AgentMove: Predicting Human Mobility Anywhere Using Large Language Model based Agentic Framework

Jie Feng, Yuwei Du, Jie Zhao, Yong Li
Department of Electronic Engineering, Tsinghua University, Beijing, China,
{fengjie, liyong07}@tsinghua.edu.cn

Abstract

Human mobility prediction plays a crucial role in various real-world applications. Although deep learning based models have shown promising results over the past decade, their reliance on extensive private mobility data for training and their inability to perform zero-shot predictions, have hindered further advancements. Recently, attempts have been made to apply large language models (LLMs) to mobility prediction task. However, their performance has been constrained by the absence of a systematic design of workflow. They directly generate the final output using LLMs, which limits the potential of LLMs to uncover complex mobility patterns and underestimates their extensive reserve of global geospatial knowledge. In this paper, we introduce **AgentMove**, a systematic agentic prediction framework to achieve generalized mobility prediction for any cities worldwide. In AgentMove, we first decompose the mobility prediction task into three sub-tasks and then design corresponding modules to complete these subtasks, including spatial-temporal memory for individual mobility pattern mining, world knowledge generator for modeling the effects of urban structure and collective knowledge extractor for capturing the shared patterns among population. Finally, we combine the results of three modules and conduct a reasoning step to generate the final predictions. Extensive experiments on mobility data from two sources in 12 cities demonstrate that AgentMove outperforms the best baseline more than 8% in various metrics and it shows robust predictions with various LLMs as base and also less geographical bias across cities.

Introduction

Mobility prediction is of great importance in many real-world scenarios, e.g., recommend travel services, pre-activating mobile applications for potential usage, seamless switching of cellular network signals and efficient traffic management. In the recent years, deep learning based models (Liu et al. 2016; Wu et al. 2017; Feng et al. 2018; Yang et al. 2020; Yang, Liu, and Zhao 2022) are widely applied and has achieved good results due to its advantages in capturing the high-order transition capabilities and mining the shared mobility patterns between users. However, existing works have several key drawbacks. First, the success of deep learning models rely on the collection of large amounts of private mobility data. Second, the trained model is difficult to be applied in the zero-shot mobility prediction settings. Finally, the prediction accuracy is still lower due to the limited sequential modelling capability of small deep learning models and lack of deeply understanding of common sense in human daily life and urban structures.

Recently, large language models (LLMs) have made significant progress, achieving advanced results far surpassing previous methods in areas such as dialogue based role-playing, code generation and testing, and mathematical problem solving. In the field of spatial temporal data mining, researchers also explore the potential of applying LLMs many real-world tasks, such as time series forecasting (Gruver et al. 2024; Li et al. 2024b), travel planning (Xie et al. 2024; Li et al. 2024a), trajectory analysis (Luo et al. 2024; Zhang et al. 2023), and so on. Furthermore, several recent works (Wang et al. 2023; Beneduce, Lepri, and Luca 2024) investigate the feasibility of employing the LLMs as the base model of mobility predictor for addressing the previous limitations of deep learning based models and achieve promising results. These works first convert the trajectory to a language based sentence and then utilize the powerful sequential modelling capacities of LLMs to directly output the mobility prediction results. However, due to the absence of a systematic design of whole procedure, they ignore the crucial components of human mobility modeling and thus achieve limited performance. In summary, these methods fail to effectively capture the complicated individual mobility pattern, ignore modeling the effects of urban structure and discovering the shared mobility patterns among population.

In this paper, we propose AgentMove, a systematic agentic framework for generalized mobility prediction in any cities around the world. In AgentMove, by integrating the domain knowledge of human mobility, we implement the core components in the general agentic framework (Wang et al. 2024; Xi et al. 2023), including the planning module, memory module, world knowledge module, external tool module and reasoning module. For the planning module in AgentMove, we introduce a manually designed mobility prediction task decomposition module by considering the most important factors for mobility prediction. After decomposition, we generate three sub-tasks: individual mobility pattern mining, shared mobility pattern discovering and urban structure modelling. Firstly, we implement a spatial-temporal memory module for individual mobility pattern mining. It contains three submodules, short-term

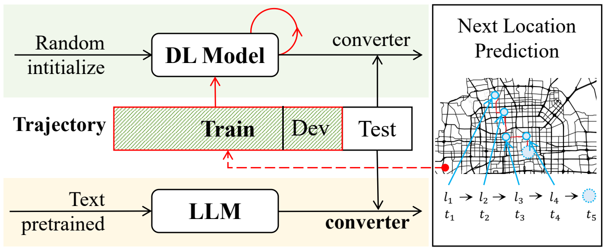


Figure 1: Different paradigms of deep learning model based mobility predictor and LLM based mobility predictor

memory, long-term memory and user profiles, to capture the multi-level mobility patterns of individual. Compared with the pure LLM methods, the memory module makes AgentMove to memorize the past mobility history and continuously learn from the experiences efficiently. Secondly, we design a world knowledge generator to explicitly extract the inherent geospatial knowledge of the urban structure from LLMs to help modelling the effects of multi-scale urban structure on the human mobility especially the exploration behavior of human mobility. Thirdly, we endow AgentMove with the ability of discovering the shared mobility patterns from various trajectories of different users by a collective knowledge extractor. It utilizes the NetworkX as a external tool to organize the trajectories from various users to form a location transition graph and then extract important neighbor locations for prediction. Finally, we combine all the results from all the modules modules and perform a final reasoning step to generate the prediction. In summary, our contributions can be summarized as follows,

- To our best knowledge, this is the first attempt to apply LLM-based agentic framework into the field of mobility prediction. We build an effective mobility prediction framework by considering the crucial characteristics of human mobility into the designs of core components.
- We design a mobility prediction task decomposition module and integrate the modelling of different aspects of human mobility into different modules in AgentMove. Specifically, it contains a spatial-temporal memory module for individual mobility pattern mining, a world knowledge generator for modeling effects of urban structures and a graph based collective knowledge extractor for discovering the shared mobility patterns among population.
- Extensive experiments on mobility trajectories from 2 sources in 12 cities demonstrate the effectiveness of proposed AgentMove, which outperforms the best baseline with a performance increase of over 8% in the majority of cases. Additionally, AgentMove presents superior adaptability to different LLMs, as well as greater stability and reduced bias in prediction results across various cities worldwide.

Preliminaries

Here, we define the mobility prediction task and related concepts, which are used in the following section.

Definition 1 (POI) *A POI point $p \in P$ is represented as a tuple $\langle id, cate, lon, lat, addr\rangle$, where id is the unique identifier, $cate$ is the category (e.g., restaurant), lon and lat are the coordinates of the POI, $addr$ is the text address of POI.*

Definition 2 (User Trajectory) *A trajectory of user $u \in U$ is represented as $T_u = \{(p_1, t_1), (p_2, t_2), \ldots, (p_n, t_n)\}$, where $p_i \in P$ is the i-th POI visited by the user and t_i is the timestamp of the visit.*

Definition 3 (Contextual Stays) *Contextual stays of user u is defined as the most recent sub-sequence in trajectory: $\mathcal{C}_u = \{(p_{n-k}, t_{n-k}), \ldots, (p_{n-1}, t_{n-1}), (p_n, t_n)\}$, which captures the user's short-term mobility patterns. k is the window size of contextual stays.*

Definition 4 (Historical Stays) *Historical stays of user u is defined as the sub-sequence before contextual stays: $\mathcal{H}_u = \{(p_1, t_1), (p_2, t_2), \ldots, (p_{n-k-1}, t_{n-k-1})\}$, which captures the user's long-term mobility patterns.*

Given the historical movement data $\mathcal{C}_u$, $\mathcal{H}_u$ as well as available external knowledge $\mathcal{K}$ (e.g., worldwide geospatial information), the objective is to predict the next POI p_{n+1} that user u will visit. Formally, this paper aims to learn a mapping function f:

$$f : (\mathcal{C}_u, \mathcal{H}_u, \mathcal{K}) \rightarrow p_{n+1}. \tag{1}$$

Figure 1 illustrates the differences between the deep learning based paradigm and the LLM based paradigm in the mobility prediction task. The deep learning model needs collecting training data before conduct the prediction task, which means it cannot directly used in the zero-shot scenario. LLM based method can directly applied into any scenario after carefully 'format converter' (known as prompt engineering).

Methods

Overview

As shown in Figure 2, AgentMove consists of five core components: task decomposition module, spatial-temporal memory module, world knowledge generator, collective knowledge extractor and the final reasoning module. As the high-level planning module, the task decomposition module is designed to break down the whole mobility prediction task into subtasks—personalized mobility pattern mining, collective mobility pattern discovery and the effects of urban structures—by considering the crucial factors of mobility. Detailed design of other components are introduced as follows.

Spatial-temporal Memory for Personalized Multi-scale Periodicity Behavior Modelling

The spatial-temporal memory module is designed to effectively capture, store and leverage mobility patterns, providing crucial insights for the personalized and multi-scale periodicity behavior modelling in mobility prediction. Inspired by the memory design principles in general LLM-based agents (Zhang et al. 2024), our spatial-temporal memory functions through three essential processes: memory organization, memory writing, and memory reading. The whole framework of spatial–temporal memory module is presented in Figure 3.

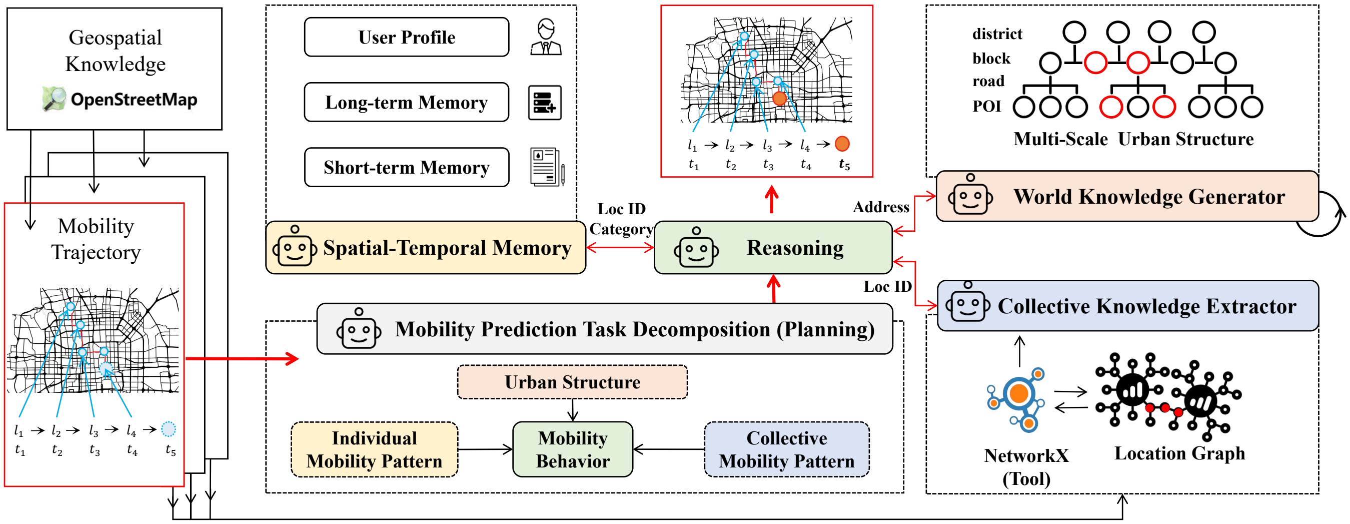


Figure 2: The framework of AgentMove, including spatial temporal memory unit for multi-scale mobility pattern of single user, spatial world model for multi-level saptial structure of nearby locations and social world model for extracting shared mobility patterns of user group.

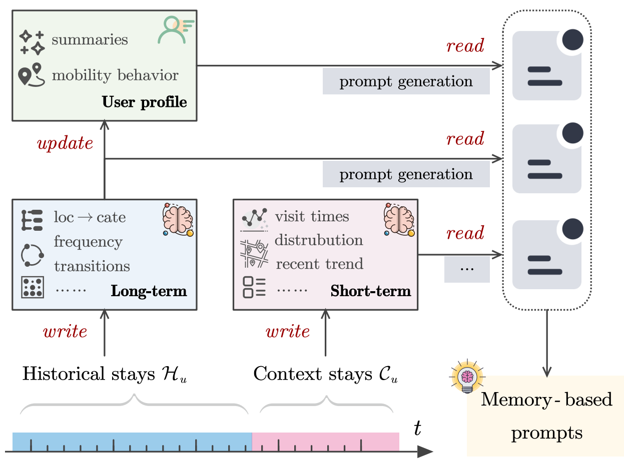


Figure 3: Illustration of Spatial-temporal Memory.

Memory Organization The spatial-temporal memory is structured into three components to capture multifaceted nature of user mobility patterns:

- *User Profile Unit*. This unit provides a summary description of the user's mobility behavior as the user mobile profile, which offers deeper insights into when and why the user visits certain locations. The user profile is dynamically generated based on the current long-term memory introduced in the following part, allowing AgentMove to adapt to the evolving user preferences.
- *Long-term Memory Unit*. This unit retains users' long-term mobility patterns, capturing overarching trends and recurring sequences in their movement history. It functions similarly to how LLMs store long-term dependencies in textual data.
- *Short-term Memory Unit*. This unit focuses on users' recent mobility patterns, providing dynamic updates that reflect the latest movements and short-term variations.

All users' memories are stored in a central memory pool, organized as key-value pairs. Each key corresponds to a unique user identifier, and the value consists of the long-term memory, short-term memory, and user profile info. This organization ensures a comprehensive extraction and storage of mobility data, enabling efficient retrieval and utilization for mobility prediction.

Memory Writing Writing to the memory involves the extraction and structured storage of spatial-temporal patterns hidden in user's trajectories. This process consists of two main steps:

Long-term Memory Writing. Given the historical stays $\mathcal{H}_u$, this module extracts long-term spatial-temporal information of user $u \in U$, including: 1) *location to category mapping*. Associating visited locations with their respective categories. 2) *top-k active times and locations*. Identifying the most active time periods and the most frequently visited locations. 3) *location visit frequency*. Recording how often various locations are visited. 4) *transition matrix*. A matrix that represents the transition probabilities between locations.

Short-term Memory Writing. Given the contextual stay $\mathcal{C}_u$, this module extracts fine-grained short-term spatial-temporal information of user $u \in U$, including: 1) *time sequence of recent visits*. Documenting the sequence of recent visit times. 2) *visit frequency of different locations*. Tracking how frequently different locations are visited in the short term. 3) *details of the last visit*. Recording specific details about the latest location visit.

By systematically organizing and storing this information by processing the trajectories, AgentMove can easily access to both long-term and short-term mobility patterns. This structured approach is crucial for enhancing the accuracy of next location predictions.

Memory Reading The memory reading process involves generating spatial-temporal context relevant prompts from the structured memory to to enhance AgentMove's predictive capabilities. This process consists of three key steps:

User Profile Prompt Generation. Utilizing the long-term memory, AgentMove constructs user profile prompts that encapsulate the intrinsic movement patterns and habitual behaviors of the user. These prompts include summaries of peak activity times, preferred locations, and temporal-spatial associations, providing a comprehensive mobility profile of user.

Long-term Memory Prompt Generation. Also based on the long-term memory, AgentMove generates prompts by summarizing the user's general mobility trends from the long-term view. These prompts include details on the most active times, frequently visited locations, and the relationships between these factors. This helps the LLM understand the user's regular movement patterns.

Short-term Memory Prompt Generation. AgentMove creates prompts from the short-term memory to reflect recent mobility patterns and contextual information of user. These prompts cover recent visit sequences, current visit frequencies, and specifics of the latest visits, which ensure LLMs efficiently adapt to recent changes in user's behavior.

Finally, these memory-based prompts are consolidated into a cohesive spatial-temporal summary of the original trajectory, which is then integrated as the first part of AgentMove's prompts. This spatial-temporal summary enhances the LLM's ability to engage in more logical and efficient reasoning, leading to more precise mobility predictions.

World Knowledge Generator for Enhancing Exploration Behavior Modelling

Lots of research (Jiang et al. 2016) shows that individual movement typically includes two types of behaviors: return and explore. As introduced before, the return behavior has been well-captured by the spatial-temporal memory module. In this section, we introduce the world knowledge generator to extract geospatial knowledge from LLMs and construct multi-scale urban structure to enable the modelling of explore behavior of mobility. To extract geospatial knowledge effectively, we propose to alignment the knowledge of LLMs and the urban space of trajectory via text address. Once the space is alignment, we explicitly motivate the LLMs to generate the potential explore candidate places from the multi-scale urban structure view.

Alignment via Address Many existing works (Feng et al. 2018; Luo, Liu, and Liu 2021; Lin et al. 2021; Cui et al. 2021; Qin et al. 2022; Hong et al. 2023) on mobility prediction usually represent the locations directly using latitude and longitude coordinates or discrete spatial area IDs. While this approach facilitates the easy construction of deep learning-based spatial encoding, it is not suitable for LLMs. Since LLMs are trained on large scale human-generated text, they, like human, are not inherently adept at understanding the precise coordinates (Gurnee and Tegmark 2023) or discrete area IDs. Thus, we propose to utilize the text address which human is familiar to describe the coarse location of trajectory. While text address is not precise as the coordinates, it is more natural and easy to be aligned with the existing spatial knowledge in the LLMs.

Thus, we adapt the address searching service [1] from Open Street Map to build address for each point in the trajectory. Due to cultural and institutional differences, address information formats vary greatly across different countries. To address this, we leverage the common-sense knowledge of LLMs to extract unified structured address information from the original address information. LLMs can easily pinpoint a user's current and past locations, laying a solid foundation for subsequent modeling.

Multi-scale Urban Structure Based on the real structured address information, we design prompts to motivate LLMs to generate multi-scale potential places which may be explored by user in the future. We introduce multi-scale generation mechanism to help LLMs reduce hallucination and improve the accuracy and usability of generate places. The multi-scale location information covers four level: district, block, street and POI name. We first ask LLMs to generate the potential districts in the future. Then, based on these districts and the past blocks in the trajectory to generate the potential blocks in the future and so on. Finally, we can generate potential location information from different levels as the potential exploration candidate for the user.

Collective Knowledge Extractor for Shared Mobility Pattern Modelling

In the previous two sections, we introduce the spatial-temporal memory module and world knowledge generator for the individual level mobility modelling. Here, we introduce the collective knowledge extractor to enable capture the sharing mobility pattern among users to further improve the mobility prediction. In one world, we first construct a global location transition graph using NetworkX [2] by aggregating the location transitions from various users. We then employ a LLM to perform simple reasoning on the graph, utilizing the function in NetworkX as tool to generate potential locations visited by other users with similar mobility pattern.

Building Location Transition Graph In the location graph, the node is location ID with various attributes, e.g., address information, function of location. The edge between nodes is constructed by considering the 1-hop transition between nearby trajectory points in each trajectory. The edge is weighted without direction. Based on the definition of graph, we use NetWorkX to build the graph from scratch and update it when infer trajectories for various users. If any history trajectory data, e.g. training data used by the deep learning based models, are available, the location graph can be initialized by them.

Reasoning on Graph via Tool After obtaining the location graph, we can utilize LLM to perform reasoning on the whole graph via the function of NetworkX as tool. The most naive strategy is to query the k-hop neighbors of the current location. When the number of the neighbors is too much, LLMs need to filter the most promising ones from them by

[1] https://nominatim.org/

[2] https://networkx.org/documentation/stable/index.html

considering the attributes of each node. Furthermore, we can extend the query nodes into the last n locations and generate the most promising ones from all the neighbors of them. In this way, we obtain the most relevant locations that has been visited by the users with similar mobility patterns.

Finally, we design prompts to employ LLM to analyze and summarize the information from different views and perform a final reasoning step to generate the prediction with reasons. The prompts for output format requirements are also be placed here to ensure that the output format meets the requirements as much as possible.

Evaluation

Settings

Datasets We use the global Foursquare checkin data (Yang, Zhang, and Qu 2016) and recent public released ISP trajectory data (Feng et al. 2019) to conduct the experiments. The Foursquare data contains checkins from 415 cities which covers about 18 months from April 2012 to September 2013. The ISP trajectory data is from Shanghai with 325215 records, covering April 19 to April 26 in 2016. Compared with Foursquare data, ISP data is much denser and was open-sourced only two months ago [3], which is beyond the training period of all the LLMs used in the experiment. This ensures that the evaluation results are not affected by potential data leakage.

To evaluate the general mobility prediction ability of AgentMove, we select 12 cities from the Foursquare dataset and the entire ISP trajectory data to conduct the experiments. We follow the preprocessing procedure (Hong et al. 2023; Feng et al. 2019) to process the trajectories data. Detailed description about preprocessing can refer to the appendix. We select the Tokyo, Moscow and SaoPaulo with the largest amount of Foursquare check-in data and the ISP data from Shanghai to conduct the main analysis in the experiments and results of 12 cities are discussed in the final section of experiment. Due to the cost of the various API calling, e.g., Llama3.1-405B, we randomly sample 200 instances from the testing set for each city to calculate the performance in the experiments.

Baselines We have three kinds of baselines, Markov methods, deep learning models and LLM-based methods.

- **FPMC** (Rendle, Freudenthaler, and Schmidt-Thieme 2010) It combines the matrix factorization and Markov chains methods together for sequential modelling.
- **RNN** (Feng et al. 2018) It is a simple RNN based mobility prediction model as regarding the mobility sequence as general sequence.
- **DeepMove** (Feng et al. 2018) It contains a LSTM for capturing the short-term sequential transition and an attention unit for extracting long-term periodical patterns.
- **LSTPM** (Sun et al. 2020) It consists of a non-local network for long-term preference modeling and a geo-dilated RNN for short-term preference learning.
- **LLM-Mob** (Wang et al. 2023) It is the first work to apply LLM (GPT-3.5) to predict the next location with carefully prompt engineering.
- **LLM-ZS** (Beneduce, Lepri, and Luca 2024) Following LLM-Mob, it simplifies the prompts and testifies more LLMs in zero-shot mobility prediction task.

We use widely used Accuracy@1, Accuracy@5, and NDCG@5 as the main evaluation metrics (Sun et al. 2020; Luca et al. 2021) in the experiments.

Implementation We use LibCity (Jiang et al. 2023) to implement the FPMC, RNN, DeepMove and LSTPM. We follow the default parameter settings of these models in the library for training and inference. For LLMs, we use OpenAI API [4] for accessing GPT4omini and DeepInfra [5] for accessing other open source LLMs, including Llama3, Qwen2, Gemma2 and GLM4. Detailed parameter settings for those baselines can be found in the appendix.

Main Results

In this section, we compare AgentMove with 6 baselines in 4 cities to demonstrate the effectiveness of proposed framework. Here, we use GPT4omini as the default base LLM of AgentMove and the results of AgentMove with other kinds of base LLMs can be found in first section in appendix. To ensure a fair comparison, the LLM-based baselines, LLM-Mob and LLM-ZS, also employ the same base LLM as AgentMove. The results are presented in Table 1.

As the representative deep learning models, DeepMove and LSTPM achieve best or second-best results in 5 out of 12 metrics, and both of their performance varies among different cities. Compared with the deep learning baselines, the best LLM-based baseline LLM-ZS can achieve better results than DeepMove and LSTPM in 7 out of 12 metrics. The promising results of LLM-ZS present the powerful sequential pattern discovery and reasoning ability of LLM in modeling mobility, which demonstrate the potential of the LLM-based method. It is noted that the results of LLM-based methods are zero-shot prediction while the deep learning based methods rely on sufficient training with enough mobility data. Compared with these baselines, our proposed method AgentMove is the best method and achieves the best results in 11 out of 12 metrics in 4 datasets. Compared with the deep learning methods, AgentMove outperforms them more than 20% in half of the metrics. Even in the weakest results, it performs very close to the strongest baseline DeepMove with less than 3% performance drop. Compared with the advanced LLM-based baselines, AgentMove outperforms both of them more than 5.7%-39.4% in all the metrics. These results demonstrate the effectiveness of proposed framework in stimulating the comprehensive ability of LLM-base agentic framework for mobility prediction.

Ablation Study

In this section, we conduct ablation study to demonstrate the contribution of each component in AgentMove. Related re-

[3]https://github.com/vonfeng/DPLink/tree/master/data

[4]https://platform.openai.com/

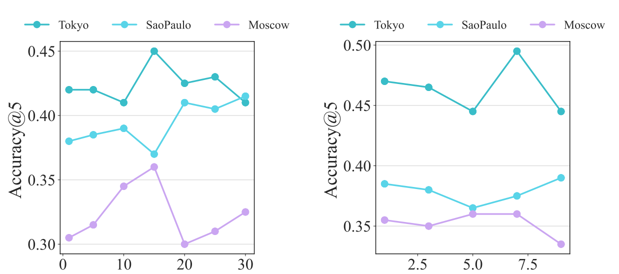
[5]https://deepinfra.com/

Model	FSQ@Tokyo			FSQ@SaoPaulo			FSQ@Moscow			ISP@Shanghai		
	Acc@1	Acc@5	NDCG@5	Acc@1	Acc@5	NDCG@5	Acc@1	Acc@5	NDCG@5	Acc@1	Acc@5	NDCG@5
FPMC	0.060	0.165	0.121	0.045	0.085	0.066	0.020	0.065	0.043	0.13	0.355	0.249
RNN	0.105	0.240	0.176	0.095	0.230	0.169	0.090	0.185	0.140	0.065	0.175	0.123
DeepMove	0.175	0.320	0.251	0.150	0.310	0.236	**0.165**	0.335	<u>0.258</u>	<u>0.175</u>	0.320	0.251
LSTPM	0.145	0.280	0.218	<u>0.190</u>	0.365	<u>0.281</u>	0.140	0.255	0.196	0.095	0.17	0.135
LLM-Mob	0.175	0.370	0.277	0.140	0.275	0.210	0.080	0.175	0.129	0.100	0.345	0.221
LLM-ZS	<u>0.175</u>	<u>0.410</u>	<u>0.299</u>	0.165	<u>0.385</u>	0.277	0.120	<u>0.340</u>	0.233	0.170	<u>0.425</u>	<u>0.298</u>
AgentMove	**0.185**	**0.465**	**0.331**	**0.230**	**0.415**	**0.326**	<u>0.160</u>	**0.365**	**0.265**	**0.190**	**0.450**	**0.329**
vs Deep Learning	5.71%	45.31%	31.87%	21.05%	13.70%	16.01%	-3.03%	8.96%	2.71%	8.57%	26.76%	31.08%
vs Best Baselines	5.71%	13.41%	10.70%	21.05%	7.79%	16.01%	-3.03%	7.35%	2.71%	8.57%	5.88%	10.40%

Table 1: The main results of baselines and AgentMove. GPT4omini is used as the base LLM for all the LLM-based methods in the table. FPMC and deep learning based models are first trained on the training set of each city and LLM-based models are directly evaluated on the test set with the zero-shot prediction settings.

sults are presented in Table 2 and Figure 4. Similar to the last section, we use GPT4omini as the base LLM of AgentMove.

We first discuss the impact of three core components individually, as detailed in the top four lines of Table 2. We have two important observations: 1) each core component not always works well for all the datasets, they may hurt the performance in some cases. For example, after adding spatial temporal memory design in the base prompt, the performance of AgentMove in Tokyo is decreased significantly, but it performs much better in Moscow, 2) the performance gain of each component in different metrics are also different. For example, while memory design leads to better performance in the Acc@1 in SaoPaulo, the performance in other three metrics are dropped. Then we discuss the effects of the combination of the core components in the last three lines in Table 2. We find that while memory or world model generator can not improve the prediction performance individually, their combination introduces significant performance gain in all the metrics of Tokyo dataset. After adding the collective knowledge extractor, the performance gain becomes larger in most of the metrics of Tokyo and SaoPaulo. In summary, compared with the base prompt design, the combination of proposed designs introduce 7%-45% performance gain in all the datasets.



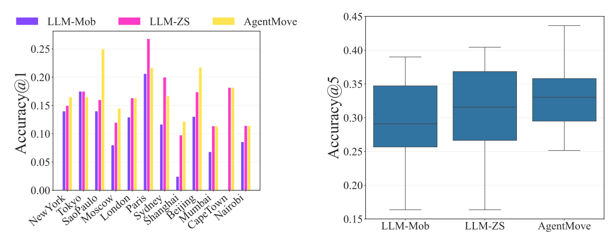
(a) The length of the input history of memory unit.

(b) The number of generated places in spatial world.

Figure 4: Effects of two key parameters in AgentMove.

In Figure 4, we further discuss the effects of several important parameters in the core component of AgentMove, including the history length in memory unit, the number of potential locations generated in the word knowledge generator. When studying the effects of one parameter, we keep all the left parameters fixed. As Figure 4(a) shows, the performance of AgentMove generally improves as the length of historical input in the memory unit increase. Figure 4(b) presents the effects of the number of potential locations based on the associations with urban spatial structures, which is varying a lot among different cities.

Geographical Bias Analysis



(a) Acc@1 of three LLM-based methods on 12 cities.

(b) Distribution of Acc@5 across 3 methods on 12 cities.

Figure 5: Geospatial bias analysis of 3 LLM-based methods.

While LLMs are trained with the online web text which can be geographically bias (Manvi et al. 2024) around the world. We investigate the potential geographical bias in LLM based mobility prediction methods and attempt to answer whether AgentMove can alleviate the geographical bias inherent in LLMs to some extent. Experiment results conducted on 12 cities are presented in Figure 5.

In Figure 5(a), we can find significant differences in the accuracy of three LMM-based methods across cities. For instance, cities like Tokyo, Paris, and Sydney generally achieve higher accuracy, while cities like Cape Town and Nairobi see notably lower performance. This suggests the presence of geographical bias in trained LLMs. We also find that proposed AgentMove performs best in most of the

Models	FSQ@Tokyo			FSQ@SaoPaulo			FSQ@Moscow			ISP@Shanghai		
	Acc@1	Acc@5	NDCG@5	Acc@1	Acc@5	NDCG@5	Acc@1	Acc@5	NDCG@5	Acc@1	Acc@5	NDCG@5
base	0.175	0.410	0.299	0.165	0.385	0.277	0.120	0.340	0.233	0.170	0.425	0.298
+memory	0.145	0.395	0.274	0.190	0.315	0.255	**0.160**	**0.365**	**0.265**	0.170	0.445	0.312
+world	0.150	0.395	0.280	0.175	0.365	0.269	0.090	0.335	0.213	0.155	0.390	0.276
+collective	0.150	0.435	0.298	0.175	0.380	0.275	0.140	0.340	0.246	0.175	**0.465**	0.317
+mem+world	**0.195**	0.455	0.328	**0.240**	0.390	0.317	0.135	0.365	0.254	**0.215**	0.455	**0.342**
+mem+world+col	0.185	**0.465**	**0.331**	0.230	**0.415**	**0.326**	0.150	0.360	0.259	0.190	0.450	0.329
vs base	11.43%	13.41%	10.98%	45.45%	7.79%	17.99%	33.33%	7.35%	13.91%	11.76%	5.88%	10.30%

Table 2: Ablation study results of AgentMove. 'base' denotes the basic prompts which is similar to the baseline LLM-ZS, '+memory' or '+mem' denotes adding spatial-temporal memory, '+world' denotes adding world knowledge generator, '+collective' or '+col' denotes adding collective knowledge extractor.

cities. Figure 5(b) provides a box-plot test by comparing the Acc@5 of the three LLM-based methods in 12 cities. Results demonstrate that AgentMove not only outperforms the other methods in terms of overall accuracy but also exhibits a smaller range of error. The performance of AgentMove is more consistent across different cities, suggesting a reduced impact of geographical bias.

Related Work

Mobility Prediction with Deep Learning

Significant efforts have been made in mobility prediction using deep learning models, encompassing research from both *sequential-based methods* and *graph-based methods*. While traditional methods typically employ Markov models (Rendle, Freudenthaler, and Schmidt-Thieme 2010; Cheng et al. 2013) to predict the next visit by learning the transition probabilities between consecutive POIs, sequential-based deep learning methods are first proposed to model the high-order movement patterns of trajectory data. It contains two kinds of works, Recurrent Neural Networks (RNNs) (Kong and Wu 2018; Huang et al. 2019; Yang et al. 2020; Zhao et al. 2020), and attention mechanisms(Feng et al. 2018; Luo, Liu, and Liu 2021; Lin et al. 2021; Cui et al. 2021; Qin et al. 2022; Hong et al. 2023) based works. Despite their success, these methods primary focus on extracting mobility patterns from an individual perspective, while overlooking the collaborative information available from other users' trajectories. To this end, recent works (Rao et al. 2022; Yang, Liu, and Zhao 2022) have explored the use of Graph Neural Networks (GNNs) for their ability to model complex relationships. However, all these methods rely on collecting large volume of private trajectory. Our proposed AgentMove leverages the world knowledge and sequential modeling abilities of LLM to enable the generalized mobility prediction with zero-shot prediction ability.

Large Language Models and Agents

Due to the powerful language based generalization and reasoning capabilities (Wei et al. 2022a), large language models (OpenAI 2022; Touvron et al. 2023) develop rapidly and have been widely applied in various tasks, e.g., coding (Qian et al. 2024) and math (Wei et al. 2022a). Recent works (Gurnee and Tegmark 2023; Manvi et al. 2023) find that LLMs have learned lots of geographical knowledge of the world. Besides, researcher also explore the potential of applying LLMs in spatial temporal data modelling by directly transferring the domain specific task into the language format, e.g., time series forecasting (Gruver et al. 2024), traffic prediction (Li et al. 2024b), trajectory mining (Wang et al. 2023; Beneduce, Lepri, and Luca 2024), trip recommendation (Xie et al. 2024; Li et al. 2024a) and so on. These early works present the potential of applying LLMs in spatial temporal modelling. To effectively utilize the vast knowledge hidden in the LLMs and stimulate their reasoning and planning abilities, lots of prompts techniques (Wei et al. 2022b; Kojima et al. 2022; Wang et al. 2022; Yao et al. 2024) are proposed in solving naive text games and math problem. However, when it comes to the complicated task in the real life and domain-specific tasks, only the prompt techniques are not enough. Recently, LLM based agents (Wang et al. 2024; Xi et al. 2023) are proposed to solve this question by equipping LLMs with explicit memory, working flow and external tools. Here, we are the first to design LLM based agent for mobility prediction task. With explicit spatial temporal memory and work flow of spatial structure and social structure mining, we succeed in leveraging the world knowledge of LLM and structured reasoning capability for spatial temporal trajectory modelling.

Conclusion

In this paper, we propose AgentMove, a systematic agentic framework for evolvable and generalized mobility prediction around the world. In AgentMove, we first decompose the mobility prediction task into three sub-tasks. To solve these sub-tasks, we design spatial-temporal memory and collective knowledge extractor to learn the individual mobility pattern and shared mobility pattern among users. Furthermore, we design a world knowledge generator to make use of the text address to understand the urban structures like human and thus enable the effective exploration modelling of human mobility. Extensive experiments on mobility trajectories from 12 cities demonstrate the superiority and robustness of AgentMove. In the future, we plan to further explore the better ways to extract and utilize the vast world knowledge and common sense of LLM for mobility modelling. Besides, we will explore the framework in other mobility trajectory related tasks, e.g., trajectory generation.

References

Beneduce, C.; Lepri, B.; and Luca, M. 2024. Large Language Models are Zero-Shot Next Location Predictors. *arXiv preprint arXiv:2405.20962*.

Cheng, C.; Yang, H.; Lyu, M. R.; and King, I. 2013. Where you like to go next: Successive point-of-interest recommendation. In *Twenty-Third international joint conference on Artificial Intelligence*.

Cui, Q.; Zhang, C.; Zhang, Y.; Wang, J.; and Cai, M. 2021. ST-PIL: Spatial-temporal periodic interest learning for next point-of-interest recommendation. In *Proceedings of the 30th ACM International conference on information & knowledge management*, 2960–2964.

Feng, J.; Li, Y.; Zhang, C.; Sun, F.; Meng, F.; Guo, A.; and Jin, D. 2018. DeepMove: Predicting Human Mobility with Attentional Recurrent Networks. In *Proceedings of the 2018 World Wide Web Conference*.

Feng, J.; Zhang, M.; Wang, H.; Yang, Z.; Zhang, C.; Li, Y.; and Jin, D. 2019. DPLink: User Identity Linkage via Deep Neural Network From Heterogeneous Mobility Data. In *The World Wide Web Conference*, 459–469. ACM.

Gruver, N.; Finzi, M.; Qiu, S.; and Wilson, A. G. 2024. Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems*, 36.

Gurnee, W.; and Tegmark, M. 2023. Language models represent space and time. *arXiv preprint arXiv:2310.02207*.

Hong, Y.; Zhang, Y.; Schindler, K.; and Raubal, M. 2023. Context-aware multi-head self-attentional neural network model for next location prediction. *Transportation Research Part C: Emerging Technologies*, 156: 104315.

Huang, L.; Ma, Y.; Wang, S.; and Liu, Y. 2019. An attention-based spatiotemporal lstm network for next poi recommendation. *IEEE Transactions on Services Computing*, 14(6): 1585–1597.

Jiang, J.; Han, C.; Jiang, W.; Zhao, W. X.; and Wang, J. 2023. LibCity: A Unified Library Towards Efficient and Comprehensive Urban Spatial-Temporal Prediction. *arXiv preprint arXiv:2304.14343*.

Jiang, S.; Yang, Y.; Gupta, S.; Veneziano, D.; Athavale, S.; and González, M. C. 2016. The TimeGeo modeling framework for urban mobility without travel surveys. *Proceedings of the National Academy of Sciences*, 113(37): E5370–E5378.

Kojima, T.; Gu, S. S.; Reid, M.; Matsuo, Y.; and Iwasawa, Y. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213.

Kong, D.; and Wu, F. 2018. HST-LSTM: A hierarchical spatial-temporal long-short term memory network for location prediction. In *Ijcai*, volume 18, 2341–2347.

Li, P.; de Rijke, M.; Xue, H.; Ao, S.; Song, Y.; and Salim, F. D. 2024a. Large language models for next point-of-interest recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1463–1472.

Li, Z.; Xia, L.; Tang, J.; Xu, Y.; Shi, L.; Xia, L.; Yin, D.; and Huang, C. 2024b. Urbangpt: Spatio-temporal large language models. *arXiv preprint arXiv:2403.00813*.

Lin, Y.; Wan, H.; Guo, S.; and Lin, Y. 2021. Pre-training context and time aware location embeddings from spatial-temporal trajectories for user next location prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 4241–4248.

Liu, Q.; Wu, S.; Wang, L.; and Tan, T. 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. In *AAAI*.

Luca, M.; Barlacchi, G.; Lepri, B.; and Pappalardo, L. 2021. A survey on deep learning for human mobility. *ACM Computing Surveys (CSUR)*, 55(1): 1–44.

Luo, Y.; Cao, Z.; Jin, X.; Liu, K.; and Yin, L. 2024. Deciphering Human Mobility: Inferring Semantics of Trajectories with Large Language Models. *arXiv preprint arXiv:2405.19850*.

Luo, Y.; Liu, Q.; and Liu, Z. 2021. Stan: Spatio-temporal attention network for next location recommendation. In *Proceedings of the web conference 2021*, 2177–2185.

Manvi, R.; Khanna, S.; Burke, M.; Lobell, D.; and Ermon, S. 2024. Large language models are geographically biased. *arXiv preprint arXiv:2402.02680*.

Manvi, R.; Khanna, S.; Mai, G.; Burke, M.; Lobell, D.; and Ermon, S. 2023. Geollm: Extracting geospatial knowledge from large language models. *arXiv preprint arXiv:2310.06213*.

OpenAI. 2022. Introducing ChatGPT. https://openai.com/blog/chatgpt/.

Qian, C.; Xie, Z.; Wang, Y.; Liu, W.; Dang, Y.; Du, Z.; Chen, W.; Yang, C.; Liu, Z.; and Sun, M. 2024. Scaling Large-Language-Model-based Multi-Agent Collaboration. *arXiv preprint arXiv:2406.07155*.

Qin, Y.; Fang, Y.; Luo, H.; Zhao, F.; and Wang, C. 2022. Next point-of-interest recommendation with auto-correlation enhanced multi-modal transformer network. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2612–2616.

Rao, X.; Chen, L.; Liu, Y.; Shang, S.; Yao, B.; and Han, P. 2022. Graph-flashback network for next location recommendation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1463–1471.

Rendle, S.; Freudenthaler, C.; and Schmidt-Thieme, L. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, 811–820.

Sun, K.; Qian, T.; Chen, T.; Liang, Y.; Nguyen, Q. V. H.; and Yin, H. 2020. Where to go next: Modeling long-and short-term user preferences for point-of-interest recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 214–221.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wang, L.; Ma, C.; Feng, X.; Zhang, Z.; Yang, H.; Zhang, J.; Chen, Z.; Tang, J.; Chen, X.; Lin, Y.; et al. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6): 186345.

Wang, X.; Fang, M.; Zeng, Z.; and Cheng, T. 2023. Where would i go next? large language models as human mobility predictors. *arXiv preprint arXiv:2308.15197*.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35: 24824–24837.

Wu, H.; Chen, Z.; Sun, W.; Zheng, B.; and Wang, W. 2017. Modeling trajectories with recurrent neural networks. IJCAI.

Xi, Z.; Chen, W.; Guo, X.; He, W.; Ding, Y.; Hong, B.; Zhang, M.; Wang, J.; Jin, S.; Zhou, E.; et al. 2023. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*.

Xie, J.; Zhang, K.; Chen, J.; Zhu, T.; Lou, R.; Tian, Y.; Xiao, Y.; and Su, Y. 2024. Travelplanner: A benchmark for real-world planning with language agents. *arXiv preprint arXiv:2402.01622*.

Yang, D.; Fankhauser, B.; Rosso, P.; and Cudre-Mauroux, P. 2020. Location prediction over sparse user mobility traces using rnns. In *Proceedings of the twenty-ninth international joint conference on artificial intelligence*, 2184–2190.

Yang, D.; Zhang, D.; and Qu, B. 2016. Participatory cultural mapping based on collective behavior data in location-based social networks. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 7(3): 1–23.

Yang, S.; Liu, J.; and Zhao, K. 2022. GETNext: trajectory flow map enhanced transformer for next POI recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on research and development in information retrieval*, 1144–1153.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; and Narasimhan, K. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Zhang, Z.; Amiri, H.; Liu, Z.; Züfle, A.; and Zhao, L. 2023. Large Language Models for Spatial Trajectory Patterns Mining. *arXiv preprint arXiv:2310.04942*.

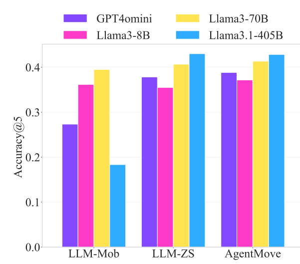
Zhang, Z.; Bo, X.; Ma, C.; Li, R.; Chen, X.; Dai, Q.; Zhu, J.; Dong, Z.; and Wen, J.-R. 2024. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501*.

Zhao, K.; Zhang, Y.; Yin, H.; Wang, J.; Zheng, K.; Zhou, X.; and Xing, C. 2020. Discovering subsequence patterns for next POI recommendation. In *IJCAI*, volume 2020, 3216–3222.

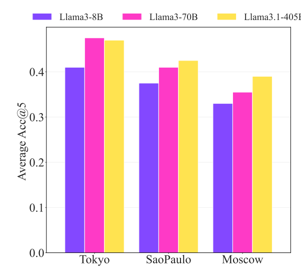
Appendix

The Impact of Different LLMs

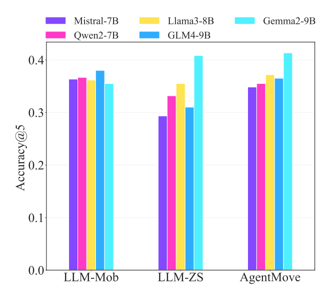
As the core foundation of AgentMove, the strength of base LLM's capabilities has a significant impact on the performance of AgentMove. Thus, we evaluate the impact of different LLMs with varying sizes and sources in Figure 6. We consider the influence of LLM size, LLM sources on the performance of three LLM based methods in Figure 6(a) and Figure 6(c). We dive into the detailed impacts of the LLM size and sources on AgentMove among three cities in Figure 6(b) and Figure 6(d).



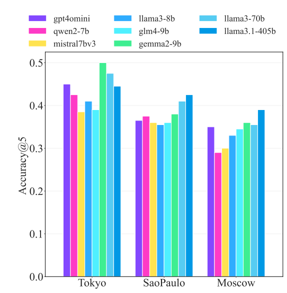
(a) Effects of LLM size on the performance of three LLM based methods.



(b) Effects of LLM size on the performance of AgentMove for three cities.



(c) Effects of LLM from different source on the performance of three LLM-based methods



(d) Effects of LLM from different source on the performance of AgentMove for three cities

Figure 6: The effects of LLM with varying sizes and sources on the prediction performance of three LLM based methods. Only results from Accuracy@5 are presented here, when similar trends are observed on other metrics.

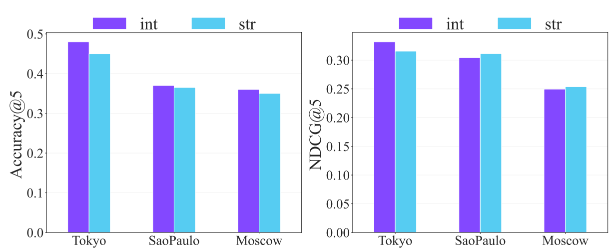
We first investigate how the LMM size affects the performance of LMM-based methods. In Figure 6(a), we find that LLMs with more parameters, such as Llama3-70B, generally outperform smaller model like LLama3-8B in the LLM-based methods, highlighting the advantage of increased model capacity in helping mobility prediction. It is also noted that the performance gain from scaling up the Llama3 is very small. The only exception is that LLM-

Mob in LLama3.1-405B performs much worse. After checking the results carefully, we find that this is caused by the wrong output format of LLM-Mob@Llama3.1-405B with too much detailed reasoning steps, which demonstrates the potential instability of the LLM-Mob among various LLMs. In Figure 6(c), we shift our focus to the impact of various 7B LLM with different training data and model structures. The results show that proposed AgentMove performs best adaptability among different LLMs. While LLM-Mob performs stable in all the 7B-LLMs, its performance on Gemma2-9B is far worse than other two methods.

We then discuss the detailed impacts of LLM size and source specifically on AgentMove's performance across three cities. Figure 6(b) reveals that that larger models, particularly Llama3.1-405B, generally deliver significant performance gains for AgentMove compared to smaller models like Llama3-8B across different cities. It is also observed that in Tokyo, Llama3-1-405B performs slightly weaker compared to Llama3-70B. This suggests that while larger models often excel, their effectiveness may vary depending on the unique mobility patterns and characteristics of each city. Figure 6(d) further shows the role of LMM with different training data and model structures in determining AgentMove's performance at different cities. Although the source Llama3-1-405B often leads in accuracy, other kinds of LLM sometimes perform similarly or better in certain contexts. In general, considering the cost and robustness, GPT4omini is a good start point for utilizing AgentMove in the mobility prediction task.

Effects of Different Location Representation

We explore the effects of the format of location ID in the prompt in Figure 7(a), where 'int' denotes to re-encoding the locations into ordered number in each city, 'str' denotes preserve the original string ID of venue in Foursquare. We find that in all the tree datasets, AgentMove with 'int' encoding strategy achieves the similar performance in two metrics compared with the representation of 'str' location ID. This result also demonstrate that the outstanding prediction performance of LLM based methods does not stem from the potential data leakage. Thus, all the results in our experiments use the 'str' encoding as the default encoding strategy.



(a)

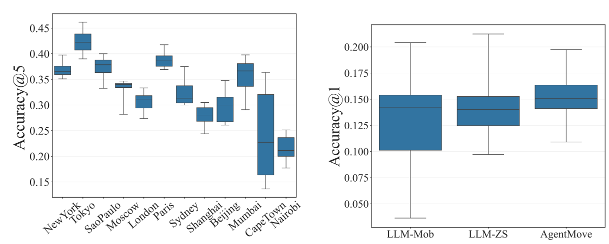
Figure 7: Effects of the format of location ID, where 'str' denotes the original location string in Foursquare, 'int' denotes that encoding the location into ordered integer in each city.

City	Users	Trajectories	Locations	Days	Records
Tokyo	12464	112942	83190	531	1030105
SaoPaulo	11856	77120	78904	531	809198
Moscow	10501	100854	93599	531	950898
NewYork	15785	28502	41386	531	380247
Sydney	1720	4557	10523	531	54250
Paris	6903	7559	19837	531	111325
London	9724	14596	28687	531	188530
Beijing	1076	1847	5753	531	21030
Shanghai-FSQ	1272	3238	8014	531	33129
Shanghai-ISP	1762	2844	12576	7	325215
Capetown	403	1234	2988	531	13303
Mumbai	1070	3070	7942	531	40592
Nairobi	356	2690	5807	531	28453

Table 3: Trajectory statistics of 12 cities around the world.

Other results of Geospatial Bias

Following the results in section , we reports the performance of AgentMove variants, as illustrated in Table 2, among different cities in Figure 8(a). In most cases, the performance fluctuation between different AgentMove variants in different cities are close. We also provide a analysis by comparing the Acc@1 of the three LLM-based methods at different cities in Figure 8(b).



(a) Performance of different variants of AgentMove on 12 cities.

(b) The distribution of Acc@1 across three methods on 12 cities.

Figure 8: Geospatial bias analysis of three LLM-based methods on the mobility prediction.

Details of Data

Detailed information of processed trajectory data from 12 cities is presented in Table 3.

Prompt Examples

Here, we present the detailed prompts for each LLM based methods.

Prompt of AgentMove

```
## Task
Your task is to predict <next_place_id> in
    <target_stay>, a location with an
    unknown ID, while temporal data is
    available.

## Predict <next_place_id> by considering:
```

```
1. The user’s activity trends gleaned from <historical_stays> and the current activities from  <context_stays>.
2. Temporal details (start_time and day_of_week) of the target stay, crucial for understanding activity variations.
3. The potential places that users may visit based on an overall analysis of multi-level urban spaces.
4. The personal profile and memory info extracted from the long trajectory history of each user.

## The potential places from the global spatial view:
{spatial_world_info}

## The nearby places visited by other users with similar mobility pattern:
{social_world_info}

## The personal profile and long memory:
{spatial_temporal_memory_info}

## The history data:
<historical_stays>: {historical_stays}
<context_stays>: {context_stays}
<target_stay>: {target_time, <next_place_id>}

## Output
Present your answer in a JSON object with:
"prediction" (list of IDs of the five most probable places, ranked by probability) and "reason" (a concise justification for your prediction).
```

Prompt for spatial-temporal memory unit.

```
### long term memory info
Place id to name mapping: {venue_id_to_name}.
In historical stays, The user frequently engages in activities at {frequent_hours}.
The most frequently visited venues are {frequent_venues}.
Hourly venue activities include {hourly_activity_desc}.
The user’s activity transitions often include sequences such as {transitions}.

### short term memory info
In recent context stays, user’s last visit was on {}
Frequently visited locations include: {}
Visit times: {}

### user profile
The user is most active at {most_frequent_hour} with {most_frequent_count} visits.
They frequently visit {most_frequent_venue_category} with {most_frequent_venue_count} visits
Based on the data, the user {’, ’.join(insights)}.
```

Prompts for world knowledge generator.

```
# Prompts for world knowledge generator

## prompts for extracting structured address info
{original address info from https://nominatim.org/ by querying via reverse API}
Please get the administrative area name, subdistrict name/neighbourhood name, access road or feeder road name, building name/POI name.
Present your answer in a JSON object with:’administrative’ (the administrative area name) ,’subdistrict’ (subdistrict name/neighbourhood name),’poi’(building name/POI name),’street’(access road or feeder road name which POI/building is on).
Do not include the key if information is not given.Do not output other content.

### block info
This trajectory moves within following administrative areas:
{administrative_area}
This trajectory sequentially visited following subdistricts, with the last subdistrict being the most recently visited:{}
Consider about following two aspects:
1.The frequency each subdistrict is visited.
2.Transition probability between two administrative areas.
Please predict the next subdistrict in the trajectory. Give {explore_num} subdistricts that are relatively likely to be visited. Do not output other content.

### poi and street info
This trajectory sequentially visited following POIs(Each POI is represented by ’POI name, the feeder road or access road it is on’), with the last POI being the most recently visited:{pois})
Consider about following two aspects:
1.The frequency each subdistrict is visited.
2.The frequency each poi is visited.
3.Transition probability between two subdistricts.
4.Transition probability between two pois.
Please predict the next poi in the trajectory.Give {explore_num} POIs that are relatively likely to be visited. Do not output other content.
```

```

# spatial world model info used in AgentMove
### Names of subdistricts that are relatively likely to be visited:
{block_info}
### Names of POIs that are relatively likely to be visited:
{poi_info}
```

Prompt for collective knowledge extractor.

```
## Finding neighbors
neighbors = list(graph.neighbors(venue_id))
sorted_neighbors_freq = [(n, 1) for n in neighbors if n not in context_trajs]

## Prompts in final reasoning step
1-hop neighbor places in the social world: {neighbors}
......
```

Prompt of LLM-Mob

```
Your task is to predict a user’s next location based on his/her activity pattern.
You will be provided with <history> which is a list containing this user’s historical stays, then <context> which provide contextual information
about where and when this user has been to recently. Stays in both <history> and <context> are in chronological order.
Each stay takes on such form as (start_time, day_of_week, duration, place_id). The detailed explanation of each element is as follows:
start_time: the start time of the stay in 12h clock format.
day_of_week: indicating the day of the week.
duration: an integer indicating the duration (in minute) of each stay. Note that this will be None in the <target_stay> introduced later.
place_id: an integer representing the unique place ID, which indicates where the stay is.

Then you need to do next location prediction on <target_stay> which is the prediction target with unknown place ID denoted as <next_place_id> and
unknown duration denoted as None, while temporal information is provided.

Please infer what the <next_place_id> might be (please output the 10 most likely places which are ranked in descending order in terms of probability), considering the following aspects:
1. the activity pattern of this user that you learned from <history>, e.g., repeated visits to certain places during certain times;
2. the context stays in <context>, which provide more recent activities of this user;
3. the temporal information (i.e., start_time and day_of_week) of target stay, which is important because people’s activity varies during different time (e.g., nighttime versus daytime)
and on different days (e.g., weekday versus weekend).

Please organize your answer in a JSON object containing following keys:
"prediction" (the ID of the five most probable places in descending order of probability) and "reason" (a concise explanation that supports your prediction). Do not include line breaks in your output.

The data are as follows:
<historical>: {historical_stays}
<context>: {context_stays}
<target_stay>: {target_time, <next_place_id>}
```

Prompt of LLM-ZS

```
Your task is to predict <next_place_id> in <target_stay>, a location with an unknown ID, while temporal data is available.

Predict <next_place_id> by considering:
1. The user’s activity trends gleaned from <historical_stays> and the current activities from  <context_stays>.
2. Temporal details (start_time and day_of_week) of the target stay, crucial for understanding activity variations.

Present your answer in a JSON object with:
"prediction" (IDs of the five most probable places, ranked by probability) and "reason" (a concise justification for your prediction).

The data:
<historical_stays>: {historical_stays}
<context_stays>: {context_stays}
<target_stay>: {target_time, <next_place_id>}
```

Parameter settings

Detailed parameter settings for each Markov and deep learning based baselines are presented in Table 4. For each baseline, we adapt the early stopping methods by considering the accuracy of validation set and learning rate schedule threshold. All the experiments of deep learning baselines are running on a machine with 64 cores, 512GB of memory, and

Table 4: Detailed parameter settings for Markov and deep learning based baselines.

Parameters	FPMC	RNN	DeepMove	LSTPM
batch size	1024	1024	128	128
learning rate (lr)	-	1e-3	1e-3	1e-3
lr schedule step	-	2	3	2
lr schedule decay	-	0.1	0.1	0.1
schedule threshold	-	1e-3	1e-3	1e-3
early stop lr	-	9e-6	9e-6	9e-6
L2	-	1e-5	1e-5	1e-6
max epoch	100	30	30	30
loc embed size	64	500	500	500
hidden embed size	-	500	500	500
dropout	-	0.3	0.5	0.8

2 NVIDIA RTX 4090 GPU, which is installed with Ubuntu 22.04.3 LTS.

All the generation parameter settings for LLM based methods are the same. The temperature is set as 0 for deterministic results, the maximum output token is 1000, the maximum input token is 2000, other parameters are not set and follow the default settings from API provider.

Data preprocessing

Preprocessing for Foursquare Data

As introduced in section , we select 12 cities around the world to evaluate the performance of proposed framework. We match each trajectories with the target cities by calculating the minimum distance to the city center. For the ordered trajectories in each city, we use 72 hours as the time window to split the trajectory into sessions. We filter the users with less than 5 sessions and filter sessions with less than 4 stays. Then, we divide each trajectory dataset into training, validation, and test sets in a ratio of 7:1:2. During the testing, we filter the users with less than 3 sessions or more than 50 sessions which is designed to ensure the quality of testing users and also balance the effects from different users. Different from the previous works, we do not specifically filter locations. All the users and trajectories of them are sorted by the id. We select one session of each user and aggregate the first n sessions from all the users to calculate the average accuracy. Here, n is utilized to control the cost of evaluation for LLMs and keep fixed in the experiment, which is set as 200 in most of the experiments. It is noted that only the volume of testing set is controlled for cost, the entire training set is provided to the deep learning based methods for training.

Preprocessing for ISP Data

Following the preprocessing in the original paper (Feng et al. 2019), we split the data into different sessions by merging trajectory points in the same day. Due to the regularity of human, there are too much repeated trajectory points in the original sessions. To make the prediction challenging, we compress the trajectory sessions by merging the same locations within a time window (2 hours) and ignoring the visiting occurred during the night (from 8 p.m. to 8 a.m.). While the ISP data lasts only 7 days, we split the whole data into training set, validation set and testing data in a ratio of 4:1:5 for preserving enough testing data. The minimum session filter parameter is changed from 3 to 1.