.18	68.2	61.5	60.5
Ours w/o PPR	✓	✓	×	✓	✓	28.2	84.6	0.38	66.7	63.4	62.2
Ours w/o CSO	✓	✓	✓	×	✓	32.8	242.8	1.04	72.1	60.2	74.2
Ours w/ GPT-3.5	✓	✓	✓	✓	✓	27.6	79.4	0.56	67.3	58.8	61.4
Ours (full)	✓	✓	✓	✓	✓	31.4	86.0	0.42	69.8	64.6	72.0

Table 2: Ablation study on Shanghai dataset.

itinerary; (2) **Average Margin (AM)**: The average difference per POI between the total distance of the generated itinerary and the shortest distance (calculated by TSP); (3) **Overlaps (OL)**: the number of self-intersection points in the generated itinerary. (4) **Fail Rate (FR)**: The percentage of POIs generated by LLM that cannot be matched with the queried POIs from the map service (serve as the full POI set). RR evaluates the method's accuracy in understanding user requests and recommending correct POIs. AM and OL measure spatial optimization for POI visit order, where lower is better. FR evaluates LLM's information accuracy and indicates the method's compatibility with the online service. Locations of failed POIs are inaccessible and cannot be displayed on the map, which lowers the user experience.

5.2.2 GPT-based Metrics. The above rule-based metrics are intuitive. Some other aspects are hard to quantify, e.g. the intrinsic appeal of the POIs or the degree of alignment between user requests and the generated itinerary. Therefore, following previous benchmarks for automatic evaluation [1, 12, 29], we propose several GPT-based metrics: (1) **POI Quality (PQ)**: how interesting and diverse the POIs are; (2) **Itinerary Quality (IQ)**: the overall quality and coherence of the itinerary; (3) **Matchness (Match)**: the matchness between the itinerary and the user request; We ask GPT-4 to rank two candidate itineraries generated with different methods and compute the win rate. We repeat the same query for at least 10 times with different seeds to obtain a reliable result.

Automated evaluation through GPT offers a more efficient and cost-effective alternative to human evaluation for large-scale experiments. For example, the cost of GPT tokens for the entire ablation study in Sec. 5.4 is less than $25. In addition, the results of the online subjective evaluation in real-world scenarios in Sec. 5.6.2 demonstrate consistent results between human and GPT assessments.

5.3 Overall Results

5.3.1 Baselines. We compare ItiNera with the following baselines:

- IP [7]: A traditional IP method. We simplified it to use LLM for time budgeting and to consider POI ratings as utilities.
- Ernie-Bot 4.0 [23]: The best-performing model on Chinese LLM tasks, selected as our dataset and service are in Chinese.
- GPT-3.5 [16]: The standard ChatGPT model.
- GPT-4 [17]: The enhanced model offering superior performance and broader knowledge coverage.
- GPT-4 CoT [31]: The model improves GPT-4 with the LLM-as-agent method, Chain-of-Thought reasoning.

We employ the same itinearry generation prompt for all baselines, including basic task requirements and output format as in $\mathbb{P}_{IG}$. Notably, for GPT-4 CoT, we extend the prompt by integrating the "thoughts". Note that the baseline IP and our method does not compute Fail Rate as the candidate POIs are from the dataset.

5.3.2 Results Analysis. We provide analysis for results. As illustrated in Tab. 1, our proposed ItiNera significantly outperforms all baselines across all metrics. For rule-based metrics, ItiNera shows ~30% improvement over the best baseline, highlighting its superior capability in personalizing user experiences. Furthermore, it maintains spatial coherence by generating itineraries that are only marginally longer (~100 meters per planned POI) than the shortest path solved by TSP, a remarkable achievement considering potential user requests that may deviate from spatially optimal routes. Notably, ItiNera is the only method to surpass the performance of GPT-4 CoT in GPT-based evaluations, particularly excelling in matchness. These findings underscore ItiNera's efficacy in enhancing spatial coherence and aligning with user requests in OUIP.

5.4 Ablation Study

To understand how the different components of ItiNera affect the quality of the generated itineraries (**RQ2**), we compare with the following variants of ItiNera:

- GPT-4 CoT + UPC: This variant integrates the UPC module to LLMs to generate itineraries conditioned on user-owned POIs.
- Ours w/o RD: This variant leverages the entire user input string's embedding to retrieve the POIs.
- Ours w/o PPR: This variant quantifies the contribution of the PPR module compared to our full system.
- Ours w/o CSO: This variant removes the CSO module and let the LLM called in the IG module determines the order of candidate POIs when generating the final itineraries.
- Ours w/ GPT-3.5: This variant replace the GPT-4 with GPT-3.5 for generating the final itinerary.

We remove Fail Rate in ablation study, since all variants equipped with UPC never generate POIs not presented in the database.

The results are shown in Tab. 2. Comparing the first two rows, we find UPC can improve Recall Rate and Matchness of the GPT-4 CoT baseline. The variants "w/o RD", "w/o PPR", and "w/ GPT-3.5" exhibit lower Recall Rate, POI Quality and Matchness than our full model. This is because they are relatively inept at understanding user requests and selecting relevant POIs. However, they display lower Average Margin and Overlaps. We attribute this to the idea that there is a trade-off between spatial optimization and matchness.

Method	Rule-based		POI Quality			Itinerary Quality			Matchness		
	AM	OL	Expert	User	GPT	Expert	User	GPT	Expert	User	GPT
GPT-4 CoT	511.4	0.79	3.2	3.6	30%	2.5	3.0	32%	2.9	2.6	28%
Ours	107.6	0.44	3.8	4.3	70%	3.2	3.8	68%	3.6	3.5	72%

Table 3: Online evaluation.

Imposing constraints to align a system with human preference will sacrifice its inherent ability. This is similar to conditional generation tasks in other domains [5, 22] , where aligning output with human input hinders the generation quality and is worth than unconditional generation. This trade-off is also observed when removing CSO module ("w/o CSO") leads to worse Average Margin and Overlaps but better Recall Rate, POI Quality and Matchness. Our full model manages to balance matchness with spatial ability.

In addition to effectiveness of the PPR module as verified by the results of "w/o PPR", the integration of the PPR module could also help us address the issue of the limited sizes of LLMs' context windows, saving costs on LLM for our system. Lastly, we find that the results of the variant "w/ GPT3.5", although compromised, still significantly outperform the GPT-3.5 baseline in Tab. 1, demonstrating our system's adaptability to alternative LLMs for OUIP.

5.5 Qualitative Results

We conduct a qualitative study to demonstrate our system's effectiveness and the importance of integrating LLM with an CSO module. Consider a user request "I'm seeking an artsy itinerary that includes exploring the river's bridges and ferries." We visualize the results from ItiNera and GPT-4 CoT in Fig. 3.

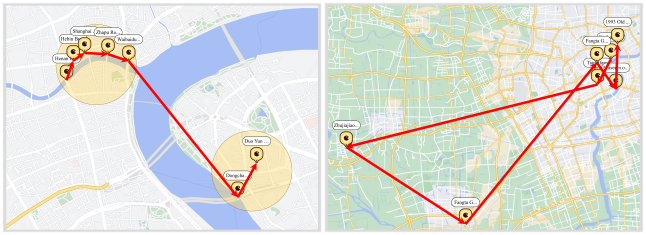

Figure 3: Itineraries of ItiNera (left) and GPT-4 CoT (right).

We find that our itinerary better matches the user preferences. The itinerary passes several bridges along the Huangpu River, include a ferry crossing, and conclude at the art-atmosphere-rich Duoyun Bookstore, offering a restful endpoint for users. In contrast, the POIs selected by GPT are more mainstream. Moreover, our spatial arrangement is more logical, avoiding detours and concentrating selected POIs within two spatial clusters. The itinerary generated by GPT is spatially poor, with a disordered sequence of visits and contains excessively distant POIs. Beyond this example, GPT also risks hallucinating non-existent POIs, highlighting the superiority of our system ItiNera in comparison.

5.6 Online Performances

5.6.1 Online Deployment. Our system has been deployed as the core algorithm of the TuTu online OUIP service. Within a month of operation, it has attracted over 1K registered users who have created more than a thousand itineraries across 60 cities, incorporating over 30K user-owned POIs. In our online production, the itinerary generation utilizes the GPT-4 API with streaming. The average latency for the initial response token is around 6 seconds, and the complete response takes around 20 seconds. Our designed prompt asks for a short output, thereby reducing the overall response time.

5.6.2 Online Evaluation. To verify the effectiveness of our system in real-world scenarios, we conduct online evaluations (**RQ3**). Subjective evaluation via user studies has been extensively employed in prior research on generative tasks [21, 22, 44] where objective metrics fail to adequately assess specific dimensions of the output quality. The TuTu online OUIP service provides a natural way to conduct subjective evaluation. For a small group of users (*User*), we provide them with a feedback interface upon the completion of each itinerary generation. For those users, we randomly replace the algorithm in the service with the baseline GPT-4 CoT to generate itineraries in the probability of 50%. In the same way as the GPT-based metrics, users are required to rate the POI Quality, Itinerary Quality and Matchness, following the Likert scale (1 for poor and 5 for excellent). We gathered 1443 results from 464 users. These ratings are subsequently utilized to upgrade our system to enhance user experience. We also invited 33 experienced travel assistants (*Expert*) from our collaborative travel agency to provide evaluation on the collected 1443 queries and itineraries from a professional perspective. For GPT-based metrics, we first use the other algorithm to generate an itinerary for the same query to compare with the existing itinerary. Then, we ask GPT-4 to rank the two candidate itineraries and compute the win rate over all queries.

The average evaluation results are reported in Tab. 3. One can observe that our method is preferred by experts and online users across all metrics, especially for Matchness, validating the effectiveness of our system in real-world scenarios. Moreover, the online evaluation results are consistent with the GPT evaluation win rate, indicating that the proposed GPT-based metrics are appropriate and adaptable where massive results need to be evaluated but rule-based evaluation is not sufficient.

6 CONCLUSION

We introduce the problem of OUIP and a solution ItiNera that integrates LLMs with spatial optimization, enabling the generation of personalized and spatially coherent itineraries directly from natural language requests. Our system is deployed in the TuTu online service. The results from offline and online experiments validate the effectiveness of our approach. This study not only establishes a new benchmark for itinerary planning technologies but also broadens venues for further innovations in leveraging LLMs for complex problem-solving in urban contexts.

REFERENCES

[1] Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023* (2023).

[2] Paolo Bolzoni, Sven Helmer, Kevin Wellenzohn, Johann Gamper, and Periklis Andritsos. 2014. Efficient itinerary planning with category constraints. In *Proceedings of the 22nd ACM SIGSPATIAL international conference on advances in geographic information systems*. 203–212.

[3] Gang Chen, Sai Wu, Jingbo Zhou, and Anthony KH Tung. 2013. Automatic itinerary planning for traveling services. *IEEE transactions on knowledge and data engineering* 26, 3 (2013), 514–527.

[4] I de Zarzà, J de Curtò, Gemma Roig, and Carlos T Calafate. 2023. LLM multimodal traffic accident forecasting. *Sensors* 23, 22 (2023), 9225.

[5] Shangzhe Di, Zeren Jiang, Si Liu, Zhaokai Wang, Leyan Zhu, Zexin He, Hongming Liu, and Shuicheng Yan. 2021. Video background music generation with controllable music transformer. In *Proceedings of the 29th ACM International Conference on Multimedia*. 2037–2045.

[6] Aristides Gionis, Theodoros Lappas, Konstantinos Pelechrinis, and Evimaria Terzi. 2014. Customized tour recommendations in urban areas. In *Proceedings of the 7th ACM international conference on Web search and data mining*. 313–322.

[7] Aldy Gunawan, Zhi Yuan, and Hoong Chuin Lau. 2014. A mathematical model and metaheuristics for time dependent orienteering problem. PATAT.

[8] Sajal Halder, Kwan Hui Lim, Jeffrey Chan, and Xiuzhen Zhang. 2024. A survey on personalized itinerary recommendation: From optimisation to deep learning. *Applied Soft Computing* 152 (2024), 111200.

[9] Ngai Lam Ho and Kwan Hui Lim. 2022. Poibert: A transformer-based model for the tour recommendation problem. In *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 5925–5933.

[10] Yu-Ling Hsueh and Hong-Min Huang. 2019. Personalized itinerary recommendation with time constraints using GPS datasets. *Knowledge and Information Systems* 60, 1 (2019), 523–544.

[11] Jingqing Jiang, Jingying Gao, Gaoyang Li, Chunguo Wu, and Zhili Pei. 2014. Hierarchical solving method for large scale TSP problems. In *International symposium on neural networks*. Springer, 252–261.

[12] Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. AlpacaEval: An Automatic Evaluator of Instruction-following Models. https://github.com/tatsu-lab/alpaca_eval.

[13] Yuebing Liang, Yichao Liu, Xiaohan Wang, and Zhan Zhao. 2023. Exploring large language models for human mobility prediction under public events. *arXiv preprint arXiv:2311.17351* (2023).

[14] Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, et al. 2023. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*. PMLR, 22137–22176.

[15] Baichuan Mo, Hanyong Xu, Dingyi Zhuang, Ruoyun Ma, Xiaotong Guo, and Jinhua Zhao. 2023. Large Language Models for Travel Behavior Prediction. *arXiv preprint arXiv:2312.00819* (2023).

[16] OpenAI. [n. d.]. Chatgpt. https://openai.com/chatgpt

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

[18] Arka Pal, Deep Karkhanis, Manley Roberts, Samuel Dooley, Arvind Sundararajan, and Siddartha Naidu. 2023. Giraffe: Adventures in expanding context lengths in llms. *arXiv preprint arXiv:2308.10882* (2023).

[19] Septia Rani, Kartika Nur Kholidah, and Sheila Nurul Huda. 2018. A development of travel itinerary planning application using traveling salesman problem and k-means clustering approach. In *Proceedings of the 2018 7th International Conference on Software and Computer Applications*. 327–331.

[20] Septia Rani, Yoga Agung Kurnia, Sheila Nurul Huda, and Sarah Ayu Safitri Ekamas. 2019. Smart travel itinerary planning application using held-karp algorithm and balanced clustering approach. In *Proceedings of the 2019 2nd international conference on E-business, information management and computer science*. 1–5.

[21] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10684–10695.

[22] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems* 35 (2022), 36479–36494.

[23] Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. 2021. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137* (2021).

[24] Kadri Sylejmani, Jürgen Dorn, and Nysret Musliu. 2017. Planning the trip itinerary for tourist groups. *Information Technology & Tourism* 17 (2017), 275–314.

[25] Yihong Tang, Junlin He, and Zhan Zhao. 2022. HGARN: Hierarchical Graph Attention Recurrent Network for Human Mobility Prediction. *arXiv preprint arXiv:2210.07765* (2022).

[26] Kendall Taylor, Kwan Hui Lim, and Jeffrey Chan. 2018. Travel itinerary recommendations with must-see points-of-interest. In *Companion Proceedings of the The Web Conference 2018*. 1198–1205.

[27] Pradeep Varakantham and Akshat Kumar. 2013. Optimization approaches for solving chance constrained stochastic orienteering problems. In *Algorithmic Decision Theory: Third International Conference, ADT 2013, Bruxelles, Belgium, November 12-14, 2013, Proceedings 3*. Springer, 387–398.

[28] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2023. A survey on large language model based autonomous agents. *arXiv preprint arXiv:2308.11432* (2023).

[29] Shenzhi Wang, Chang Liu, Zilong Zheng, Siyuan Qi, Shuo Chen, Qisen Yang, Andrew Zhao, Chaofei Wang, Shiji Song, and Gao Huang. 2023. Avalon's Game of Thoughts: Battle Against Deception through Recursive Contemplation. *arXiv preprint arXiv:2310.01320* (2023).

[30] Xinglei Wang, Meng Fang, Zichao Zeng, and Tao Cheng. 2023. Where would i go next? large language models as human mobility predictors. *arXiv preprint arXiv:2308.15197* (2023).

[31] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems* 35 (2022), 24824–24837.

[32] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2023. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864* (2023).

[33] Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. 2024. TravelPlanner: A Benchmark for Real-World Planning with Language Agents. *arXiv preprint arXiv: 2402.01622* (2024).

[34] Hao Xue and Flora D Salim. 2023. Promptcast: A new prompt-based learning paradigm for time series forecasting. *IEEE Transactions on Knowledge and Data Engineering* (2023).

[35] Hao Xue, Bhanu Prakash Voutharoja, and Flora D Salim. 2022. Leveraging language foundation models for human mobility forecasting. In *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*. 1–9.

[36] Yibo Yan, Haomin Wen, Siru Zhong, Wei Chen, Haodong Chen, Qingsong Wen, Roger Zimmermann, and Yuxuan Liang. 2023. When urban region profiling meets large language models. *arXiv preprint arXiv:2310.18340* (2023).

[37] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601* (2023).

[38] Phatpicha Yochum, Liang Chang, Tianlong Gu, and Manli Zhu. 2020. An adaptive genetic algorithm for personalized itinerary planning. *IEEE Access* 8 (2020), 88147–88157.

[39] Yu Zhang and Jiafu Tang. 2018. Itinerary planning with time budget for risk-averse travelers. *European Journal of Operational Research* 267, 1 (2018), 288–303.

[40] Ou Zheng, Mohamed Abdel-Aty, Dongdong Wang, Chenzhu Wang, and Shengxuan Ding. 2023. Trafficsafetygpt: Tuning a pre-trained large language model to a domain-specific expert in transportation safety. *arXiv preprint arXiv:2307.15311* (2023).

[41] Ou Zheng, Mohamed Abdel-Aty, Dongdong Wang, Zijin Wang, and Shengxuan Ding. 2023. ChatGPT is on the horizon: Could a large language model be all we need for Intelligent Transportation? *arXiv preprint arXiv:2303.05382* (2023).

[42] Ou Zheng, Dongdong Wang, Zijin Wang, and Shengxuan Ding. 2023. Chat-GPT Is on the Horizon: Could a Large Language Model Be Suitable for Intelligent Traffic Safety Research and Applications? *ArXiv* (2023).

[43] Zhilun Zhou, Yuming Lin, and Yong Li. 2024. Large language model empowered participatory urban planning. *arXiv preprint arXiv:2402.01698* (2024).

[44] Le Zhuo, Zhaokai Wang, Baisen Wang, Yue Liao, Chenxi Bao, Stanley Peng, Songhao Han, Aixi Zhang, Fei Fang, and Si Liu. 2023. Video background music generation: Dataset, method and evaluation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 15637–15647.