Unified Modeling and Clustering of Mobility Trajectories with Spatiotemporal Point Processes

Haowen Lin*, Yao-Yi Chiang†, Li Xiong‡, and Cyrus Shahabi *

*University of Southern California.{haowenli,shahahbi}@usc.edu
†University of Minnesota. yaoyi@umn.edu
‡Emory University. lxiong@emory.edu

Abstract

In various application domains like transportation, urban planning, and public health, analyzing human mobility, represented as a sequence of consecutive visits (aka trajectories), is crucial for uncovering essential mobility patterns. Current practices often discretize space and time to model trajectory data with sequence-analysis techniques like Transformers and LSTM, but this discretization tends to obscure the intrinsic spatial and temporal characteristics inherent in trajectories. Recent work shows the effectiveness of modeling trajectories directly in continuous space and time using the spatiotemporal point process (STPP). However, these approaches often assume that all observed trajectories originate from a single underlying dynamic. In reality, real-world trajectories exhibit varying dynamics or moving patterns. We hypothesize that grouping trajectories governed by similar dynamics into clusters before trajectory modeling could enhance modeling effectiveness. Thus, we present a novel approach that simultaneously models trajectories in continuous space and time using STPP while clustering them. Our method leverages a variational Expectation-Maximization (EM) framework to iteratively improve the learning of trajectory dynamics and refine cluster assignments within a single training phase. Extensive tests on synthetic and real-world data demonstrate its effectiveness in clustering and modeling trajectories.

1 Introduction

Recent advances in the Global Positioning System (GPS) and wireless technologies have led to the accumulation of a vast amount of trajectories, such as human mobility and vessel positioning data [10]. These real-world trajectories often exhibit diverse dynamics due to different moving behaviors or patterns (see Fig. 1). Understanding and modeling the underlying spatiotemporal dynamics of objects is important to many practical applications, such as predicting next locations, analyzing traffic flow, and detecting spatial outliers [21, 11].

Modeling trajectory is challenging due to the inherently irregular and asynchronous characteristics of moving dynamics, with each data point existing in continuous time and space. Prior research on trajectory mining applications often projects GPS coordinates onto discrete geographical grids and time intervals, utilizing recurrent neural networks (RNN) or Transformers to capture the sequential information of the trajectories for various analysis tasks such as trajectory simulation [21] and next location prediction [22]. However, modeling human movements requires algorithms that can effectively capture inherently complex spatial and temporal dependencies and transforming trajectories into regular grids and time intervals cannot accurately model real-world trajectories with irregular moving patterns.[24].

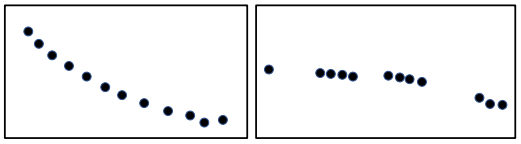

Figure 1: Example of trajectories with two different moving patterns.

To address the above issues, we build a trajectory learning model based on spatiotemporal point processes (STPPs), a robust and structured framework for modeling trajectory data in continuous space-time. [16] A recent study demonstrates the effectiveness of STPPs in modeling Point-of-Interest trajectories, emphasizing the importance of spatiotemporal dynamics. [24] However, they implicitly assume that all observed trajectories are generated by a single moving dynamic to which they try to fit. Instead, real-world trajectories exhibit varying dynamics. We hypothesize that if we can group trajectories governed by similar dynamics into a single cluster, STPP will exhibit greater efficacy in modeling trajectories within each cluster. For instance, research shows that modeling the inherent modality or moving behaviors of trajectories can help understand how humans move in space and time, which improves other down-

stream tasks such as urban mobility simulation [27, 12]. Unfortunately, this gives rise to a classic 'chicken-and-egg' predicament: we must first employ STPP to model the trajectories and capture their underlying dynamics before we can cluster similar ones; on the other hand, we must cluster them based on the dynamics first before we can effectively learn the STPP model.

There are several potential approaches to go around the problem. First, we can employ a two-stage approach (clustering-then-modeling) where we first cluster trajectories based on the raw features in the trajectory space, such as their spatiotemporal similarity (e.g., using Euclidean distance between the raw trajectories) and then model each cluster with an STPP. Clearly, these clusters may not capture the underlying moving dynamics and may not be optimally aligned with subsequent trajectory learning. Second, we can employ an alternative two-stage approach (modeling-then-clustering), where we first employ models to learn representations from the trajectories, enabling the modeling of continuous-time spatiotemporal dynamics and feature extraction from the trajectories. Subsequently, we can utilize these features to discern the cluster structure. However, previous research has indicated that such a two-phase training paradigm leads to unstable clustering outcomes, often sensitive to the quality of the learned features [25]. Despite some recent efforts to jointly learn representations and clustering information in a single training phase for discrete domain sequences [26], we are unaware of any previous work that directly applies this approach to trajectories embedded within continuous time and space, aiming to capture the underlying moving dynamics.

In this paper, we propose a novel framework, named Mobility-aware Deep Trajectory Modeling and Clustering (DTMC), which unifies modeling the trajectory dynamics and clustering them based on their dynamics simultaneously via spatiotemporal point processes. Specifically, we decompose the hidden embedding for the trajectories into two representations: the individual representation acquired through a neural STPP model, and another cluster representation that encapsulates the clustered moving patterns. To obtain cluster assignments, we introduce variational inference where each trajectory can learn a conditional probability based on cluster assignment in an Expectation-Maximization (EM) framework that iteratively refine the trajectory embeddings and cluster assignment and resolve the chicken and egg problem. To show the effectiveness of DTMC, we first compare its clustering results with three types of clustering methods: clustering-only, modeling-then-clustering, and concurrent-modeling-clustering methods. Subsequently, we compare the predictive performance of the DTMC model with three types of modeling approaches: single STPP model, clustering-then-modeling, and concurrent-clustering-modeling methods. The findings underscore DTMC's superiority not only in enhancing trajectory modeling but also in achieving remarkable results in trajectory clustering.

In summary, our contributions are:

- We propose a novel unified framework to simultaneously model and cluster trajectories based on inherent moving patterns. Based on the inferred clustering results for each trajectory, our method improves the performance of trajectory representations learning by capturing the underlying clustered moving patterns.
- We model the trajectories with STPPs, which can learn continuous spatiotemporal moving dynamics via neural differential equations.
- Extensive experiments on both synthetic and real datasets show the expressiveness of our proposed framework on both trajectory clustering and predictive tasks.

2 Preliminaries

2.1 Problem Definition

Definition 1. ***Mobility Trajectory*** A mobility trajectory is a sequence of spatiotemporal points generated by an individual in daily life and recorded through GPS. It is represented by a sequence of chronologically ordered points $S = \{(t_i, \boldsymbol{x}_i)\}_{i=1}^{L}$, where t_i is the timestamp, $\boldsymbol{x}_i$ is the location, and L is the total length of the mobility trajectory.

2.2 Background and Related Works

Spatiotemporal Point Processes [6] are concerned with modeling sequences of random events in continuous space and time. We denote an event sequence as $S = \{(t_i, \boldsymbol{x}_i)\}_{i=1}^{L}$, where $t_i \in \mathbb{R}$ is the timestamp, $\boldsymbol{x}_i \in \mathbb{R}^d$ is the associated spatial location at each timestamp, and L is the total number of events. An STPP first defines the conditional intensity function:

$$\lambda(t, \boldsymbol{x}|S_t) = \lim_{\Delta t \to 0, \Delta \boldsymbol{x} \to 0} \frac{p(t_i \in [t, t+\Delta t], \boldsymbol{x}_i \in B(\boldsymbol{x}, \Delta \boldsymbol{x})|S_t)}{|B(\boldsymbol{x}, \Delta \boldsymbol{x})|\Delta t}, \tag{2.1}$$

where $S_t = \{(t_i, \boldsymbol{x}_i)|t_i < t, (t_i, \boldsymbol{x}_i) \in S\}$ denotes the history of events prior to time t, and $B(\boldsymbol{x}, \Delta \boldsymbol{x})$ denotes a ball centered at $\boldsymbol{x} \in \mathbb{R}^d$ and with radius $\Delta \boldsymbol{x}$. The non-negative conditional intensity function $\lambda(t, \boldsymbol{x}|S_t)$, often denoted as $\lambda^*(t, \boldsymbol{x})$, describes the instantaneous probability of the i-th event occurring at t and location

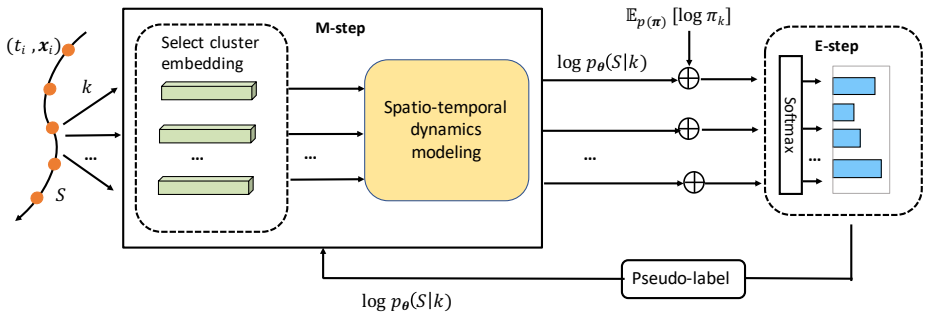

Figure 2: The workflow of our proposed modeling and clustering framework.

$\boldsymbol{x}$ given $i-1$ previous events. The joint log-likelihood of observing the trajectory history S within a time interval of $[0, T]$ is then given by

(2.2)
$$\log p(S) = \sum_{i=1}^{L} \log \lambda^*(t_i, \boldsymbol{x}_i) - \int_0^T \int_{\mathbb{R}^d} \lambda^*(\tau, \boldsymbol{u})\, d\boldsymbol{u} d\tau.$$

Trajectory Clustering Methods aim to gain space time insights inside trajectory data. Most clustering techniques working on raw trajectories adopt predefined distance or similarity metrics such as the classic Euclidean, Hausdorff and dynamic time warping (DTW) suited to specific applications [2]. However, these methods are ineffective due to the strong parametric assumption, which fails to account for the complex spatiotemporal associations underlying trajectories. Recent advances have shifted towards deep learning approaches for trajectory clustering [25, 8]. These techniques commonly utilize autoencoder-based strategies, converting trajectories into fixed-length vectors for clustering within a bifurcated training process [13]. However, such clustering methods are sensitive to small changes in the learned features. A recent work provides an end-to-end clustering algorithm considering temporal dynamics [28]. However, it only focuses on temporal data such as stocks and clinic visits, without considering spatial modeling. Another end-to-end trajectory clustering algorithm is [26], but it transforms the trajectories with a description of the POIs (as opposed to using raw GPS points in our method) to cluster based on mobility purposes (e.g., shopping, eating).

In our work, we propose to learn trajectories based on grouped moving dynamics where we operate on GPS points localized in continuous time and space. Therefore, our proposed framework tries to learn a better trajectory representation by considering the continuous characteristics of states in time and space and at the same time, taking into consideration different moving dynamics by clustering the trajectories based on the dynamics.

3 Mobility-aware Deep Trajectory Modeling and Clustering (DTMC)s

In this section, we present our DTMC approach, which integrates trajectory representation learning and clustering. The workflow of our proposed framework is shown in Fig. 2. More specifically, we present the variational EM framework and explain how the probabilistic model infers the cluster assignment (Section 3.1). We then elaborate on how we learn spatiotemporal dynamics by incorporating cluster embedding (Section 3.2) and finally summarize the training procedure that iteratively updates the cluster embedding and spatiotemporal models for each cluster based on the cluster membership (Section 3.2.2).

3.1 Trajectory Cluster Inference Given a set of trajectories $\boldsymbol{S} = \{S_n\}_{n=1}^N$, where $S_n = \{(t_i, \boldsymbol{x}_i)\}_{i=1}^L$ is a sequence defined in Def 1, our goal is to divide these N sequences into K groups such that the trajectories generated by similar spatiotemporal dynamics are grouped. We assume for each S_n, there is a corresponding latent variable $\boldsymbol{z}_n \in \{0,1\}^K, \sum_{k=1}^K z_{nk} = 1$ denoting the cluster membership i.e. $z_{nk} = 1$ if and only if S_n belongs to group k. Each $\boldsymbol{z}_n$ is drawn from a categorical distribution defined on $\boldsymbol{\pi} = [\pi_1, ..., \pi_K] \in \mathbb{R}^K$ where $\boldsymbol{\pi}$ can be a static prior on cluster types or drawn from a Dirichlet distribution. Denoting the variable to represent the distribution of cluster assignment as $\boldsymbol{Z} = \{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_N\}$, and our goal is to both maximize the likelihood of the data while also infer the latent variables $\boldsymbol{Z}$.

Given the prior $\boldsymbol{\pi}$, the conditional distribution of $\boldsymbol{Z}$ is formed as: $p(\boldsymbol{Z}|\boldsymbol{\pi}) = \prod_{n=1}^N \prod_{k=1}^K \pi_k^{z_{nk}}$ where π_k is the k-th dimension of $\boldsymbol{\pi}$. Given $\boldsymbol{Z}$, we model the conditional probability of $\boldsymbol{S}$ as:

(3.3)
$$p_{\boldsymbol{\theta}}(\boldsymbol{S}|\boldsymbol{Z}) = \prod_{n=1}^{N} \prod_{k=1}^{K} p_{\boldsymbol{\theta}}(S_n|k)^{z_{nk}},$$

where $\boldsymbol{\theta}$ denotes all the learnable parameters of the model. For a trajectory S_n, we input it together with cluster label k into the model, where we adopt a

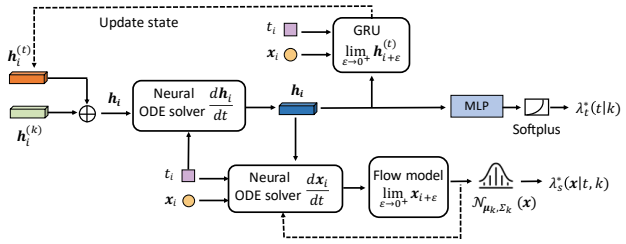

Figure 3: Illustration of the spatiotemporal modeling with cluster embedding.

parametric latent embedding for each cluster and obtain the conditional probability $p_{\boldsymbol{\theta}}(S_n|k)$. Thus, we can factorize the joint distribution of all variables $p_{\boldsymbol{\theta}}(\boldsymbol{S}, \boldsymbol{Z}, \boldsymbol{\pi})$ by

$$
\begin{aligned}
p_{\boldsymbol{\theta}}(\boldsymbol{S}, \boldsymbol{Z}, \boldsymbol{\pi}) &= p(\boldsymbol{\pi}) p(\boldsymbol{Z}|\boldsymbol{\pi}) p_{\boldsymbol{\theta}}(\boldsymbol{S}|\boldsymbol{Z}) \\
&= p(\boldsymbol{\pi}) \prod_{n=1}^{N} \prod_{k=1}^{K} [\pi_k \exp(\sum_{i=1}^{L} \log \lambda^*_{\boldsymbol{\theta}}(t_i, \boldsymbol{x}_i|k) \\
&\quad - \int_0^T \int_{\mathbb{R}^d} \lambda^*_{\boldsymbol{\theta}}(\tau, \boldsymbol{u}|k)\, d\boldsymbol{u} d\tau)]^{z_{nk}}.
\end{aligned} \tag{3.4}
$$

3.2 Learning spatiotemporal Dynamics of Cluster In this section, we elaborate on how to model the trajectory and learn the conditional intensity function $\lambda^*(t, \boldsymbol{x}|k)$ for cluster k so that $p_{\boldsymbol{\theta}}(\boldsymbol{S}|\boldsymbol{Z})$ in Eqn. 3.3 can be computed. Specifically, we decompose the conditional intensity based on [5] as

$$
\lambda^*(t, \boldsymbol{x}|k) = \underbrace{\lambda^*_t(t|k)}_{\text{Temporal}} \underbrace{\lambda^*_s(\boldsymbol{x}|t, k)}_{\text{Spatial}}. \tag{3.5}
$$

where $\lambda^*_t(t|k)$ is the intensity function for temporal process and $\lambda^*_s(\boldsymbol{x}|t, k)$ is the conditional intensity of spatial location $\boldsymbol{x}$ at t given the past trajectory history. Consequently, derived from Eqn. 2.2, we can compute $p_{\boldsymbol{\theta}}(S_n|k)$ accordingly:

$$
\begin{aligned}
\log p_{\boldsymbol{\theta}}(S_n|k) = &\underbrace{\sum_{i=1}^{L} \log \lambda^*_t(t_i|k) - \int_0^T \lambda^*_t(\tau|k)\, d\tau}_{\text{Temporal log-likelihood}} \\
&+ \underbrace{\sum_{i=1}^{L} \log \lambda^*_s(\boldsymbol{x}_i|t_i, k)}_{\text{Spatial log-likelihood}}.
\end{aligned} \tag{3.6}
$$

In the following, we will first introduce how we incorporate cluster latent embedding to get the hidden states of a trajectory. These hidden states not only encapsulate the latent characteristics of a cluster but are also informed by the trajectory data at each time point. We show how to construct the model to learn the temporal and spatial dynamics which will be jointly conditioned on the hidden states. Fig. 3 shows the overall learning process which we will explain in detail below.

3.2.1 Decomposing Hidden Variables To model the conditional intensity function $\lambda^*(t, \boldsymbol{x}|k)$ for K clusters respectively, we introduce cluster hidden states $\boldsymbol{h}^{(k)}_{1:L}$ where we try to combine cluster latent information with the representations learned from the temporal dynamics and spatial dynamics together. Similar to a recurrent neural network, where at every time point, we acquire a hidden state $\boldsymbol{h}^{(t)}_{1:L}$ that acts as a summary of the history trajectory and would be used to predict future temporal and spatial variables t_i and $\boldsymbol{x}_i$. We then augment these representations for each step by adding cluster embedding:

$$
\boldsymbol{h}_{1:L} = \boldsymbol{h}^{(k)}_{1:L} + \boldsymbol{h}^{(t)}_{1:L}. \tag{3.7}
$$

3.2.2 Temporal Modeling After augmenting the cluster embedding, we model hidden state dynamics with jumps to parameterize the intensity function $\lambda^*_t(t|k)$ which has been proved effective in [9]. Specifically, we apply the Neural ODE to ensure a continuous-time hidden state and then trigger instantaneous updates in response to the introduction of a new point. This mechanism is essential because it not only captures the continuous temporal pattern between each point, but also allows historical points to influence future movement. In summary, the continuous flow and instantaneous update can be formulated as:

$$
\frac{d\boldsymbol{h}_i}{dt} = f_h(t_i, \boldsymbol{h}_i), \quad \lim_{\varepsilon \to 0^+} \boldsymbol{h}^{(t)}_{i+\varepsilon} = g_h\left(t_i, \boldsymbol{x}_i, \boldsymbol{h}_i\right), \tag{3.8}
$$

where f_h is a multi-layer perceptron (MLP) modeling the continuous evolution between event times, g_h is a gated recurrent unit (GRU) modeling the instantaneous updates of the hidden states at event time, and $\boldsymbol{h}^{(t)}_{i+\varepsilon}$ denotes the hidden state at time $t_i + \varepsilon$. As a decoder of the hidden representations, we use a standard multi-layer fully connected neural network with a softplus activation to ensure the intensity is positive. Thus, given $\boldsymbol{h}^{(t)}_{1:L}$, the conditional temporal intensity is computed as:

$$
\lambda^*_t(t_{1:L}|k) = \text{Softplus}(\text{MLP}(\boldsymbol{h}^{(t)}_{1:L})). \tag{3.9}
$$

3.2.3 Spatial Modeling Similar to temporal modeling, a core component for modeling locations in the

spatial domain is an interpolated continuous spatial intensity function. We determine the spatial dynamic based on Continuous Normalizing Flow (CNF) to model the conditional spatial density $p(\boldsymbol{x}|t)$. In several recent studies, CNFs have been prove effective to model distributions on a real-valued axis such as spatial locations and point clouds [15]. In the same way to temporal domain, we want the spatial intensity function to be continuous everywhere except for the observed point. Consequently, the spatial pattern is updated by a continuous-time normalizing flow that evolves the distribution continuously, and a standard flow model that changes the distribution instantaneously after conditioning on new events. Additionally, the update of the normalizing flow is conditioned on $\boldsymbol{h}_{1:L}$ (with cluster information included), with the assumption that the trajectory history augmented with cluster information has an impact on the future spatial distribution. Such dynamics can be formulated as follows:

$$\frac{d\boldsymbol{x}_i}{dt} = f_x(t_i, \boldsymbol{x}_i, \boldsymbol{h}_i), \quad \lim_{\varepsilon \to 0^+} \boldsymbol{x}_{i+\varepsilon} = g_x\left(t_i, \boldsymbol{x}_i, \boldsymbol{h}_i\right), \tag{3.10}$$

where f_x is modeled by a continuous normalizing flow, g_x is realized by a standard linear flow, and $\boldsymbol{x}_{i+\varepsilon}$ denotes the location at time $t_i + \varepsilon$.

Since the spatial variables are real-valued features, we parameterize the conditional spatial intensity $\boldsymbol{x}$ with a Gaussian mixture model as:

$$\lambda_s^*(\boldsymbol{x}_{1:L}|t_{1:L}, k) = \mathcal{N}_{\boldsymbol{\mu}_k, \Sigma_k}(\boldsymbol{x}_{1:L}), \tag{3.11}$$

where $\boldsymbol{\mu}_k$ and Σ_k are respectively the learnable mean vector and the learnable covariance matrix of the Gaussian distribution for the k-th cluster. The spatial log-likelihood in Eqn. 3.6 can then be evaluated accordingly.

3.3 Training Algorithm In this section, we explain how we optimize and train the framework. We leverage a variational EM framework, where the spatiotemporal modeling learns the spatial and temporal intensity function of each cluster to predict the joint log-likelihood of the trajectory data, whereas we can infer cluster assignment and get the posterior $p_{\boldsymbol{\theta}^*}(\boldsymbol{Z}|\boldsymbol{S})$ based on the spatiotemporal log-likelihood from each cluster. To be more specific, the framework tries to maximize the log-likelihood function $p_{\boldsymbol{\theta}}(\boldsymbol{S})$. As directly optimizing the function is often hard, we resort to variational methods and introduce a varational distribution $q(\boldsymbol{Z}, \boldsymbol{\pi})$ to approximate the posterior $p_{\boldsymbol{\theta}}(\boldsymbol{Z}, \boldsymbol{\pi}|\boldsymbol{S})$, and thus the framework instead optimizes the evidence lower bound (ELBO) as below :

$$\mathcal{L}(q, \boldsymbol{\theta}) = \log p_{\boldsymbol{\theta}}(\boldsymbol{S}) - \mathrm{KL}(q(\boldsymbol{Z}, \boldsymbol{\pi})||p_{\boldsymbol{\theta}}(\boldsymbol{Z}, \boldsymbol{\pi}|\boldsymbol{S})). \tag{3.12}$$

Using mean field approximation [19], we have $q(\boldsymbol{Z}, \boldsymbol{\pi}) = q(\boldsymbol{Z})p(\boldsymbol{\pi})$. Therefore, we derive :

$$\mathcal{L}(q, \boldsymbol{\theta}) = \int q(\boldsymbol{Z}) \mathbb{E}_{p(\boldsymbol{\pi})} \left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{S}, \boldsymbol{Z}|\boldsymbol{\pi})}{q(\boldsymbol{Z})}\right] d\boldsymbol{Z}. \tag{3.13}$$

The ELBO can be optimized by alternating between optimizing the variational distribution $q(\boldsymbol{Z})$ (i.e., E-step) to approximate the posterior and optimizing the model parameters $\boldsymbol{\theta}$ (i.e., M-step) such that $\log p_{\boldsymbol{\theta}}(\boldsymbol{S})$ is maximized to better characterize the trajectories.

3.3.1 E-step: Update Cluster Assignment In the E-step, we fix the model parameters $\boldsymbol{\theta}$ and aim to update $q(\boldsymbol{Z})$ to maximize the ELBO. The log of the optimized $q(\boldsymbol{Z})$ is given by:

$$\begin{aligned} \log q^*(\boldsymbol{Z}) =& \mathbb{E}_{p(\boldsymbol{\pi})} \log p_{\boldsymbol{\theta}}(\boldsymbol{S}, \boldsymbol{Z}|\boldsymbol{\pi}) \\ =& \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \{\mathbb{E}_{p(\boldsymbol{\pi})}[\log \pi_k] + \log p_{\boldsymbol{\theta}}(S_n|k)\}. \end{aligned} \tag{3.14}$$

Normalizing the above formulation, we obtain:

$$q^*(\boldsymbol{Z}) = \prod_{n=1}^{N} \prod_{k=1}^{K} r_{nk}^{z_{nk}}, \tag{3.15}$$

where $r_{nk} = \frac{\exp\left(\mathbb{E}_{p(\boldsymbol{\pi})}[\log \pi_k] + \log p_{\boldsymbol{\theta}}(S_n|k)\right)}{\sum_{\kappa=1}^{K} \exp\left(\mathbb{E}_{p(\boldsymbol{\pi})}[\log \pi_\kappa] + \log p_{\boldsymbol{\theta}}(S_n|\kappa)\right)}$ is the pseudo-label.

As $q(\boldsymbol{Z})$ is an approximation of $p_{\boldsymbol{\theta}}(\boldsymbol{Z}|\boldsymbol{S})$, when the model is well-trained, $p_{\boldsymbol{\theta}^*}(z_{nk} = 1|S_n) = r_{nk}$, i.e. the posterior probability that S_n in group k is r_{nk}.

3.3.2 M-step: Update Model Parameters In M-step, we fix $q(\boldsymbol{Z})$, and optimize $\mathcal{L}(q, \boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$. Since $\boldsymbol{\theta}$ are learnable parameters of the model, we optimize $\boldsymbol{\theta}$ via gradient descent with the loss function given by :

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{Z})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{S}|\boldsymbol{Z})] = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \log p_{\boldsymbol{\theta}}(S_n|k). \tag{3.16}$$

4 Experiments

To understand different moving patterns in trajectories and validate our hypothesis that capturing the underlying moving dynamics can help with trajectory modeling,

we conduct experiments on both the task of trajectory clustering (Section 4.4 and 4.5) and trajectory modeling (Section 4.6) to evaluate the performance of DTMC.

4.1 Dataset Synthetic Datasets We use two types of traditional STPPs (with different parameters) to simulate three different moving patterns (moving differently in both temporal and spatial domain)[1]: Spatiotemporal homogeneous Poisson process (STHP), and Spatiotemporal Hawkes process with gaussian diffusion kernel (STHG) with two different parameter settings (labeled as STHG1 and STHG2 to represent different moving patterns with similar behavior in temporal axis but differ significantly in spatial distributions). Additionally, we generate a moving pattern based on the "uniform walk assumption", where an agent consistently moves at a uniform speed (UNI). All trajectory simulations are defined within $S \times T = [0, 2]^2 \times [0, 10]$. Each moving dynamics has 1000 event sequences, each containing $L = 25$ points. We merge the above trajectories to generate three datasets with different number of clusters (K) as ground truth: $\boldsymbol{K = 2}$: STHP + STHG1; $\boldsymbol{K = 3}$: STHP + STHG1 + STHG2; $\boldsymbol{K = 4}$: STHP + STHG1 + STHG2 + UNI. In this way, we generate the synthetic datasets to mimic the situations in real-world where different moving patterns of trajectories are mixed together.

Real-world datasets We evaluate the performance of DTMC on real-world datasets where the trajectories are specified as a list of tuples of: user identifier, latitude and longitude, and timestamp. We collect mobility trajectories in Houston from Veraset[2], which provides movement data collected through GPS signals from cell phones in March 2020. We sample 1,000 trajectories due to the large data size. Another real-world dataset [23] was collected from Foursquare, Tokyo, which includes 1000 user check-ins within a duration of one month.

4.2 Compared Methods To evaluate the clustering ability of DTMC, we compare our model with three types of baselines: clustering-only, modeling-then-clustering, and concurrent-modeling-clustering. Later, for evaluating the modeling accuracy, we combine some of the baseline approaches to create clustering-then-modeling approaches for comparison.

The clustering-only methods extract features from trajectories and then apply clustering on them. **KM-RAW** uses raw trajectory as input and applies K-means[3] clustering with Euclidean distance. **KM-DTW** [14] calculates distance matrix using Dynamic Time Warping (DTW) distance[4] and subsequently apply K-means clustering on sequential data. **DBSCAN** employs a density-based clustering approach [17] to cluster the raw trajectories. **GMVAE** [7] applies a Gaussian Mixture Variational Autoencoder for unsupervised clustering. We develop the encoder and decoder with GRU [4] layers to work with sequences (without modeling the continuous spatiotemporal dynamics in the network). **GMVAE+** a variant of GMVAE, where we introduce a supervised loss into GMVAE to align its cluster results with those produced by K-means aligning with our algorithm which is pre-trained with K-means cluster results.

Modeling-then-clustering baselines model trajectories using traditional STPP model and apply clustering on the model parameters. **HPGM +BGM [28]** learns a specific Hawkes process for the temporal domain and applies a history-dependent Gaussian mixture model for the spatial domain and applies Bayesian Gaussian Mixture model to the learned parameters for clustering.

Concurrent-modeling-clustering baseline models trajectories using neural network and perform clustering simultaneously. **THP-EM [29]**. THP leverages the Transformer encoder for temporal point process representation learning. In the original work, THP was limited to learning the representation for sequences with continuous time only (without proper spatial modeling). We discretize our spatial points into 10×10 grids and use grid IDs as markers. For the K clusters, we initialize K different THP models and then apply the EM algorithm to learn the cluster assignment [28].

4.3 Evaluation Metrics To evaluate the clustering ability, we use three clustering metrics that are widely used: **Clustering Purity (CP) [3]**: The ratio between the number of correctly matched class samples and the number of total data points. **Adjusted Rand Index (ARI)**: The similarity of predicted and ground truth assignments. [20] **Normalized Mutual Information (NMI)**: The reduction in entropy of class labels when the cluster labels are given. Note that CP, ARI, and NMI only work when the ground-truth cluster assignments are known, which is only available for synthetic datasets. For real-world datasets, since the ground truth clustering assignment is unknown, we report **silhouette score** [18] based on the clustering results and calculate the Euclidean distance on the raw trajectories.

Learning Performance To evaluate the learning ability, we report **Log-likelihood** as the metric for trajectory sequences fitting (the higher, the better) [28]. We randomly split the data set into training (80%) and

[1] https://github.com/meowoodie/spatiotemporal-Point-Process-Simulator

[2] https://www.veraset.com/about-veraset

[3] https://scikit-learn.org/stable/

[4] https://github.com/maikol-solis/trajectory_distance

Dataset	$K=2$			$K=3$			$K=4$		
	CP ↑	ARI ↑	NMI ↑	CP ↑	ARI ↑	NMI ↑	CP ↑	ARI ↑	NMI ↑
KM-RAW	0.91	0.71	0.60	0.65	0.41	0.42	0.71	0.54	0.57
KM-DTW	0.83	0.65	0.61	0.66	0.42	0.43	0.62	0.38	0.47
DBSCAN	0.84	0.47	0.50	0.63	0.45	0.50	0.54	0.30	0.40
HPGM+BGM	0.62	0.05	0.11	0.41	0.07	0.03	0.35	0.05	0.04
GMVAE	0.75	0.41	0.36	0.60	0.28	0.27	0.50	0.23	0.29
GMVAE+	0.76	0.43	0.38	0.66	0.32	0.41	0.67	0.41	0.43
THP-EM	0.92	0.72	0.70	0.65	0.40	0.42	0.72	0.55	0.58
DTMC	**0.97**	**0.88**	**0.83**	**0.73**	**0.48**	**0.51**	**0.77**	**0.61**	**0.63**

Table 1: Cluster performance comparison of our model and baselines on the synthetic datasets, where the lower value indicates a better performance. Bold denotes the best(highest) results and the underline denotes the second-best results.

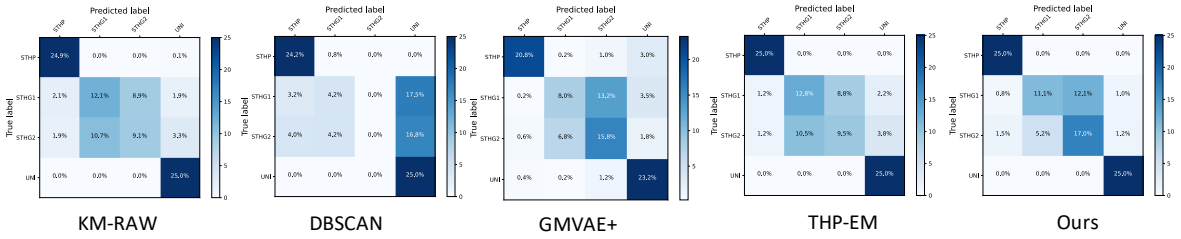

Figure 4: Confusion matrix on synthetic dataset when $K=4$.

testing sets (20%), where we train the model on the training set and report log-likelihood on spatial and temporal domains separately on held-out test data.

4.4 Clustering Performance on Synthetic Dataset We assume that the ground truth K and cluster size distribution $\boldsymbol{\pi}$ is given in the synthetic dataset evaluation. As we can observe in Table 1, the modeling-then-clustering baseline HPGM+BGM performs worst among all metrics. This is because it assumes all the trajectories strictly follow parametric Hawkes processes, which does not match reality. Additionally, it employs a two-step process for feature extraction and clustering, where each step is relatively independent. GMVAE and GMVAE+ outperforms HPGM+BGM significantly. It is reasonable as these two methods fit trajectories with the neural network and provide an end-to-end clustering framework. However, the backbones of both versions of GMVAE are simple RNNs that lack proficiency in modeling trajectories with GPS points in continuous time and space. THP-EM achieves the second-best performance, possibly reflecting its ability to discern diverse movement patterns presented in temporal space. Overall, DTMC achieves significant improvement across various datasets, which demonstrates the expressiveness of our clustering framework. We further verify our assumption by visualizing the confusion matrix of selected models on $K=4$ in Fig. 4. Note that although most methods (including ours) effectively distinguish STHP and UNI due to their significantly distinct temporal and spatial movement patterns, our method excels in differentiating STHG1 and STHG2, where these two moving patterns are simulated under same parameters in temporal domain but with different parameters for spatial distributions. It proves the importance and necessity of modeling trajectories as STPP to accurately learn the inherent moving patterns.

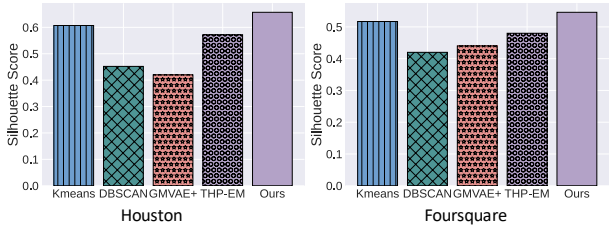

Figure 5: Silhouette score on real-world dataset. The higher value represents a better cluster quality.

4.5 Clustering Performance on Real-World Dataset For real-world datasets, we calculate the silhouette score on the clusters generated from each method. Fig. 5 shows the comparison results of two real-world datasets. Overall, DTMC exhibits superior performance, indicating that its generated clusters are distinctly separated from one another. K-means ranks second. This is not surprising as we directly calculate the silhouette score based on Euclidean distance on raw trajectories, consistent with how K-means perform the clustering. However, note that the purpose of the clustering is not necessarily to group the trajectories that are similar in Euclidean distance but on how well each group can be modeled later. Towards this end, in the next section, we

Model	STHP		STHG1		STHG2		UNI	
	Temporal	Spatial	Temporal	Spatial	Temporal	Spatial	Temporal	Spatial
Single STTP-model	0.603	-2.925	0.179	-2.064	0.136	-2.345	-0.181	-0.334
Clustering-then-modeling (K-Means)	0.613	-2.72	0.213	-1.944	0.148	-2.323	0.601	-0.177
Clustering-then-modeling (GMVAE+)	0.608	-2.715	0.203	-1.945	0.154	-2.329	0.604	-0.174
THP-EM	0.615	–	0.210	–	0.151	–	0.619	–
DTMC	**0.649**	**-2.530**	**0.243**	**-1.733**	**0.218**	**-2.051**	**0.637**	**-0.135**

Table 2: Log-likelihood per event on synthetic dataset (higher is better).

compare the modeling quality of the clusters.

4.6 Representation Learning Performance via Log-likelihood Our main thesis in this paper is that segregating trajectories into groups based on shared spatiotemporal dynamics would enhance the accuracy of trajectory modeling. In this section, we report log-likelihood to verify this claim and evaluate the effectiveness of our approach. In order to show the benefits of utilizing the clusters for the trajectory modeling, we compare DTMC with three types of learning methods: 1) Modeling-only: **Single STPP-model** where we do not cluster the trajectories and only apply a single STPP model on all trajectory data; 2) Clustering-then-modeling: here we first cluster trajectories with either K-means or GMVAE+[5] and then train K different STPP models on each cluster separately. The GMVAE+ variation represents a clustering approach that specifically clusters for better modeling while K-Means represents methods that clusters for similarity. 3) Concurrent-clustering-modeling: **THP-EM** where we apply the same EM algorithm to learn the cluster assignment and report learning in the temporal domain.

Table 2 reports log-likelihood evaluation on the synthetic datasets. As we can observe, Single STPP-model performs the worst since it does not group trajectories and thus must capture different moving dynamics using a single set of model parameters. THP-EM ranks second in the temporal learning performance as it also efficiently capture the underlying group patterns in temporal domains. However, it does not outperform our model in the temporal domain, suggesting that modeling temporal distribution without conditioning on the space attributes hampers comprehensive temporal domain learning. Our DTMC approach achieves the best performance in all cases, which demonstrates the power of clustering trajectories for the purpose of better learning. The results also show that the relative performance gain of our method is more obvious with datasets with more complex spatiotemporal dynamics patterns such as STHG1 and STHG2, which would be more useful in practical applications.

[5]Since the original GMVAE+ network cannot be directly used for prediction and log-likelihood valuation, we utilize the clustering result of GMVAE+ and then train STPP for each cluster

Model	Houston		Foursquare	
	Temporal	Spatial	Temporal	Spatial
Single ST-model	0.627	1.336	2.013	-2.115
Clustering-then-modeling (K-Means)	0.823	1.349	2.075	-1.939
Clustering-then-modeling (GMVAE+)	0.813	1.356	2.063	-1.936
THP-EM	0.836	–	2.076	–
DTMC	**0.881**	**1.423**	**2.082**	**-1.912**

Table 3: Log-likelihood per event on real-world data.

For real-world datasets, where the ground truth number of clusters (K) is unavailable, we employ K-means clustering on the trajectories and apply elbow method [1] to determine the best choice of K (6 for Houston and 4 for Foursquare, respectively). Table 3 reports the corresponding log-likelihood. Similar to synthetic datasets, we observe that the explicit clustering of moving patterns can significantly boost the performance of modeling the spatiotemporal dynamics of trajectories.

5 Conclusion

Real-world trajectories are governed by different underlying moving dynamics. Capturing groups of similar trajectories during the learning process enhances the quality of trajectory representation for predictive analysis. In this paper, we proposed a novel deep learning framework, DTMC, that can concurrently clusters and models trajectories based on their inherent moving patterns. Extensive experiments demonstrate the superior performance of DTMC in differentiating trajectory moving patterns and representation learning as compared to various baseline methods that follow a sequential clustering-then-modeling or modeling-then-clustering approach. Moreover, it surpasses approaches that project trajectories into discrete space, resulting in the loss of detailed spatial and temporal characteristics. In future research, we aim to expand the application of our model across diverse datasets and and utilize the model to gen-

erate synthetic datasets with different moving patterns.

6 Acknowledgement

Research supported by the Intelligence Advanced Research Projects Activity (IARPA) via the Department of Interior/Interior Business Center (DOI/IBC) contract number 140D0423C0033. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes, notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA or the U.S. Government.

References

[1] P. Bholowalia and A. Kumar. Ebk-means: A clustering technique based on elbow method and k-means in wsn. *IJCA*, 2014.

[2] J. Bian and et al. A survey on trajectory clustering analysis. *arXiv preprint arXiv:1802.06971*, 2018.

[3] D. Cai and et al. Locally consistent concept factorization for document clustering. *TKDE*, 2010.

[4] K. Cho and et al. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[5] D. J. Daley and et al. *An introduction to the theory of point processes: volume I: elementary theory and methods*. Springer, 2003.

[6] P. J. Diggle. Spatio-temporal point processes: methods and applications. *MSAP*, 2006.

[7] N. Dilokthanakul and et al. Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

[8] H. Hu and et al. Clustering human mobility with multiple spaces. In *Big Data*. IEEE, 2022.

[9] J. Jia and A. R. Benson. Neural jump stochastic differential equations. *NeurIPS*, 2019.

[10] J.-G. Lee and et al. Traclass: trajectory classification using hierarchical region-based and trajectory-based clustering. *PVLDB*, 2008.

[11] J.-G. Lee, J. Han, and K.-Y. Whang. Trajectory clustering: a partition-and-group framework. In *SIGMOD*, 2007.

[12] H. Lin and et al. Generating realistic and representative trajectories with mobility behavior clustering. In *SIGSPATIAL*, 2023.

[13] X. Olive and et al. Deep trajectory clustering with autoencoders. In *ICRAT*, 2020.

[14] F. Petitjean and et al. A global averaging method for dynamic time warping, with applications to clustering. *Pattern recognition*, 2011.

[15] D. Rezende and S. Mohamed. Variational inference with normalizing flows. In *ICML*. PMLR, 2015.

[16] F. P. Schoenberg and et al. Point processes, spatial-temporal. *Encyclopedia of environmetrics*, 2002.

[17] E. Schubert and et al. Dbscan revisited: why and how you should use dbscan. *TODS*, 2017.

[18] K. R. Shahapure and C. Nicholas. Cluster quality analysis using silhouette score. In *DSAA*. IEEE, 2020.

[19] T. Tanaka. A theory of mean field approximation. *NeurIPS*, 1998.

[20] N. X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *ICML*, 2009.

[21] H. Wei and et al. How do we move: Modeling human movement with system dynamics. *AAAI*, 2021.

[22] H. Xue and et al. Mobtcast: Leveraging auxiliary trajectory forecasting for human mobility prediction. *Neurips*, 2021.

[23] D. Yang and et al. Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns. *IEEE TSMC: Systems*, 2014.

[24] Y. Yuan and et al. Activity trajectory generation via modeling spatiotemporal dynamics. In *KDD*, 2022.

[25] M. Yue and et al. Detect: Deep trajectory clustering for mobility-behavior analysis. *Big Data*, 2019.

[26] M. Yue and et al. Vambc: A variational approach for mobility behavior clustering. *ECML-PKDD*, 2021.

[27] M. Zhang and et al. Csgan: Modality-aware trajectory generation via clustering-based sequence gan. In *MDM*. IEEE, 2023.

[28] Y. Zhang and et al. Learning mixture of neural temporal point processes for multi-dimensional event sequence clustering. In *IJCAI*, 2022.

[29] S. Zuo and et al. Transformer hawkes process. In *ICML*. PMLR, 2020.