Synthetic Trajectory Generation via Clustering-Based Semi-supervised Generative Adversarial Networks

by

Minxing Zhang

Li Xiong
Advisor

Computer Science

2022

Synthetic Trajectory Generation via Clustering-Based Semi-supervised Generative Adversarial Networks

By

Minxing Zhang

Li Xiong

Advisor

Abstract

Synthetic Trajectory Generation via Clustering-Based Semi-supervised Generative Adversarial Networks
By Minxing Zhang

Analyzing human mobility data and gaining insights from it is crucial for city planning and epidemic modeling. Synthetic trajectory generation is the task of generating large-scale fake trajectories that mimic the real ones by preserving all essential properties. As one of the fundamental problems in geoscience, it plays a vital role in studying population flow. Analyzing individual movements helps understand the traffic or public transportation system and may be used to predict the future position of a moving object. However, the limited open real-life human mobility data with complicated properties invalidates existing approaches. Moreover, effectively capturing the modality patterns (moving purpose, transportation mode, etc.) of the real-life trajectories and generating synthetic trajectories with these modality patterns preserved is also a critical issue. Third, there is a short of a systematic way to measure whether the transitional information from one location to another has been effectively captured in the generated trajectories. Given a user's incomplete sequence of visits, the existing generation model has yet to be tasked with predicting the following few locations. To address these challenges, we propose a **Clustering-based Semi-supervised Generative Adversarial Network (CS-GAN)** that, based on limited actual trajectories reported by users, can generate synthetic trajectories which mimic the real ones by preserving all the essential properties. Our proposed model leverages the idea of clustering and semi-supervised GANs to capture real-life modality patterns. Moreover, we develop a novel transitional probability-related metric to measure whether the synthetic trajectories capture the transitional information. We also conducted an ablation study to verify the effectiveness of our proposed generation model in predicting the possible subsequent few visits given an incomplete sequence of visits. Extensive experiments have been conducted on real-world datasets and demonstrated our model's superiority in performance over state-of-the-art models.

Synthetic Trajectory Generation via Clustering-Based Semi-supervised Generative Adversarial Networks

By

Minxing Zhang

Li Xiong

Advisor

An abstract of
a thesis submitted to the Faculty of
Emory College of Arts and Sciences of Emory University
in partial fulfillment of the requirements for the degree of
Bachelor of Science with Honors

Computer Science

2022

Acknowledgments

I would like to thank my research advisor, Professor Li Xiong, and my committee members: Professor Jinho Choi, Professor Kevin McAlister, and Professor Dwight Duffus. I would also like to thank Professor Cyrus Shahabi, University of Southern California, Professor Yang Cao, Hokkaido University, and Ph.D. student Haowen Lin, University of Southern California.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Analyzing human mobility data and gaining insights from it is crucial for city planning and epidemic modeling [1]. As one of the fundamental problems in geoscience, mobility simulation plays a vital role in studying population flow. Analyzing individual movements helps understand the traffic or public transportation system and may be used to predict the future position of a moving object. Recently, the worldwide outbreak of COVID-19 further stimulated the demand for mobility simulation [6]. The trajectory data records individuals' interaction with each other, which is a critical factor in preventing the spread of disease at an early stage. Furthermore, mobility data also has unique commercial values. For example, recommendation systems rely on population flow to identify where to place the advertisement. Considering the huge potential of mobility patterns, trajectory simulation has aroused wide attention recently.

However, mobility simulation advances face several significant limitations: 1) **Data limitation.** Training an effective synthetic trajectory generation model typically necessitates the availability of large-scale real-life trajectories reported by users. However, access to large-scale, high-quality mobility trajectory data is obstructed due to privacy issues and commercial concerns [2]. 2) **Preservation of real-life modality patterns.**

Real-world trajectories always consist of various modalities with semantics meanings; for instance, in terms of moving purpose modalities, real-life trajectories may consist of shopping, going to work, going back home, etc. Regarding transportation mode modalities, real-life trajectories may include walking, running, biking, driving, etc. Yet current research cannot generate synthetic trajectories that preserve such modality patterns. 3) **Systematic metric to measure the transitional information.** A single human daily trajectory consists of multiple transitions. Effectively capturing the transitional probability from one location to another from the actual trajectories and generating synthetic trajectories that preserve similar transitional patterns contribute significantly to traffic forecasting, epidemic modeling, urban planning, etc. However, recent advancements fail to develop a systematic metric to measure the transitional probability from one location to another. 4) **Predicting the following possible location based on an incomplete sequence of visits.** Developing an effective generation model to generate a group of trajectories that preserves all the essential properties reflects the generation model's power on a global scale. However, given an incomplete sequence of an individual user's trajectory on a local scale, whether the aforementioned generation model can effectively predict the possible subsequent few visits based on the existing incomplete ones is still a critical problem.

Therefore, simulating realistic mobile trajectories has become a fundamental topic of recent research. To address these challenges, we propose a **Clustering-based Semi-supervised Generative Adversarial Network (CS-GAN)** that, based on limited actual trajectories reported by users, can generate synthetic trajectories which mimic the real ones by preserving all the essential properties. Our proposed model leverages the idea of clustering and semi-supervised learning to capture real-life modality patterns. Moreover, we develop a novel transitional probability-related metric to measure whether the synthetic trajectories capture the transitional information. We also conducted an ablation study to verify the effectiveness of our proposed generation model in predicting the possible subsequent few visits given an incomplete sequence of visits. In short, the key contributions of this paper are summarized as follows:

1. **Developing a Clustering-based Semi-supervised Generative Adversarial Network (CS-GAN).** The idea of clustering is leveraged to obtain modality labels for each actual trajectory. By assigning the generated trajectories to the $\{k+1\}$-st cluster, we extend the semi-supervised GANs to generate synthetic trajectories with modality information preserved. The popular recurrent neural networks (RNNs) serve as the generation and discrimination backbone with the addition of reinforcement learning (policy-gradient) and Monte Carlo search with a roll-out policy to enhance the power of our proposed model.

2. **Proposing a novel transitional probability-related metric to measure whether the synthetic trajectories capture the transitional information.** It builds the transitional Origin-Destination Matrix for both the actual trajectories and the generated ones, takes the difference, and leverages the Frobenius Norm to measure how well the transitional information is captured.

3. **Designing a next location prediction task to evaluate the ability to capture the incomplete sequence information and extend it to predict the next visit on an individual trajectory-level scale.** We leverage our proposed generation model to conduct the next location prediction task: given an incomplete sequence of visits, what is the accuracy of successfully predicting the following few locations?

4. **Conducting comprehensive experimental analysis to validate the effectiveness of the proposed model.** Extensive experiments on two real-world

datasets, the GeoLife Dataset, and the PeopleFlow Dataset, demonstrate that our proposed framework achieves superior results in synthetic trajectory generation.

The rest of the paper is organized as follows. We first report the related work in Chapter 2. Then, we formulate the problem of synthetic trajectory generation in Chapter 3. Next, we present our **Clustering-based Semi-supervised Generative Adversarial Network (CS-GAN)** in Chapter 4 and evaluate the effectiveness of our model in Chapter 5. Finally, we conclude the paper in Chapter 6.

Chapter 2

Related Work

2.1 Generative Adversarial Networks

To simulate the fine-grained movement of a large number of individuals in daily life, model-based methods are proposed to capture the regularity of human mobility behaviors [3]. Another research stream proposes model-free methods for trajectory-generating and privacy-preserving with the generative adversarial network (GAN). [8] proposes a non-parametric generative model for location trajectories that can capture high-order geographic and semantic features of human mobility. The generator and the discriminator are Convolutional Neural Network (CNN) layers. The generated images are transformed into a sequence of trajectories during the generation process, and the location information is embedded into the images for the discriminator. [10] presents an end-to-end LSTM-TrajGAN model to generate privacy-preserving synthetic trajectory data for data sharing and publication. [2] leverages the self-attention networks as the backbone of the generator, utilizes the prior knowledge of human mobility patterns, and considers the effects of the urban structure during the generation process to integrate the domain knowledge of human mobility regularity utilized in the model-based methods. Instead of generating a discrete sequence of visits, a DeltaGAN [12] framework is proposed to generate a continuous sequence of visits to better capture temporal irregularity in human mobility behaviors by leveraging the idea of the spatiotemporal point process. Based on the idea of reinforcement learning, the SeqGAN [13] model becomes one of the state-of-the-art strategies for synthetic trajectory generation by treating the output of the discriminator as a reward sent back to the generator.

2.2 Semi-supervised Learning

Semi-supervised learning [15] is a type of machine learning that lies between supervised and unsupervised learning, which refers to leveraging a small portion of labeled data with the rest of the training data unlabeled to train a model.

Moreover, for the general generation task, semi-supervised GAN [7] gains popularity by treating the discriminator network as a multi-class classifier. During the training phase, the generator and the discriminator are trained on a dataset with inputs belonging to one of the k classes. Then, the discriminator is asked to predict which of the $k+1$ classes the input belongs to, where an extra $\{k+1\}$-st fake class is added to correspond to the outputs of G.

2.3 Reinforcement Learning

Under the umbrella of machine learning, reinforcement learning (RL) [5] considers how an agent can take actions in an environment to maximize the reward it receives. RL does not need labeled data as supervised learning requires. Instead, basic reinforcement learning is modeled as a Markov decision process (MDP) [9] with a set of environment and agent states, S; a set of actions, A, of the agent, the transition probability of transitioning from state s to the next state s' given the current action a, and the immediate reward obtained by the agent after transitioning from state s to state s' by taking action a.

Chapter 3

Problem Setting

3.1 Normal Synthetic Trajectories Generation

People's mobility patterns are reflected by the aggregation of individual spatiotemporal trajectories. A single trajectory Y can be defined as a list of visiting records $Y = [y_1, y_2, y_3, ..., y_i, ..., y_n]$, where y_i denotes the i-th visit of the trajectory, which is a ordered pair $(time_i, location_i)$, $time_i$ denotes the timestamp of the i-th visit, $location_i$ denotes the user's location of the i-th visit, represented by the region's Grid ID.

Thus, the problem of synthetic trajectory generation can be defined as:

Definition 1. *Given a set of trajectories collected from the real world, train a θ-parameterized generator G_θ to generate a set of synthetic trajectories that mimic the real-world trajectories by preserving all the essential properties. The generation of each synthetic trajectory $Y = [y_1, y_2, y_3, ..., y_{i-1}, y_i, ..., y_n]$ is a sequential decision process: the generation of the visit y_i is determined by the multiplication of the probability of each previously generated visits $[y_1, y_2, y_3, ..., y_{i-1}]$, which can be expressed as:*

$$p_\theta(Y) = \prod_{i=1}^{n} p_\theta(y_i | y_1, y_2, y_3, ..., y_{i-1}) \tag{3.1}$$

where p_θ denotes the probability distribution of the generator.

Besides the generator, we also train a ϕ-parameterized discriminator D_ϕ to play a "Two Player Game": the generator G_θ and discriminator D_ϕ, are trained together. The generator generates a batch of trajectories, and these, along with real trajectories, are provided to the discriminator and classified as real or fake. The generator is trained to fool the discriminator in terms of not being able to distinguish the generated trajectories from the real ones, while the discriminator is trained to classify the real trajectories as real or generated trajectories as fake. The discriminator is updated to get better at discriminating real and generated trajectories in the next round, and the generator is also updated based on how well, or not, the generated trajectories fool the discriminator. In short, the optimization objective can be expressed as:

$$\min_{G_\theta} \max_{D_\phi} E_{Y\sim p_d(Y)}[log(D_\phi(Y))] + E_{Y\sim G_\theta(Y)}[log(1 - D_\phi(Y))] \tag{3.2}$$

where p_d denotes the probability distribution of the real trajectories and Y denotes an individual trajectory.

3.2 Synthetic Trajectories Generation with Modality Patterns Preserved

Real-world trajectories always consist of various modalities with semantics meanings, either distinguished by their moving purposes, such as shopping trajectories, trajectories of going back home, trajectories of going to school, etc., or transportation modes, such as walking, biking, running, driving, etc. How to generate synthetic trajectories that mimic the real ones and simultaneously preserve the modality patterns has not been clearly specified by the current trajectories generation problem definition, as specified in Definition 1, and thus there is no potent approach to address them.

Thus, to address the aforementioned problem, we first adjust the problem definition of synthetic trajectory generation:

Definition 2. *Given a set of trajectories collected from the real world, train a θ-parameterized generator G_θ to generate a set of synthetic trajectories that mimic the real-world trajectories by preserving all the essential properties and preserving the real-world modality patterns. The generation of each synthetic trajectory $Y = [y_1, y_2, y_3, ..., y_{i-1}, y_i, ..., y_n]$ is a sequential decision process: the generation of the visit y_i is determined by the multiplication of the probability of each previously generated visits $[y_1, y_2, y_3, ..., y_{i-1}]$, which can be expressed as:*

$$p_\theta(Y) = \prod_{i=1}^{n} p_\theta(y_i | y_1, y_2, y_3, ..., y_{i-1}) \tag{3.3}$$

where p_θ denotes the probability distribution of the generator.

The generation of the set of synthetic trajectories that preserve the real-world modality patterns can be treated as: given a modality $m \in M$, the set of all modalities, minimizing the proportion of trajectories belonging to modality m between the synthetic ones and the real ones, which can be express as:

$$\min \sum_{m \in M} Proportion_{real}(m) - Proportion_{gen}(m) \tag{3.4}$$

where $Proportion_{real}(m)$ denotes the proportion of trajectories in the real ones belonging to modality m and $Proportion_{gen}(m)$ denotes the proportion of trajectories in the generated ones belonging to modality m.

Chapter 4

Proposed Framework

4.1 Clustering

Given an individual trajectory $Y = [y_1, y_2, y_3, ..., y_i, ..., y_n]$ as a sequence of visits, each visit y_i is often associated with several modalities with semantic meanings; for instance, the transportation mode (walking, running, car, airplane, etc.) or the moving purpose (shopping, going to school, going back home, etc.). Thus, given an input trajectory Y, we can represent it via two regimes: 1) A global feature F_G denoting the general modality, and 2) a group of local features $F_L = [f_1, f_2, f_3, ..., f_i, ..., f_n]$ with f_i denoting the modality of the i-th visit. In this paper, we explore the global representation of the trajectory.

However, it is not always the case that the modality information for each trajectory is provided. How to develop a model that, in the absence of modality information, effectively generates synthetic trajectories that preserve the modality patterns of the real ones, as elaborated in Section 4.2, becomes a critical problem and has yet to be well addressed. Our proposed model can effectively address the problem by leveraging a clustering-based technique based on the semantics feature of each trajectory, i.e., the average speed, to capture the modality information without ground-truth features.

4.1.1 Clustering in the absence of Modality Features

Given that the modality information for each real trajectory is not provided, we believe that average speed is a potent attribute to capture the modality pattern of each trajectory on a global scale. Thus, we propose a speed-computation algorithm, illustrated in Algorithm 1, to obtain the average speed of each real trajectory. Then, we leverage the K-means Clustering algorithm to cluster the real trajectories based on average speed and obtain the cluster label for each trajectory.

Algorithm 1: Speed-computation Algorithm

```
Data: A trajectory $\mathbf{Y}$, bounding box $bbox$, horizontal resolution $horizontal\_n$,
      vertical resolution $vertical\_n$, time resolution $time\_res$
Result: Computed speed of trajectory $Y$
Compute a reference matrix to find the centered point (in latitude and
longitude) given a region's Grid ID based on $bbox$, $horizontal\_n$, and
$vertical\_n$;
Initialize the average speed to be 0 ;
Initialize the count of transitions ;
for $n = 1 : N - 1$ do
    Obtain the current visit's Region ID $r_n$;
    Obtain the next visit's Region ID $r_{n+1}$;
    Find the centered point $c_n$ of $r_n$;
    Find the centered point $c_{n+1}$ of $r_{n+1}$;
    if $c_n$ != $c_{n+1}$ and both $c_n$ and $c_{n+1}$ are not unknown then
        Compute the haversine distance $dist$ between $c_n$ and $c_{n+1}$;
        average speed ← $dist/time\_res$+average speed;
        count ← count+1
    end
end
average speed ← average speed/count;
return average speed
```

4.2 Generator

We leverage recurrent neural networks (RNNs) as the backbone of our generator G_θ to generate synthetic trajectories. More specifically, to generate a trajectory, the generator first generates the starting location by either randomly selecting from the entire probability distribution of the locations or simply selecting the first location. Then, to generate the next location loc_i based on the previously generated locations $loc_1, loc_2, loc_3, ..., loc_{i-1}$, our proposed generator consists of an embedding function $e(\cdot)$ to map the sequence of previously generated locations into embedding representations, a mapping function $g(\cdot)$ to map the embedded sequence into hidden states, and finally a predicting function $z(\cdot)$ to map the hidden states to the probability distribution of location, which can be written as:

$$\begin{aligned} p(loc_i|loc_1, loc_2, loc_3, ..., loc_{i-1}) &= G_\theta(loc_1, loc_2, loc_3, ..., loc_{i-1}) \\ &= G(z(g(e(loc_1, loc_2, loc_3, ..., loc_{i-1})))). \end{aligned} \tag{4.1}$$

More specifically, the function $e(\cdot)$ takes the previously generated locations loc_1, loc_2, loc_3, ..., loc_{i-1} and outputs the embedded representations, which can be written as:

$$\overline{loc}_1, \overline{loc}_2, \overline{loc}_3, ..., \overline{loc}_{i-1} = e(loc_1, loc_2, loc_3, ..., loc_{i-1}). \tag{4.2}$$

Then, the proposed function $g(\cdot)$, which is the gated recurrent unit (GRU) framework, maps the embedding representations of the previously generated locations to a sequence of hidden states $h_1, h_2, h_3, ..., h_{i-1}$, which can be written as:

$$pred(loc_i), h_{i-1} = g(\overline{loc}_1, \overline{loc}_2, \overline{loc}_3, ..., \overline{loc}_{i-1}). \tag{4.3}$$

Finally, the predicting function $z(\cdot)$ maps the $pred(loc_i)$ into the output probability distribution of locations with a soft-max output later to determine the most probable next location, which can be expressed as:

$$p(loc_i|loc_1, loc_2, loc_3, ..., loc_{i-1}) = z(pred(loc_i)). \tag{4.4}$$

4.3 Discriminator

4.3.1 $k + 1$ Cluster Labels

As elaborated in section 4.1, we leverage the clustering technique to split the entire real-world trajectories into k clusters, denoting k distinct modalities, and thus are able to obtain the cluster label for each real-world trajectory. In this way, we have k cluster labels.

For the generated trajectories from the generator, we assign the cluster label "$k + 1$" denoting the fake class.

Thus, we have in total $k + 1$ cluster labels, where the first k clusters denote the k modalities in the real trajectories and the $\{k + 1\}$-st cluster denotes the fake class.

4.3.2 Semi-supervised Learning-based Discriminator

Leveraging the idea of semi-supervised learning, our proposed discriminator D_ϕ functions as a multi-class classifier. Given a trajectory, which is either from the group of generated trajectories from the proposed generator or from the real ones, as input, our proposed discriminator aims to distinguish: 1) whether it is real or fake and 2) given it is real, the specific cluster it belongs to. More specifically, given the trajectory generated from the generator, our discriminator aims to classify it into the $\{k + 1\}$-st fake cluster; given the trajectory from the real ones, our discriminator aims to classify it into the specific modality cluster it belongs to among the k real clusters.

To capture the entire sequence information, we first leverage bidirectional recurrent neural networks (RNNs) to comprehensively evaluate the input trajectory, and then followed by dense layers to output the probability of being classified into each cluster, which can be written as:

$$\begin{aligned} p(cluster_Y) = D_\phi(Y) &= D(z_d(g_d(e_d(Y)))) \\ &= D(z_d(g_d(e_d(\langle loc_1, loc_2, loc_3, ..., loc_n \rangle)))) \end{aligned} \tag{4.5}$$

where $p(cluster_Y)$ denotes the output probability distribution of $k+1$ clusters corresponding to input trajectory Y, $e_d(\cdot)$ denotes a embedding function, $g_d(\cdot)$ denotes a mapping function, and $z_d(\cdot)$ denotes a predicting function.

Given an entire input trajectory, similar to our proposed generator G_θ, our discriminator D_ϕ first consists of an embedding function $e_d(\cdot)$, which takes the entire locations of the input trajectory Y, $\langle loc_1, loc_2, loc_3, ..., loc_n \rangle$, and outputs the embedded representations, which can be written as:

$$\overline{Y} = \langle \overline{loc}_1, \overline{loc}_2, \overline{loc}_3, ..., \overline{loc}_n \rangle = e_d(Y) = e_d(\langle loc_1, loc_2, loc_3, ..., loc_n \rangle). \tag{4.6}$$

Then, our proposed discriminator leverages a function $g_d(\cdot)$, which is the gated recurrent unit (GRU) framework, and maps the embedding representations of the locations to the hidden state h_n, which can be written as:

$$\_, h_n = g(\overline{Y}) = g(\langle \overline{loc}_1, \overline{loc}_2, \overline{loc}_3, ..., \overline{loc}_n \rangle). \tag{4.7}$$

Finally, the predicting function $z(\cdot)$, which consists of a stack of U fully connected layers FC_u, where $u = 1, 2, ..., U$, followed by nonlinear activation function σ, maps the h_n into the output probability distribution of clusters to determine the most probable cluster the trajectory belongs to, which can be expressed as:

$$p(cluster_Y) = z(h_n). \tag{4.8}$$

4.4 Model Training

4.4.1 Reinforcement Learning-based Training

The normal training algorithm of GANs via gradient back-propagation does not perform well due to the discreteness of the output from the generator. To address the problem, we interpret the trajectories generation problem via reinforcement learning. More specifically, we treat our proposed generator as the agent, the group of current generated locations as the state, generating the next location based on the previously generated locations is the action, and the probability of "fooling" the discriminator is the reward. Despite the fact that the generator $G_\theta(y_i|y_1, y_2, y_3, ..., y_{i-1})$ is stochastic, given the current state s and the action a, the transition to the next state is deterministic:

$$\begin{cases} \delta^a_{s,s'} = 1 & \text{for the next state } s'; \\ \delta^a_{s,s''} = 0 & \text{for other states } s''. \end{cases}$$

Leveraging the idea of reinforcement learning, our proposed generator $G_\theta(y_i|y_1, y_2, y_3, ..., y_{i-1})$ to generate the next location based on the previously generated location aims to maximize its expected end reward:

$$J(\theta) = E[R_n|\theta, s_0] = \sum_{y_1 \in Y} G_\theta(y_1|s_0) \cdot T^{G_\theta}_{D_\phi}(s_0, y_1) \tag{4.9}$$

where R_n is the reward gained for the complete trajectory Y with n visits, $T^{G_\theta}_{D_\phi}(s_0, y_1)$ is the action-value function and thus represents the expected accumulative reward from starting from the state s_0 and taking the action y_1, following policy G_θ.

More specifically, the reward is based on the output of the discriminator. According to Equation 4.5, the output of the discriminator is the probability distribution of $k+1$ clusters (k real clusters and the $\{k+1\}$-st fake cluster) given the input trajectory Y. Thus, the summation of the probability for trajectory Y of being classified into the k real clusters represents the probability of "fooling" the discriminator and thus should be treated as the reward, which can be written as:

$$R_{D\phi}(Y) = \sum_{c \in C} p(cluster_Y(c)) \tag{4.10}$$

where C denotes the group of k real clusters and $p(cluster_Y(c))$ denotes the probability of the trajectory Y being classified into cluster c.

However, given the current state $y_1, y_2, y_3, ..., y_{i-1}$ and the action to generate the next location y_i, where $i < n$, which is an incomplete sequence, how to connect the aforementioned $R_{D\phi}(Y)$ to the current action-value function $T_{D_\phi}^{G_\theta}(s = y_1, y_2, y_3, ..., y_{i-1}, a = y_t)$ becomes a problem. Given that the input trajectory Y to the discriminator to compute the reward must be a complete sequence of the trajectory of total n visits, we decide to leverage Monte Carlo search with a roll-out policy (we use G_θ) to generate the unknown last $n - i$ locations. More specifically, given the previously generated locations $y_1, y_2, y_3, ..., y_{i-1}$, the next location y_i is generated by a deterministic process based on current state s and action a; a group of next locations $y_{i+1}, y_{i+2}, y_{i+3}, ..., y_n$ is generated by the roll-out policy (generator G_θ) based on the current state. Leveraging the idea behind the Monte Carlo search with the roll-out policy, we can generate a complete sequence of trajectory for the discriminator to generate reward based on Equation 4.10.

Thus, given the task of generating the next location y_i based on the previous locations $y_1, y_2, y_3, ..., y_{i-1}$ and $i < n$, we first leverage the Monte Carlo search with a roll-out policy (G_θ) to generate the complete sequence of visits Y. Then, for $i = n$, we compute the reward and thus have:

$$T_{D_\phi}^{G_\theta}(s = \{y_1, y_2, y_3, ..., y_{i-1}\}, a = y_t) = R_{D\phi}(Y) = \sum_{c \in C} p(cluster_Y(c)). \tag{4.11}$$

Thus, we can compute the gradient of the generator's objective function with respect to θ from Equation 4.9:

$$\nabla_\theta J(\theta) = E_{\{y_1, y_2, y_3, ..., y_{i-1}\} \sim G_\theta} \left(\sum_{y_t \in Y} \nabla\theta G_\theta(y_t | \{y_1, y_2, y_3, ..., y_4\}) \cdot T_{D_\phi}^{G_\theta}(s = \{y_1, y_2, y_3, ..., y_{i-1}\}, a = y_t) \right) \tag{4.12}$$

In this way, we can update the generator's parameter θ:

$$\theta = \theta + \alpha \cdot \nabla_\theta J(\theta) \tag{4.13}$$

where α is the learning rate.

Then, as the objective of the discriminator is to minimize the multi-class classification loss by classifying the real trajectories as real and generated trajectories as fake, we have:

$$\min_\phi -E_{Y \sim p_d(Y)}[\log D_\phi(Y)] - E_{Y \sim G_\theta(Y)}[log(1 - D_\phi(Y))] \tag{4.14}$$

where $D_\phi(Y)$ denotes the total probability of the trajectory Y being classified into the k real classes, and $1 - D_\phi(Y)$ denotes the probability of the trajectory Y being classified into the $\{k+1\}$-st fake class.

Thus, the objective function of the discriminator can be further written as:

$$\min_\phi -E_{Y \sim p_d(Y)}[\log(\sum_{c \in C} p(cluster_Y(c)))] - E_{Y \sim G_\theta(Y)}[log(p(cluster_Y(k+1)))] \tag{4.15}$$

where C denotes the group of k real clusters, $p(cluster_Y(c)$ denotes the probability of the discriminator to classify the trajectory Y into cluster c, and $p(cluster_Y(k+1)$ denotes the probability of the discriminator to classify the trajectory Y into the $\{k+1\}$-st fake class.

4.4.2 Model Pre-training

Due to the complicated nature of human mobility data, training a powerful generator with a large number of parameters that can generate synthetic trajectories that mimic the real ones in a time-efficient manner is hard. Thus, to accelerate the training process and improve the overall model's performance, we perform model pre-training on both the generator and the discriminator.

To pre-train the generator, we leverage the idea behind maximum likelihood estimation (MLE) and pre-train the generator on the real trajectories. We aim to minimize the negative log-likelihood loss between the generated ones and the real ones.

To pre-train the discriminator, we mix the real trajectories with the generated trajectories in an equal proportion. Then, we pre-train the discriminator to minimize the negative log-likelihood loss between the predicted cluster labels and the ground-truth cluster labels (one of the k clusters for the real trajectory and the $\{k+1\}$-st cluster for the generated trajectory).

4.4.3 Algorithm Pseudocode

Based on reinforcement learning-based training and model pre-training, we have our entire CS-GAN algorithm illustrated in 2.

Algorithm 2: CS-GAN Framework for Synthetic Trajectory Generation

```
Data: Real group of trajectories $\mathbf{Y}$, noise distribution $P_{\mathbf{Z}}$, number of classes $k$,
      batch size $b$, total number of iterations $T$, number of iterations $T_G$ to
      train the generator, number of iterations $T_D$ to train the discriminator
Result: A well-trained generator $G_\theta$ for synthetic trajectory generation
Initialize parameters of the generator $G_\theta$ and the discriminator $D_\phi$;
Perform KM-means clustering on the real group of trajectories $\mathbf{Y}$ based on $k$
number of classes and thus obtain $k$ centroids;
Conduct pre-training of the generator $G_\theta$ via MLE using $\mathbf{Y}$ ;
Conduct pre-training of the discriminator $D_\phi$ via minimizing the negative log
likelihood loss ;
for $t = 1 : T$ do
    for $t = 1 : T_G$ do
        Use the generator to generate $b$ synthetic trajectories $\{G_\theta(\mathbf{z}_i)\}_{i=1}^{b}$ from
         $P_{\mathbf{Z}}$;
        Assign clustering labels to "fake" (the $\{k+1\}$-st cluster) of the $b$
         generated trajectories;
        Compute the reward of the $b$ generated trajectories via Equation 4.11;
        Update $\theta$ via policy gradient Equation 4.13 ;
    end
    for $t = 1 : T_D$ do
        Sample $b$ real trajectories $\{Y_i\}_{i=1}^{b}$ from $\mathbf{Y}$;
        Obtain clustering labels with respect to the $k$ centroids of the $b$ sampled
         trajectories;
        Use the generator to generate $b$ synthetic trajectories $\{G_\theta(\mathbf{z}_i)\}_{i=1}^{b}$ from
         $P_{\mathbf{Z}}$;
        Assign clustering labels to "fake" (the $\{k+1\}$-st cluster) of the $b$
         generated trajectories;
        Update $\phi$ via Equation 4.15 ;          /* Update the parameters of
         Discriminator w.r.t the NLL of Discriminator's outputs on
         the combined minibatch of size $2b$ */
    end
end
```

Chapter 5

Evaluation

For this study, we utilize real-world datasets to evaluate our proposed model for answering the following questions:

Q1. What are the similarities and differences between the GeoLife Dataset and the PeopleFlow Dataset in terms of both the individual trajectory-level information: trajectory speed, cumulative distance, number of distinct visits, and modality pattern-level (clustering-level) information?

Q2. How effective is our proposed CS-GAN model in terms of generating synthetic trajectories compared with the state-of-the-art approach?

Q3. How would the resulting similarities and differences between the datasets affect the proposed CS-GAN's and the state-of-the-art model's performance?

Q4. How does our proposed CS-GAN model perform on the next location prediction task compared to the state-of-the-art approach?

5.1 Experimental Setup

5.1.1 Data.

We experiment on two real-world datasets (open GeoLife Dataset [14] and semi-open PeopleFlow Dataset [11]) to verify the effectiveness of our proposed model.

1. GeoLife Dataset: This GPS trajectory dataset was collected in the (Microsoft Research Asia) Geolife project by 182 users over three years (from April 2007 to August 2012). This dataset contains 17,621 trajectories with a distance of about 1.2 million kilometers and a total duration of 48,000+ hours.

2. PeopleFlow Dataset: This data is based on 2008 Tokyo Metropolitan Area PT Data (provided by Tokyo Metropolitan Circle Transportation Planning Council) and is lent by the University of Tokyo CSIS.

5.1.2 Comparison Methods.

We compare our proposed CS-GAN model with different variations of the state-of-the-art SeqGAN [13] model, which extends the idea of reinforcement learning and Monte Carlo search to generate a discrete sequence of trajectory. The SeqGAN model is the cornerstone of our proposed CS-GAN model: we extend the baseline SeqGAN model with clustering and semi-supervised GANs. Our intuition is that by leveraging the idea of clustering and semi-supervised GANs, our proposed CS-GAN model can generate more realistic synthetic trajectories by preserving both the individual trajectory-level information and the modality information compared with the existing method, and thus we select the SeqGAN model and its variations as baselines. Thus, the SeqGAN model with its different variations are the comparison methods:

1. SeqGAN: we train a SeqGAN model on the entire real trajectories.

Table 5.1: Model Hyper-parameters

	Model Hyper-parameters			
	Embedding Dimension	Hidden Dimension	Bidirectional	Pre-training Epoch
Generator	32	32	False	150
Discriminator	64	64	True	75

2. Pre-clustering + SeqGAN: we conduct clustering on the real trajectories and train an individual SeqGAN model on each cluster.

3. SeqGAN + Post-clustering: we train a SeqGAN model on the entire real trajectories and then conduct clustering on the generated trajectories.

5.1.3 Implementation Details.

As our CS-GAN model leverages the idea of clustering to split the real trajectories into k modalities, we extend the elbow method to select the optimal k for each dataset: k is set to 6 for the PeopleFlow Dataset and 4 for the GeoLife Dataset.

Batch size is set to 32, dropout is set to 0.2, adversarial training epoch is set to 75, the learning rate is set to $1e^{-2}$, and other generator/discriminators' hyper-parameters are indicated in Table 5.1.

5.1.4 Evaluation Metrics.

We define 8 metrics to evaluate the quality of the synthetic trajectories in different aspects. 4 of them are defined in previous works [4, 8], which measure the individual trajectory-level information:

- $P(r)$: Probability of a trajectory visiting location r.
- $P(r,t)$: Probability of a trajectory visiting location r at time t.
- $P(d)$: Probability of the accumulated distance of a trajectory equaling d.
- $P(v)$: Probability of the number of distinct visits of a trajectory being v.

For each probability metric, we compute the Jensen-Shannon Divergence between the probability distribution of the real trajectories and that of trajectories generated by our proposed model and baselines, which can be written as:

$$JSD(X||Y) = H(\frac{X+Y}{2}) - \frac{H(X)+H(Y)}{2} \tag{5.1}$$

where X and Y are two probability distributions and H is the Shannon information.

Moreover, we define 4 new metrics to evaluate whether the transitional sequence information and the modality pattern (clustering) information are preserved in the synthetic trajectories, which :

- $P(r_1, r_2)$: Probability of a trajectory transitioning from location r_1 to location r_2. Given the entire Q regions, we build the Origin-Destination Matrix $OD \in \mathbb{R}^{Q*Q}$ for both real and synthetic trajectories, where the element corresponding to row i and column j of the Origin-Destination matrix denotes the transitional probability from location i to location j. Then we take the Frobenius norm of the difference between the two OD matrices, which can be written as:

  $$P(r_1, r_2) = ||OD_{real} - OD_{fake}||_F = \sqrt{\sum_{i=1}^{Q}\sum_{j=1}^{Q} |OD_{real}(i,j) - OD_{fake}(i,j)|^2} \tag{5.2}$$

  where $OD_{real}(i,j)$ denotes the element of the matrix OD_{real} corresponding to row i and column j.

- $P(c_i^0)$: Proportion of trajectories within each cluster c_i. We cluster the real trajectories and, based on the resulting real centroids, cluster trajectories generated by our proposed model and baselines. In this way, we have a distribution of the proportion of trajectories in each cluster corresponding to the real trajectories and another distribution of the proportion of trajectories in each cluster corresponding to the generated ones. Then, we compute the JS divergence between the two distributions based on Equation 5.1.

- $D, P(c_i^1)$: D denotes the accumulated absolute distance between real and generated centroids. $P(c_i^1)$ denotes the proportion of trajectories within each cluster-pair c_i^1. We cluster the real trajectories and trajectories generated by our proposed model and baselines separately and pair the real centroids and the generated centroids based on distance. Then, given two distributions, similar to $P(c_i^0)$, we compute the JS divergence between the two distributions based on Equation 5.1.

5.2 Comparison between GeoLife and Peopleflow Dataset

Generally, based on Figures 5.1a, 5.1b, 5.2a, 5.2b, and Table 5.2, we can conclude that in terms of variety of the modalities, GeoLife_15min is less diverse while PeopleFlow_15min is more diverse. The pattern can be reflected by the following individual analysis of essential attributes: average speed, cumulative distance, distinct visit, and clustering inertia.

5.2.1 Average Speed

As indicated in Figure 5.1a and Table 5.2, the average speed of GeoLife_15min is 5.324 km/h with a standard deviation of 5.744. In comparison, the average speed of PeopleFlow_15min is 13.592 km/h with the most significant standard deviation of 8.303, which denotes that it consists of resting, walking, running, biking, and also driving trajectories. This pattern is reflected in the kernel density plot, where the blue curve (PeopleFlow_15min) is flat and more equally distributed from 0 to 40 km/h.

Thus, in terms of average speed, we can conclude that PeopleFlow_15min has the most diverse speed distribution and modality patterns.

5.2.2 Cumulative Distance

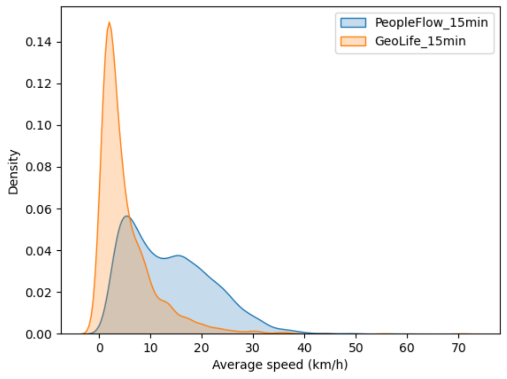
The same pattern is reflected in the cumulative distance attribute. GeoLife_15min is less diverse, and most of the trajectories have a speed of 8km/h. According to the table, the average distance of 8.401 is relatively low, with a lower standard deviation of 9.340. PeopleFlow_15min has a more significant average cumulative distance of 27.662 km and a larger standard deviation of 26.494, which means most of the trajectories in PeopleFlow_15min has a relatively large travel distance instead of staying in a single region. The pattern is also reflected in the kernel density plot, where the blue curve is flatter and lies to the right of the orange curves.

5.2.3 Distinct Visit

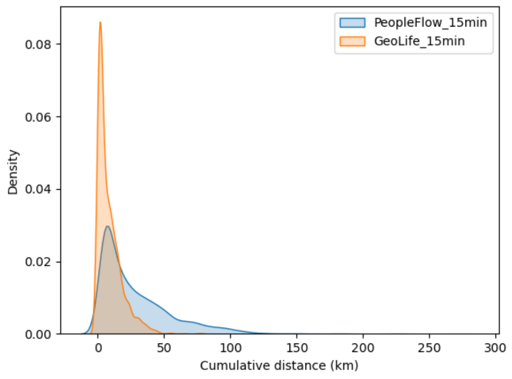
In terms of distinct visits, we can see from Figure 5.2a that both PeopleFlow_15min and GeoLife_15min have roughly the same distribution; the orange curve (GeoLife_15min) overlaps with the blue curve (PeopleFlow_15min), and this pattern is also verified from the Table 5.2, where the average distinct visit and standard deviation are roughly the same.

5.2.4 Clustering Inertia

As verified by the important trajectory attributes: speed, cumulative distance, and distinct visit, we can conclude that PeopleFlow_15min has more diverse modalities while GeoLife_15min is less diverse. To further verify the claim, we leverage the elbow method to plot the clustering inertia with respect to the number of clusters. Based

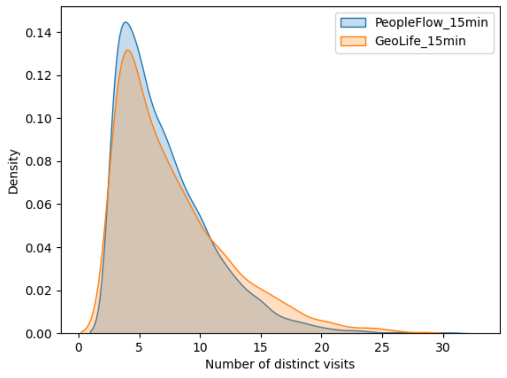


(a) Average Speed Distribution

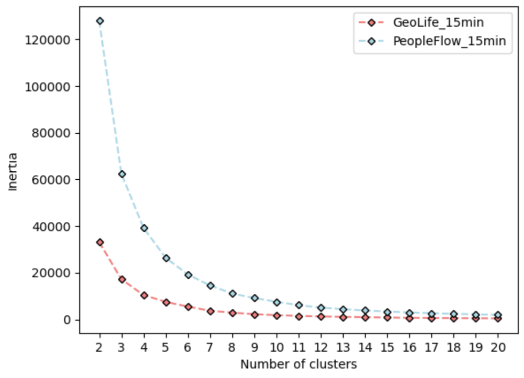


(b) Cumulative Distance Distribution

Figure 5.1: Dataset Comparison Visualization (Speed and Distance)



(a) Distinct Visits Distribution



(b) Elbow Curve (Clustering Inertia)

Figure 5.2: Dataset Comparison Visualization (Visits and Elbow Curve)

on Figure 5.2b, we can see that the convergence speed of the red curve, which denotes People_15min is slow while GeoLife_15min converges the fastest.

5.3 Effectiveness Comparison with the Baselines

We present the performance comparison of all baselines in Table 5.3. Based on the table, our proposed model performs consistently the best over all the metrics on both datasets.

Table 5.2: Comparison between GeoLife data and PeopleFlow data using different processing techniques

	Data Comparison		
Data	Average Speed (km/h)	Average Accumulated Distance (km)	Average Distinct Visits
PeopleFlow_15min	13.592±8.303	27.662±26.494	7.123±3.895
GeoLife_15min	5.324±5.744	8.401±9.340	7.713±4.558

5.3.1 Global Comparison

Regarding the general comparison between our proposed model and baselines, on People_15min, our model consistently outperforms the baselines over all the metrics. In particular, in terms of the individual trajectory-level metrics, our model excels over the baselines on average 21% in $P(r)$, 11% in $P(r,t)$, 40% in $P(d)$, and 27% in $P(v)$; in terms of the transitional probability metric, our model outperforms the baselines on average 49% in $P(r_1, r_2)$; in terms of the modality patterns (clustering) metrics, our model excels over the baselines on average 59% in $P(c_i^0)$, 57% in D, and 61% in $P(c_i^1)$.

Similarly, on GeoLife_15min, our model consistently outperforms the baselines over all the metrics. In particular, in terms of the individual trajectory-level metrics, our model excels over the baselines on average 30% in $P(r)$, 16% in $P(r,t)$, 25% in $P(d)$, and 1% in $P(v)$; in terms of the transitional probability metric, our model outperforms the baselines on average 48% in $P(r_1, r_2)$; in terms of the modality patterns (clustering) metrics, our model excels over the baselines on average 58% in $P(c_i^0)$, 66% in D, and 78% in $P(c_i^1)$.

5.3.2 Inner-cluster Comparison

Besides comparing the general performance by aggregating all the trajectories, we also conduct inner-cluster comparisons for both datasets. We compare the individual trajectory-level and transitional probability metrics within each cluster and find that our model excels over the baselines greatly.

On People_15min, our proposed model consistently outperforms the baselines over all the metrics for each cluster. For instance, in cluster 1, in terms of the individual trajectory-level metric, our model excels over the baselines on average 19% in $P(r)$, 15% in $P(r,t)$, 36% in $P(d)$, and 25% in $P(v)$. Furthermore, in terms of the transitional probability metric, our model outperforms the baselines on average 30% in $P(r_1, r_2)$.

Similarly, on GeoLife_15min, our proposed model consistently outperforms the baselines over all the metrics for each cluster. For instance, in cluster 0, in terms of the individual trajectory-level metric, our model excels over the baselines on average 29% in $P(r)$, 16% in $P(r,t)$, 57% in $P(d)$, and 44% in $P(v)$. Furthermore, in terms of the transitional probability metric, our model outperforms the baselines on average 62% in $P(r_1, r_2)$.

5.3.3 Finding

Based on Table 5.3 and the above analysis, we can see the superiority of our model in generating synthetic trajectories that preserve all the essential properties. In particular, our model is good at identifying and preserving the modality patterns (clustering information). The power in identifying clusters renders our model achieves the highest performance in the clustering-level metrics: for instance, in terms of the modality patterns (clustering) metrics on GeoLife_15min, our model excels over the baselines on average 58% in $P(c_i^0)$, 66% in D, and 78% in $P(c_i^1)$. Moreover, the successful identification and preservation of modality patterns enable our model to perform the best on the transitional probability metric, as the explanation is intuitive: given a trajectory with a specific modality, for instance, walking, the user cannot travel a significant distance and thus, there is merely a limited potential destination. Therefore, if the model can identify the modality successfully, the transitional information can be successfully preserved.

Moreover, we realize that the state-of-the-art SeqGAN model is not good at identifying clusters and thus cannot generate synthetic trajectories that preserve the modality patterns. As illustrated in Section 5.2, in terms of the variety of the modality patterns, GeoLife_15min is less diverse, while PeopleFlow_15min is more diverse. Then, as reflected in Table 5.3, the general SeqGAN model outperforms the "Pre-clustering + SeqGAN" model on a global scale, and the "SeqGAN + Post-clustering" model outperforms the "Pre-clustering + SeqGAN" in terms of the inner-cluster comparison when the dataset's variety is low (GeoLife_1hour). However, as the variety of the dataset's modality increases (from the GeoLife_15min, which is less diverse, and to the more diverse People_15min), the general SeqGAN model tends to perform less satisfying and is surpassed by the "Pre-clustering + SeqGAN" model on a global scale; similarly, in terms of the inner-cluster comparison, the "SeqGAN + Post-clustering" model is no longer better than the "Pre-clustering + SeqGAN" approach. As the dataset's modality patterns become more diverse, only externally conducting the clustering first for the SeqGAN to identify ("Pre-clustering + SeqGAN") can guarantee the performance of the SeqGAN model.

5.4 Next Location Prediction

5.4.1 Motivation

Section 5.3 successfully verifies that our proposed CS-GAN model consistently outperforms the baselines in generating a large number of synthetic trajectories by preserving both the individual trajectory-level information and the modality information. Our proposed model demonstrates power in generating trajectories on a global scale. However, whether our proposed model performs better in generating the next location given

Table 5.3: Comparison with baselines on GeoLife and PeopleFlow data with clustering based on **average speed**. The table shows the average statistics of 5 experiments. The best performance is in boldface. The second-best is underlined.

	PeopleFlow (2008 6:00 - 20:00 with 15 minutes time resolution)										
Cluster_ID	Centroid Speed	Proportion_traj	Method	P(r)	P(r,t)	P(r1, r2)	$P(c_i^0)$	D	$P(c_i^1)$	P(d)	P(v)
-	-	-	SeqGAN	0.378	0.437	$\underline{0.092}$	0.406	$\underline{29.694}$	$\underline{0.167}$	0.368	$\underline{0.275}$
-	-	-	Pre-clustering + SeqGAN	$\underline{0.344}$	$\underline{0.406}$	0.105	$\underline{0.311}$	31.046	0.194	$\underline{0.363}$	0.317
-	-	-	Our	**0.284**	**0.376**	**0.050**	**0.146**	**13.136**	**0.070**	**0.218**	**0.215**
Cluster_1	4.497	27%	SeqGAN + Post-clustering	0.460	0.482	0.160	-	-	-	0.576	0.597
			Pre-clustering + SeqGAN	$\underline{0.364}$	$\underline{0.408}$	$\underline{0.104}$	-	-	-	$\underline{0.548}$	$\underline{0.407}$
			Our	**0.334**	**0.379**	**0.092**	-	-	-	**0.360**	**0.374**
Cluster_3	9.141	21%	SeqGAN + Post-clustering	0.512	0.558	0.155	-	-	-	$\underline{0.411}$	0.347
			Pre-clustering + SeqGAN	$\underline{0.424}$	$\underline{0.475}$	$\underline{0.093}$	-	-	-	0.454	$\underline{0.339}$
			Our	**0.361**	**0.438**	**0.080**	-	-	-	**0.243**	**0.215**
Cluster_0	14.590	20%	SeqGAN + Post-clustering	0.553	0.614	$\underline{0.109}$	-	-	-	$\underline{0.382}$	$\underline{0.251}$
			Pre-clustering + SeqGAN	$\underline{0.492}$	$\underline{0.557}$	0.124	-	-	-	0.492	0.368
			Our	**0.433**	**0.521**	**0.072**	-	-	-	**0.264**	**0.162**
Cluster_4	19.924	17%	SeqGAN + Post-clustering	0.585	0.648	0.165	-	-	-	0.460	0.338
			Pre-clustering + SeqGAN	$\underline{0.511}$	$\underline{0.586}$	$\underline{0.129}$	-	-	-	$\underline{0.442}$	$\underline{0.265}$
			Our	**0.451**	**0.556**	**0.073**	-	-	-	**0.258**	**0.146**
Cluster_2	25.868	11%	SeqGAN + Post-clustering	0.564	0.639	$\underline{0.122}$	-	-	-	$\underline{0.432}$	$\underline{0.273}$
			Pre-clustering + SeqGAN	$\underline{0.540}$	$\underline{0.622}$	0.135	-	-	-	0.546	0.321
			Our	**0.471**	**0.587**	**0.087**	-	-	-	**0.325**	**0.181**
Cluster_5	34.627	4%	SeqGAN + Post-clustering	$\underline{0.576}$	$\underline{0.654}$	$\underline{0.114}$	-	-	-	$\underline{0.461}$	$\underline{0.220}$
			Pre-clustering + SeqGAN	0.628	0.673	0.267	-	-	-	0.716	0.396
			Our	**0.553**	**0.646**	**0.093**	-	-	-	**0.455**	**0.184**

	GeoLife (2008 6:00 - 20:00 with 15 minutes time resolution)										
Cluster_ID	Centroid Speed	Proportion_traj	Method	P(r)	P(r,t)	P(r1, r2)	$P(c_i^0)$	D	$P(c_i^1)$	P(d)	P(v)
-	-	-	SeqGAN	$\underline{0.407}$	$\underline{0.478}$	0.100	$\underline{0.162}$	$\underline{18.682}$	$\underline{0.220}$	$\underline{0.208}$	0.288
-	-	-	Pre-clustering + SeqGAN	0.506	0.562	0.082	0.402	83.441	0.298	0.313	**0.234**
-	-	-	Our	**0.319**	**0.439**	**0.047**	**0.147**	**17.128**	**0.058**	**0.195**	$\underline{0.258}$
Cluster_2	2.498	55%	SeqGAN + Post-clustering	$\underline{0.505}$	$\underline{0.555}$	0.191	-	-	-	$\underline{0.297}$	0.432
			Pre-clustering + SeqGAN	0.607	0.633	$\underline{0.135}$	-	-	-	0.496	**0.276**
			Our	**0.403**	**0.487**	**0.058**	-	-	-	**0.237**	$\underline{0.402}$
Cluster_0	7.314	34%	SeqGAN + Post-clustering	$\underline{0.507}$	$\underline{0.589}$	$\underline{0.075}$	-	-	-	$\underline{0.225}$	$\underline{0.169}$
			Pre-clustering + SeqGAN	0.563	0.626	0.140	-	-	-	0.336	0.252
			Our	**0.382**	**0.509**	**0.041**	-	-	-	**0.120**	**0.118**
Cluster_1	14.784	10%	SeqGAN + Post-clustering	$\underline{0.540}$	$\underline{0.597}$	$\underline{0.097}$	-	-	-	$\underline{0.293}$	**0.210**
			Pre-clustering + SeqGAN	0.700	0.721	0.248	-	-	-	0.477	0.335
			Our	**0.507**	**0.586**	**0.092**	-	-	-	**0.319**	$\underline{0.229}$
Cluster_3	30.747	1%	SeqGAN + Post-clustering	0.650	0.672	0.358	-	-	-	$\underline{0.459}$	$\underline{0.358}$
			Pre-clustering + SeqGAN	**0.487**	**0.560**	$\underline{0.335}$	-	-	-	0.705	0.423
			Our	$\underline{0.632}$	$\underline{0.660}$	**0.328**	-	-	-	**0.446**	**0.333**

Table 5.4: Comparison with baselines on GeoLife and PeopleFlow data on the task of next location prediction. The best performance is in boldface.

	Next Location Prediction on GeoLife Dataset							
Model	Accuracy@1	Accuracy@2	Accuracy@3	Accuracy@4	Accuracy@5	Accuracy@6	Accuracy@7	Accuracy@8
SeqGAN	0.842	0.913	0.934	0.944	0.951	0.956	0.959	0.963
Our	**0.880**	**0.930**	**0.944**	**0.954**	**0.960**	**0.964**	**0.967**	**0.970**
	Next Location Prediction on PeopleFlow Dataset							
Model	Accuracy@1	Accuracy@2	Accuracy@3	Accuracy@4	Accuracy@5	Accuracy@6	Accuracy@7	Accuracy@8
SeqGAN	0.831	0.882	0.905	0.916	0.927	0.932	0.937	0.941
Our	**0.888**	**0.912**	**0.921**	**0.927**	**0.933**	**0.936**	**0.940**	**0.942**

a known but incomplete sequence of visits remains to be explored.

To address the question, we conduct the ablation study of the next location prediction using our proposed model and the SeqGAN baseline.

5.4.2 Setting and Evaluation Metric

To evaluate the model's effectiveness on the task of next location prediction, we leverage the metric: accuracy@k, which denotes whether the ground-truth next location exists in top k predicted locations given the predicted probability distribution of the entire n locations from the generator. We set k from 1 (the ground-truth location is precisely the predicted next location) to 8 and leverage both our proposed generator and SeqGAN's generator to generate the probability distributions of the next possible location given the current location. The data is selected from the real trajectories.

5.4.3 Comparison and Finding

As indicated in Table 5.4, our proposed generator consistently performs better than the baseline model on both datasets. Our model outperforms the baseline at most 5% and at least 1%. More importantly, as k gets smaller, our model's advantage is more obvious. When k equals 1, our model performs 5% better than the SeqGAN model on the GeoLife Dataset and 7% better on the PeopleFlow Dataset.

Chapter 6

Conclusion and Discussion

6.1 Conclusion

Synthetic trajectory generation is an essential yet challenging task. How to effectively generate trajectories that preserve all essential trajectory-level properties and simultaneously capture various modality patterns in the real world is still a highly open research domain due to the complicated nature of real-world mobility patterns, limited labeled mobility data, and complex generation models with corresponding deficiencies. Besides, the lack of systematic metrics to measure whether the real-world transitional information is preserved and the lack of ablation studies of the generation model on the task of next location generation based on the incomplete sequence of visits invalidates the existing trajectory generation model. This paper proposes a novel yet generic framework, **Clustering-based Semi-supervised Generative Adversarial Network (CS-GAN)**; based on limited actual trajectories reported by users, our proposed model can generate synthetic trajectories which mimic the real ones by preserving all the essential properties. Our proposed model leverages the idea of clustering and semi-supervised GANs to capture real-life modality patterns. Moreover, we develop a novel transitional probability-related metric based on the OD matrix and the Frobenius norm to measure whether the synthetic trajectories capture the transitional information. We also conduct ablation studies to verify the effectiveness of our proposed generation model in predicting the possible subsequent few visits given an incomplete sequence of visits. Extensive experiments and case studies on two real-world datasets demonstrate our proposed model's consistent and superior performance in synthetic trajectory generation. Specifically, our proposed model outperforms other approaches with significant improvement.

6.2 Discussion

Several deficiencies need to be conquered in future works to make our proposed model more general and applicable:

1. The privacy of the real-life training trajectories is not guaranteed;
2. The model's robustness is not guaranteed from adversarial samples;
3. The clustering strategy needs to be more powerful to capture the various modality patterns in real-life trajectories fully.

These problems are prevalent and significantly affect the generalizability of the proposed model in real life. Thus to address them, the future directions can be:

1. Adding differential privacy or levering other privacy-preserving strategies to protect the model from membership inference attacks;
2. Leveraging adversarial ML to test the robustness of ML against adversarial samples;

3. Extending a multi-agent system to the generator by leveraging each separate agent to generate synthetic trajectories belonging to a particular modality.

Bibliography

[1] Fereshteh Asgari, Vincent Gauthier, and Monique Becker. A survey on human mobility and its applications. *arXiv preprint arXiv:1307.0814*, 2013.

[2] Jie Feng, Zeyu Yang, Fengli Xu, Haisu Yu, Mudan Wang, and Yong Li. Learning to simulate human mobility. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3426–3433, 2020.

[3] Sibren Isaacman, Richard Becker, Ramón Cáceres, Margaret Martonosi, James Rowland, Alexander Varshavsky, and Walter Willinger. Human mobility modeling at metropolitan scales. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, page 239–252, New York, NY, USA, 2012. Association for Computing Machinery. ISBN 9781450313018. doi: 10.1145/2307636.2307659. URL `https://doi.org/10.1145/2307636.2307659`.

[4] Shan Jiang, Yingxiang Yang, Siddharth Gupta, Daniele Veneziano, Shounak Athavale, and Marta C González. The timegeo modeling framework for urban mobility without travel surveys. *Proceedings of the National Academy of Sciences*, 113(37):E5370–E5378, 2016.

[5] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[6] Shengjie Lai, Nick W Ruktanonchai, Liangcai Zhou, Olivia Prosper, Wei Luo, Jessica R Floyd, Amy Wesolowski, Mauricio Santillana, Chi Zhang, Xiangjun Du, et al. Effect of non-pharmaceutical interventions to contain covid-19 in china. *nature*, 585(7825):410–413, 2020.

[7] Augustus Odena. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*, 2016.

[8] Kun Ouyang, Reza Shokri, David S Rosenblum, and Wenzhuo Yang. A non-parametric generative model for human trajectories. In *IJCAI*, volume 18, pages 3812–3817, 2018.

[9] Martin L Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

[10] Jinmeng Rao, Song Gao, Yuhao Kang, and Qunying Huang. Lstm-trajgan: A deep learning approach to trajectory privacy protection, 2020. URL `https://arxiv.org/abs/2006.10521`.

[11] Yoshihide Sekimoto, Ryosuke Shibasaki, Hiroshi Kanasugi, Tomotaka Usui, and Yasunobu Shimazaki. Pflow: Reconstructing people flow recycling large-scale social survey data. *IEEE Pervasive Computing*, 10(4):27–35, 2011.

[12] Nan Xu, Loc Trinh, Sirisha Rambhatla, Zhen Zeng, Jiahao Chen, Samuel Assefa, and Yan Liu. Simulating continuous-time human mobility trajectories.

[13] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[14] Yu Zheng, Hao Fu, Xing Xie, Wei-Ying Ma, and Quannan Li. *Geolife GPS trajectory dataset - User Guide*, geolife gps trajectories 1.1 edition, July 2011. URL `https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/`. Geolife GPS trajectories 1.1.

[15] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. 2005.