HRNet: Differentially Private Hierarchical and Multi-Resolution Network for Human Mobility Data Synthesization

Shun Takagi *
Kyoto University
shun021677@gmail.com

Li Xiong
Emory University
lxiong@emory.edu

Fumiyuki Kato
Kyoto University
fumilemon79@gmail.com

Yang Cao
Tokyo Institute of Technology
cao@c.titech.ac.jp

Masatoshi Yoshikawa
Osaka Seikei University
yoshikawa-mas@osaka-seikei.ac.jp

ABSTRACT

Human mobility data offers valuable insights for many applications such as urban planning and pandemic response, but its use also raises privacy concerns. In this paper, we introduce the Hierarchical and Multi-Resolution Network (HRNet), a novel deep generative model specifically designed to synthesize realistic human mobility data while guaranteeing differential privacy. We first identify the key difficulties inherent in learning human mobility data under differential privacy. In response to these challenges, HRNet integrates three components: a hierarchical location encoding mechanism, multi-task learning across multiple resolutions, and private pre-training. These elements collectively enhance the model's ability under the constraints of differential privacy. Through extensive comparative experiments utilizing a real-world dataset, HRNet demonstrates a marked improvement over existing methods in balancing the utility-privacy trade-off.

PVLDB Reference Format:
Shun Takagi, Li Xiong, Fumiyuki Kato, Yang Cao, and Masatoshi Yoshikawa. HRNet: Differentially Private Hierarchical and Multi-Resolution Network for Human Mobility Data Synthesization. PVLDB, 17(11): 3058 - 3071, 2024.
doi:10.14778/3681954.3681983

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/Emory-AIMS/priv_traj_gen.

1 INTRODUCTION

In recent years, the use of human mobility data has gained significant attention for its potential to contribute to societal benefits, such as traffic forecasting, urban planning, and pandemic response, including COVID-19 spread analysis [9, 39, 59]. However, it also raises critical privacy concerns even if the data are aggregated and anonymized [49, 56].

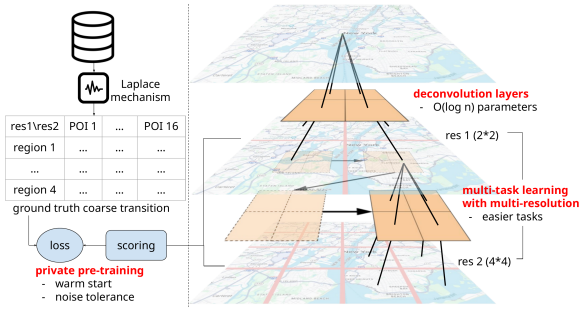

Figure 1: HRNet utilizes three novel components to address the bottlenecks of applying DP-SGD in human mobility generation: 1) a hierarchical location encoding mechanism using deconvolution networks, 2) multi-task learning across multiple resolutions, and 3) private pre-training using a DP coarse transition matrix.

Differential Privacy (DP) [16] has emerged as the leading standard for maintaining data privacy. DP offers a strong, mathematically grounded privacy guarantee without relying on restrictive assumptions about potential adversaries. The core principle of DP is to ensure that output from the data analysis does not substantially differ (bounded by a privacy parameter or privacy budget), regardless of whether any specific individual's data is included or excluded from the dataset. The broad applicability and growing adoption of DP in the real world [14, 19, 28, 54], as well as its endorsement by the US Census Bureau [7], signify its importance and effectiveness.

In this paper, we study the problem of DP data synthesis [6, 33, 65] for human mobility data. DP data synthesis generates synthetic data that possesses statistical properties similar to the real data while ensuring DP. According to the post-processing theorem of DP, unlimited analysis can be performed on this synthetic data without introducing further privacy concerns. Our goal is to synthesize human mobility data, defined as a sequence of locations, to closely resemble the real human mobility data, which can be then used for a variety of downstream tasks in previously mentioned applications.

Existing DP data synthesis methods for human mobility primarily rely on DP-aware data structures, which can be categorized into tree-based [8, 10, 24], Markov-based [23, 58], and clustering approaches [27, 32, 41]. These methods generally employ generalization to reduce dimensionality and sensitivity, thereby balancing

*Work partially done while visiting Emory University.

Proceedings of the VLDB Endowment, Vol. 17, No. 11 ISSN 2150-8097.
doi:10.14778/3681954.3681983

utility and privacy, but often at the cost of information loss. For instance, clustering approaches only generate transitions between k centroids by location generalization using clustering, leading to a loss of precise transition information within the clusters. Moreover, the state-of-the-art Markov-based method [58] does not account for correlations beyond two-step intervals, and many methods overlook auxiliary information such as time, which is crucial in human movement. Tree-based methods aim to minimize such loss by constructing a data-dependent tree with a portion of the privacy budget. However, such consumption leads to a lack of privacy budget during the data generation phase, which results in diminished utility.

MTNet [60] is a deep learning approach designed for generating short trips on road networks, such as taxi trips, while ensuring DP. By utilizing meta-information from road networks, which is not sensitive, MTNet adeptly manages the utility-privacy trade-off. However, this approach imposes the inherent limitation of restricting mobility to road networks. Human mobility, in reality, is not confined to such networks; it encompasses various modes including subway travel and movement through non-road spaces like parks. Additionally, even when focused on road networks, human mobility often evolves into lengthy sequences of road segments, a complexity that poses significant challenges under DP constraints. Consequently, adapting MTNet for a broader spectrum of human mobility appears to be an unpromising approach.

To the best of our knowledge, this is the first paper to adopt a deep learning approach with DP for generating human mobility which is not constrained to road networks. Recent advancements have shown that deep learning approaches handling human mobility data outperform traditional models [43, 59]. There are several generative models such as adversarial network (GAN) based models for generating realistic synthetic trajectories [21, 67]. However, these models do not ensure formal privacy guarantee. Theoretically, it is possible to ensure DP for any deep learning models by replacing the optimization method with differentially private stochastic gradient descent (DP-SGD) [1]. Unfortunately, simply adopting DP-SGD on these generative models results in either poor utility or infeasibly high privacy cost due to the large space domain and complexity of the models. More concretely, we elaborate the two primary challenges as follows.

(1) **The number of model parameters:** It has been demonstrated [4, 5] that DP-SGD inevitably increases the lower bound of empirical risk as the number of parameters increases. In most models about human mobility (e.g., RNN model [63] and attention model [62]), the number of model parameters depends on the size of the space domain. If we encode the space or Points of Interests (POIs) using an encoding scheme such as embedding matrices or linear embeddings, the number of model parameters is $O(n_{\text{POI}})$ where n_{POI} denotes number of POIs, which can be significantly large.
(2) **Learning difficulty:** the difficulty of learning human mobility escalates with increasing n_{POI} due to the increasing number of labels for training. Therefore, a larger n_{POI} necessitates larger models and longer training epochs to discern subtle POI differences. This leads to decreased utility due to the large number of parameters and higher consumption of privacy budgets due to the composition theorem [30].

Contributions. In response to these challenges, we present the Hierarchical and Multi-Resolution Network (HRNet), which encompasses a novel network structure and learning methodology. The key features of HRNet, as illustrated in Figure 1, are threefold:

(1) **Hierarchical location embedding:** to alleviate the first issue, HRNet adopts a hierarchical structure for location embedding via transposed convolution. This approach significantly reduces the number of parameters to $O(\log(n_{\text{POI}}))$, thereby alleviating the decrease in utility commonly associated with traditional embedding methods that have $O(n_{\text{POI}})$ parameters.
(2) **Multi-task learning with multiple resolutions:** to alleviate the second issue, we introduce multi-task learning. Our hierarchical network design enables encoding of multiple resolutions. Beyond mastering the primary task, the network concurrently engages in learning additional, less complex tasks with coarser resolutions. This approach allows for the intricate primary task to be deduced from these simpler sub-tasks, thereby efficiently mitigating the learning difficulties associated with a large number of POIs.
(3) **Effective and private pre-training:** to alleviate both issues, we conduct private pre-training. The effectiveness of pre-training in addressing the first issue of DP-SGD has been underscored [3, 29]. Additionally, pre-training provides a 'warm start', effectively reducing the need for extensive training epochs, thus alleviating the second issue. A common challenge with pre-training is its reliance on public data, which may not always be readily available or suitable for all scenarios. HRNet enables efficient pre-training without public data by leveraging a dense DP-compatible transition matrix.

2 PRELIMINARIES

2.1 Differential Privacy

Differential Privacy (DP), as introduced by Dwork et al. [16], provides a robust mathematical framework for quantifying privacy leakage in data publication scenarios. The formal definition of DP is as follows:

Definition 2.1 ((ε, δ)-Differential Privacy). Consider a dataset domain $\mathcal{D}$ and output domain $\mathcal{Z}$. Given $\varepsilon \in \mathbb{R}^+$ and $\delta \in [0, 1]$, a randomized mechanism m, which randomly outputs $m(D) \in \mathcal{Z}$ with input $D \in \mathcal{D}$, satisfies (ε, δ)-DP if, for any two datasets $D, D' \in \mathcal{D}$ differing in at most one element, and for any subset of outputs $Z \subseteq \mathcal{Z}$, the following inequality holds:

$$\Pr[m(D) \in Z] \leq e^{\varepsilon} \Pr[m(D') \in Z] + \delta.$$

DP-SGD: Differentially Private Stochastic Gradient Descent (DP-SGD), as developed by Abadi et al. [1], adapts the conventional SGD optimization process for deep learning models to satisfy DP. In traditional SGD, parameters of a model θ are iteratively updated to minimize empirical risk, with gradient computations performed using sampled data (minibatches). DP-SGD modifies this process to ensure DP by introducing gradient clipping and noise addition

steps. Specifically, the l_2 norm of the gradient is first clipped to limit sensitivity, followed by the addition of the Gaussian noise to the averaged clipped gradients. The model parameters are then updated using these randomized gradients, similar to conventional SGD. This process continues until convergence is achieved or the allocated privacy budget is depleted. From a DP perspective, this process is interpreted as the sequential composition of the Gaussian mechanism with subsampling (i.e., minibatch). Therefore, by employing composition techniques, the final published model satisfies DP. We adopt numerical composition, which numerically computes the upper bound of the privacy loss parameter ε using privacy loss distributions [50]. Several studies enhance its computation using techniques such as the Fast Fourier Transform [31] and the "connect-the-dots" algorithm [15]. For implementation, we utilize Google's differential-privacy library[1].

2.2 Problem Formulation

This study aims to generate synthetic human mobility data that accurately mirrors actual human mobility (e.g., daily human mobility such as home to workplace to home). Due to the inherently continuous nature of human mobility in terms of both time and geographical coordinates (latitude and longitude), this presents significant challenges in direct modeling and evaluation. To address this, we adopt a conventional approach of discretizing human mobility data, thereby simplifying the modeling process. This section first formulates discretized human mobility representation. It then details the formulation of a generator for discretized human mobility and outlines the evaluation methodology.

2.2.1 Trajectory: As a fundamental step, we define Points of Interests (POIs)[2] as uniformly distributed grid cells on a map, with each grid cell assigned a unique integer ID. For instance, if a target map is divided into a $w \times w$ grid, the domain of POIs is defined as $L := [w^2]$, where $[n]$ denotes the set $(1, 2, \ldots, n)$. That is, $n_{\text{POI}} = |L|$. As illustrated in Figure 1, different levels of discretization are possible, such as 1×1, 2×2, and 4×4. A geographical location, defined by its latitude and longitude, is transformed into an integer representing the grid cell that encompasses the location. The mapping is performed such that the grid cell located at $(\lfloor l/w \rfloor, l\%w)$ is assigned the integer $l \in L$. We then represent a human mobility sequence as a sequence of stay points, denoted as $\mathbf{v} = [v_1, v_2, \ldots]$, where $i \in \mathbb{N}, v_i = (l_i, t_i), l_i \in L$. Note that this representation aligns POIs that change from the previous POI so that $l_i \neq l_{i+1}$. Here, t_i is the index of the time slot that includes the corresponding time. Time slots are made by discretizing entire time by n_{time}, where n_{time} is the number of time slots. For example, v_2 indicates the tuple of the second POI visited and the time at which this POI was reached. From this point forward, the term 'trajectory' will refer to this discretized representation of human mobility.

2.2.2 Generator: In our study, we employ a neural network-based generator, denoted as G_θ, to stochastically generate synthetic trajectories. The process can be mathematically represented as $\mathbf{v} \sim G_\theta$, where $\mathbf{v}$ represents the generated synthetic trajectory, and θ symbolizes the trainable parameters of the neural network. A detailed example of such a generator, including its architecture and operational mechanics, will be discussed in Section 2.3.

2.2.3 Evaluation: Direct evaluation of the performance of G_θ is intractable due to the inherent sparsity of trajectory data. To overcome this, we adopt an indirect evaluation approach, focusing on various statistical properties of trajectories. These statistical properties are computed from the dataset generated by G_θ and are then compared with empirical statistics derived from the provided trajectory dataset D. Key aspects of trajectory data that we examine include waypoints, routes, destinations, transitions, and travel distances. By analyzing these specific statistics, we can gain insights into the accuracy and fidelity of the trajectories generated by G_θ. The formal definitions and methodologies for computing these statistics are detailed in Section 5.1. This approach enables a thorough and nuanced evaluation of the generator's performance in replicating realistic human mobility patterns.

2.3 The Baseline Generator

This section delineates the architecture and operational mechanics of the baseline generator used in our study.

2.3.1 The baseline generator. The baseline generator is structured upon the chain rule of conditional probabilities:

$$G_\theta = \Pr(\mathbf{v}) = \prod_{i=1}^{|\mathbf{v}|} \Pr(v_i | v_1, ..., v_{i-1}). \tag{1}$$

This formulation allows the generator to sequentially construct a trajectory from $i = 1$ by modeling each conditional probability $\Pr(v_i | v_1, \ldots, v_{i-1})$. To accomplish this, the generator integrates three core components: embedding matrices for data points v_i, a recurrent neural network (RNN) to process sequences, and a scoring component to make a distribution.

Embedding Matrix: The embedding matrix translates each data point $v_i = (l_i, t_i)$ to a vector representation. Two separate embedding matrices are employed: one for the POI (l_i) and another for time (t_i). The POI embedding matrix M_{POI} contains $|L|$ trainable vectors, whereas the time embedding matrix M_{time} includes n_{time} trainable vectors. The vector representation of v_i is formed by concatenating the t_ith vector from the time embedding matrix with the l_ith vector from the POI embedding matrix:

$$\text{encode}(v_i) = [M_{\text{time}}[t_i], M_{\text{POI}}[l_i]]. \tag{2}$$

Here, $M[i]$ represents the access to the vector of M at ith index and $[a, b]$ represents the concatenation of vectors a and b.

Recurrent Neural Network: For encoding trajectory prefixes $(v_1, \ldots, v_{i-1})$, we employ Gated Recurrent Units (GRUs) [12]. This is because we have empirically found that GRU is slightly more effective than alternative methods such as LSTM [25] and attention mechanisms [57] in the context of DP-SGD. We attribute this to the GRU's simpler architecture and reduced parameters. The GRU cell updates its state based on the previous state h_{i-1} and the encoded v_{i-1}: $h_i = f_{\text{GRU}}(h_{i-1}, \text{encode}(v_{i-1}))$. This process is applied recursively to encode the entire trajectory prefix.

[1] https://github.com/google/differential-privacy (Accessed: July 14, 2024)

[2] Here, the term 'POI' is not used in its original sense of indicating semantic locations. However, our method can be applied in its original context as described in Section 3.2.2.

Scoring component: The scoring component is responsible for converting the encoded prefix into the probability distribution over $[n_{\text{POI}}] \times [n_{\text{time}}]$ for next predicted location. This conversion is accomplished using feed-forward neural networks $g_{\text{POI}} : \mathbb{R}^{n_{\text{hidden}}} \rightarrow \mathbb{R}^{|L|}$ and $g_{\text{time}} : \mathbb{R}^{n_{\text{hidden}}} \rightarrow \mathbb{R}^{n_{\text{time}}}$ where n_{hidden} is the dimensionality of the hidden state h_i, followed by a softmax function to ensure proper probabilistic normalization. The probability distribution over spatial location $l \in L$ given the hidden state h_i is computed as $\Pr(l = l_i|h_i) = \text{softmax}(g_{\text{POI}}(h_i))[l_i]$. Similarly, the probability distribution over time $t \in [n_{\text{time}}]$ given h_i is determined by $\Pr(t = t_i|h_i) = \text{softmax}(g_{\text{time}}(h_i))[t_i]$. Ultimately, the conditional probability $\Pr(v_i|v_1, \ldots, v_{i-1})$ given the previous observations is the product of these two distributions, representing a joint probability over POI and time: $\Pr(v_i|v_1, \ldots, v_{i-1}) = \Pr(l = l_i|h_i) \Pr(t = t_i|h_i)$.

2.3.2 The objective function: To optimize the baseline generator, given a training trajectory sequence $\mathbf{v} = (v_1, \ldots, v_i)$, we employ cross entropy loss as our objective function. The loss is calculated as follows:

$$\text{loss}_{\text{POI}} + \text{loss}_{\text{time}} = \sum_{i=1}^{|v|} \left(\sum_{c \in L} \delta_{c,l_i} \log(\Pr(l = l_i|h_i)) + \sum_{c \in [n_{\text{time}}]} \delta_{c,t_i} \log(\Pr(t = t_i|h_i)) \right),$$

where $\delta_{i,j}$ is the Kronecker delta.

2.3.3 The sequential generation: The trained generator is capable of generating human mobility sequences by employing the chain rule as delineated in Equation (1). Specifically, the process begins with the initial POI. Subsequently, each following POI is recursively sampled based on the conditional probability that is informed by the previously sampled POIs. This procedure is iteratively executed until a predefined condition, such as reaching a maximum sequence length or a specific vocabulary signifies the end of the sequence.

2.4 The two bottlenecks

The baseline model discussed earlier shows a decrease in performance when DP-SGD is applied, particularly as the number of POIs increases. In this section, we identify and elaborate on the two primary factors causing these performance bottlenecks.

2.4.1 The number of parameters $|\theta|$. Under the constraints of DP, it is known that the lower bound of empirical risk of the private model polynomially scales [4, 5] as the number of model parameters increases. In the baseline model, the size of the parameters θ expands linearly with increasing number of POIs (n_{POI}) due to the location embedding matrix and the scoring component. Consequently, the empirical risk would polynomially escalate as n_{POI} increases, which leads to worse utility of the generated human mobility.

2.4.2 The number of labels. With an increase in the number of class labels (i.e., n_{POI}), the model faces challenges in discerning more subtle feature distinctions between classes (see Section 2.3.2). This often necessitates the use of larger model architectures or extended training epochs. However, such strategies are impractical under DP-SGD constraints. In addition to the increased empirical risk due to the increased model size as mentioned above, increasing the number of training epochs results in greater privacy loss due to the composition theorem of DP [30]. Therefore, learning difficulty caused by the large number of labels is a bottleneck of DP-SGD.

2.4.3 Discussion: impact of the discretization parameter. In many deep learning methods [35, 37, 58], as in this study, latitudes and longitudes are discretized using a grid. However, determining the parameters for this discretization (i.e., w in this study) is not straightforward. This is because finer discretization makes the data more sparse and increases the number of parameters required for the network (location encoding). As discussed above, DP-SGD leads to a decrease in accuracy in these cases. However, finer grids have the advantage of capturing more detailed movements between grid cells, improving the quality of the training data and potentially increasing accuracy. Thus, there is a trade-off between the accuracy improvement from higher-quality training data and the accuracy decrease due to the noise required by DP. Therefore, mitigating the two bottlenecks makes it less susceptible to the negative effects of high-resolution discretization and thereby improves accuracy.

2.5 Transposed Convolution

In this section, we delve into the transposed convolutional operation, a key element of the main component in our proposed model. Originating from the domain of computer vision [42], the transposed convolution layer, also known as a deconvolution layer, is typically employed to upsample or reconstruct the original images from feature maps generated by convolutional layers. In contrast to its conventional usage, our model uniquely incorporates the specific instance of this layer, which we have termed the '2D quad deconvolutional network'. This incorporation of the transposed convolution layer is instrumental in addressing the two primary bottlenecks identified earlier.

2D quad deconvolutional network. Consider the matrix M with shape (w, w, n_{dim}), where n_{dim} denotes the number of channels, and each channel comprises a $w \times w$ square matrix. Similarly, let the deconvolutional kernel be defined as a matrix of shape $(2, 2, n_{\text{dim}})$, containing n_{dim} kernels, and each of these kernels is 2×2 matrix denoted by kernel_k. The 2D quad deconvolutional network expands the matrix M to the matrix M' with dimensions $(2w, 2w, n_{\text{dim}})$ using a deconvolutional kernel. See Figure 1 for the running example. In the second layer, the kernel slides over the input of (2×2) matrix to perform the deconvolution operation for the four grids, expanding a 2×2 matrix into a 4×4 matrix.

Let x and y be integers, the transformation can be mathematically represented as:

$$M'[2x + \delta_x, 2y + \delta_y, k] = \sum_{k'=1}^{n_{\text{dim}}} \text{kernel}_k[\delta_x, \delta_y, k'] M[x, y, k'],$$

where, $k \in [n_{\text{dim}}]$, δ_x and δ_y are binary values (0 or 1), and notation $M[i, j, k]$ represents access to an element of the matrix M with the index (i, j, k). This process effectively quadruples each n_{dim}-dimensional vector in the matrix. As a result, the original n_{dim}-dimensional vector $M[x, y]$ is expanded into four distinct n_{dim}-dimensional vectors: $M'[2x, 2y]$, $M'[2x+1, 2y]$, $M'[2x, 2y+1]$, and $M'[2x + 1, 2y + 1]$. Applying the 2D quad deconvolutional layer d times multiplies the number of n_{dim}-dimensional vectors by 4^d.

When w is a power of 2, originating from a single n_{dim}-dimensional root vector θ_{root}, the final transformation of the network can be represented as:

$$M = \text{deconv}^{\log_4 w^2}(\theta_{\text{root}}),$$

where deconv^a denotes the iterative application of the deconvolution operation a times.

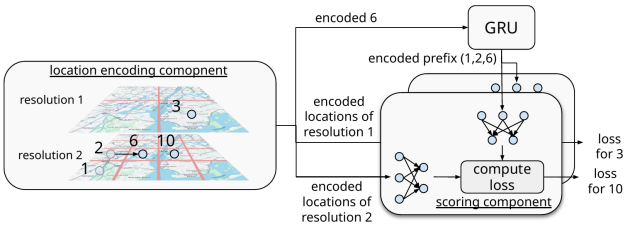

Figure 2: Overview of multi-task training with hierarchical location encoding. In this example, we assume that $w = 4$, so we have 4*4 POIs at resolution 2 and 2*2 regions at resolution 1 due to the hierarchical location encoding. Given prefix $(1, 2, 6)$, the model learns to infer grid cell 10 at resolution 2, as well as grid cell 3 at resolution 1.

3 HRNET

In this section, we introduce the Hierarchical and Multi-Resolution Network (HRNet). HRNet's core novelty is the integration of a hierarchical location encoding component, which supersedes the traditional embedding matrix and scoring component. This design reduces the number of parameters, effectively alleviating the first bottleneck. Moreover, it facilitates multi-resolution interpretation, enabling multi-task learning. This approach allows HRNet to process and learn from data at multiple resolutions simultaneously, adeptly handling the difficulty of learning at the finest resolution, which alleviates the second bottleneck. Together, they contribute to HRNet's robustness to the number of POIs n_{POI} when optimized with DP-SGD.

We begin by outlining HRNet, using a simple example illustrated in Figure 2. Subsequently, we delve into detailed explanations of each component.

3.1 Overview

We provide the overview of multi-task training with hierarchical location encoding of HRNet, illustrated in Figure 2. HRNet builds upon the baseline model that models conditional probabilities. See Section 2.3 for detail of the baseline model. Note that we simplify our explanation, sidelining the training of time information which is the same as the baseline, to emphasize the principal distinctions.

Consider a scenario depicted in Figure 2. The example involves two resolutions, featuring 4 and 16 grid cells respectively, and trajectory $(1, 2, 6, 10)$ at resolution 2. In the learning phase for $v = (1, 2, 6, 10)$, step 1 is learning 2 from prefix (1), step 2 is learning 6 from prefix $(1, 2)$, and step 3 is learning 10 from prefix $(1, 2, 6)$. Consider step 3, HRNet operates as follows:

The hierarchical location encoding component encodes location 6. The GRU cell computes h_3, embedding of the prefix $(1, 2, 6)$, using h_2 and the encoded vector of 6. h_3 forms a *query vector* with the feed-forward neural networks for subsequent steps. Unlike the baseline model that focuses exclusively on resolution 2, HRNet concurrently considers resolution 1, that is, learning 3 at resolution 1 from $(1, 2, 6)$. For each resolution, the model encodes all grid cells and convert them to *key vectors* with feed forward neural networks. It then calculates scores for the next location given the prefix $(1, 2, 6)$ by performing a dot product with the query vector. This operation generates a probability distribution for each resolution (i.e., over [4] and [16]), using the softmax function. The model computes the cross-entropy loss based on this probability distribution and the actual value, i.e., 3 at resolution 1, and 10 at resolution 2. Finally, the model calculates the gradients of the parameters θ from the summed cross-entropy losses. It then employs DP-SGD for model update.

3.2 Hierarchical Location Encoding

HRNet introduces a novel approach in its location encoding mechanism, which has smaller number of parameters than the baseline model. The key feature is the employment of a hierarchical network structure with 2D quad deconvolutional network.

3.2.1 The structure. The core of HRNet's location encoding is a trainable root vector, denoted as θ_{root}. This root vector is augmented with a 2D quad deconvolutional network comprising of d layers. The process involves the application of this 2D quad deconvolutional network to θ_{root}, producing 4^d distinct vectors. Essentially, this operation results in an embedding matrix of dimensions $(2^d, 2^d, n_{\text{dim}})$: $M_d = \text{deconv}^d(\theta_{\text{root}})$. For a detailed explanation of the deconv operation, refer to Section 2.5. Notably, the number of parameters in each deconvolutional layer only depends on the kernel size and is independent of n_{POI}, sustaining a constant number of parameters. Consequently, this architecture requires only $O(d) = O(\log n_{\text{POI}})$ parameters to generate n_{POI} vectors, which is a significant reduction compared to the $O(n_{\text{POI}})$ parameters typically needed in embedding matrix.

3.2.2 Vector assignment. The assignment of the 4^d vectors generated by the hierarchical location encoding component to the n_{POI} POIs follows a specific methodology. Given that the grid size, $w * w$, is a power of 4, we set $d = \log_4(w * w)$. This approach facilitates an alignment of vectors with the grid, described as:

$$\text{hiencode}(l) = M_d[\lfloor l/w \rfloor, l\%w],$$

where $l \in L$. This assignment method naturally incorporates the first law of geography [55], suggesting that geographically proximate locations have similar embeddings. This proximity principle is a direct result of the deconvolutional operation, where a single parent vector generates four adjacent embeddings.

Scattered POI assignment. The above assignment assumed that POIs are the grid cells according to a uniform grid as described in Section 2.2.1 for simplicity. Note that this grid assumption is not necessary and HRNet can work with scattered POIs as long as the number of POIs is equal to or less than 4^d. As mentioned above, we should follow the first law of geography rather than randomly assigning vectors to the scattered POIs. Thus, we propose a method for assigning scattered POIs to vectors in M_d following the

proximity principle. First, we create a grid that covers all scattered POIs. Here, we set the minimum d so that the number of POIs is less than 4^d, and $w = 2^d$. Then, we compute the coordinates of the center of each grid cell (cell_j) and assign each POI (POI_i) to a unique cell so that the Euclidean distance $d(\text{POI}_i, \text{cell}_j)$ is as small as possible. This assignment can be formalized as a linear assignment problem with a cost matrix C, where

$$C = \{d(\text{POI}_i, \text{cell}_j)\}_{i \in [n_{\text{POI}}], j \in [4^d]}.$$

This optimization can be solved using the Jonker-Volgenant algorithm [13]. Once each POI is assigned to a unique grid cell, we can use the vector assignment method described above. Since POIs are aligned based on geographical proximity, this embedding also incorporates the proximity principle of the deconvolutional network.

3.3 Scoring

HRNet employs an efficient scoring mechanism that combines the hierarchical location encoding with dot product operations to establish a probability distribution over POIs.

Formulation of the Scoring Process. In this process, the encoded prefix sequence $(v_1, \ldots, v_{i-1})$, represented by h_{i-1}, is transformed into a query vector. This transformation is executed using a feed-forward neural network, denoted as f_{query}. Concurrently, the encoded POI, $\text{hiencode}(l)$, is processed into a key vector through another feed-forward neural network, named f_{key}. Mathematically, this can be represented as $\text{query} = f_{\text{query}}(h_i)$ and $\text{key}_l = f_{\text{key}}(\text{hiencode}(l))$. Following these transformations, the score of l as v_i is determined by computing the dot product between the query and the key vector $\text{score}_l = \text{query} \cdot \text{key}_l$. The final step involves applying the softmax function to these scores across all POIs, resulting in a probability distribution over POIs: $\Pr(v_i = l_i | v_1, \ldots, v_{i-1}) = \text{softmax}(\text{score}_L)[l_i]$, where $\text{score}_L = (\text{score}_1, \ldots, \text{score}_{|L|})$.

3.4 Multi-resolution Learning

In this section, we address the challenge of learning difficulty associated with a large number of labels by redefining the objective function in HRNet. The solution lies in employing a multi-resolution interpretation, facilitated by the hierarchical location encoding component. The essence of our approach is to extend beyond the original task by incorporating multiple tasks across various resolutions. This multi-resolution strategy entails not only focusing on the primary task at hand but also simultaneously considering tasks at coarser resolutions, which are easier than the primary task. By doing so, the model gains a broader understanding of the data from the coarser resolutions, allowing it to better manage the complexity that comes with a large label space even in the noise of DP.

3.4.1 Multi-resolution Interpretation. Here, we maintain the assumption that the number of grid cells, denoted as $w \times w$, is a power of 4, with $w = 2^d$. This forms the basis for our multi-resolution interpretation, which extends beyond the primary grid division of $2^d \times 2^d$ (as described in Section 2.2).

We consider grid divisions of $2^{i_{\text{res}}} \times 2^{i_{\text{res}}}$ at each resolution level $i_{\text{res}} \in \{1, 2, \ldots, d-1\}$. Here, the i_{res}th resolution includes $L_{i_{\text{res}}}$ grid cells, represented as: $i_{\text{res}} \in [d], L_{i_{\text{res}}} := [4^{i_{\text{res}}}]$. The matrix $M_{i_{\text{res}}}$ (refer to Section 3.2.1) encodes these grid cells by applying the vector assignment in Section 3.2.2. The encoding for grid cell l at resolution i_{res} is thus:

$$\text{hiencode}_{i_{\text{res}}}(l) = M_{i_{\text{res}}}[\lfloor l/2^{i_{\text{res}}} \rfloor, l\%2^{i_{\text{res}}}],$$

where $l \in L_{i_{\text{res}}}$. This encoding is then fed into the scoring component to generate a probability distribution over the grid cells at resolution i_{res}:

$$\Pr(l_{i_{\text{res}}} = l | v_1, \ldots, v_{i-1}) = \text{softmax}(\text{score}_{L_{i_{\text{res}}}})[l],$$

where $l \in L_{i_{\text{res}}}$ and $\text{score}_{L_{i_{\text{res}}}} = (\text{score}_1, \ldots, \text{score}_{|L_{i_{\text{res}}}|})$. Note that we use the query vector that is computed by GRU, which is common in all the resolutions.

3.4.2 The Objective Function. Our novel training strategy leverages the multi-resolution interpretation, incorporating multiple tasks across different resolutions. For a given grid cell y in L_d (the POI at the primary task level), the goal is to learn this grid cell and simultaneously learn its corresponding grid cells at lower resolutions i_{res}, where $i_{\text{res}} < d$. That is, this approach introduces additional $d-1$ tasks into our training.

Each task at resolution i_{res} comes with its own loss function:

$$\text{loss}_{i_{\text{res}}} = \sum_{l \in L_{i_{\text{res}}}} \delta_{l, \text{up}_{i_{\text{res}}}(y)} \log(\Pr(l_{i_{\text{res}}} = l | v_1, \ldots, v_{i-1})),$$

where $\text{up}_{i_{\text{res}}}(y) \in L_{i_{\text{res}}}$ is the grid cell covering y at resolution i_{res}. $\Pr(l_{i_{\text{res}}} = l | v_1, \ldots, v_{i-1})$ represents the inferred probability of the next grid cell being l at resolution i_{res}. These losses for all resolutions are then summed up, and this sum is used to compute the gradient for backward propagation.

Optimizing these loss functions facilitates the determination of the probability distribution for the next grid cell not only at the primary resolution d but also at each additional resolution i_{res}. Moreover, the tasks defined at smaller i_{res} are inherently simpler due to the reduced number of labels $|L_{i_{\text{res}}}|$. Inferring finer cells from well-trained coarser resolution due to the simplicity, guided by the first law of geography, assists in mitigating the learning difficulty associated with the primary task.

4 PRIVATE PRE-TRAINING

Pre-training using publicly available data has been recognized as an effective strategy to mitigate the limitations of DP-SGD [3]. However, the reliance on publicly accessible data is often a significant constraint, as such data may not always be available. To address this challenge, we introduce a novel private pre-training methodology that utilizes a DP compliant transition matrix, eliminating the need for public data. This private pre-training strategy not only provides a 'warm start' to accelerate the model's convergence but also enhances the model's capability to reduce empirical risk.

4.1 DP Transition Matrix

In our approach, we utilize a first-order transition matrix as ground truth for pre-training. The core task in this context is to predict the distribution of the next POI from a given POI. However, when dealing with a large number of POIs n_{POI}, a direct POI-to-POI first-order transition matrix becomes excessively sparse. To avoid this issue, we leverage the multi-resolution interpretation of HRNet. Instead of a direct POI-to-POI transition, we consider transitions

from a broader region, specifically a grid cell at a coarser resolution $l_{i_{\text{res}}} \in L_{i_{\text{res}}}$, to a POI $l \in L_d$.

$$\text{TRAN}_{i_{\text{res}}}[l_{i_{\text{res}}}, l] = \sum_{\mathbf{v} \in D} \mathbf{1}_{V_{l_{i_{\text{res}}}, l}}(\mathbf{v})/|\mathbf{v}|, \tag{3}$$

where $\mathbf{1}$ is an indicator function and $V_{l_{i_{\text{res}}}, l}$ is a subset of trajectory universe $\mathcal{V}$, which includes trajectories that have transition from $l_{i_{\text{res}}}$ to l: $V_{l_{i_{\text{res}}}, l} := \{\mathbf{v} \in \mathcal{V} | \exists i, \text{up}_{i_{\text{res}}}(\mathbf{v}[i]) = l_{i_{\text{res}}} \,\&\, \mathbf{v}[i{+}1] = l\}$. Note that this calculation normalizes the count with the sequence length $|\mathbf{v}|$, to limit sensitivity to 1 for DP. To ensure DP, we introduce Laplace noise into the matrix as follows:

$$\text{DPTRAN}_{i_{\text{res}}}[l_{i_{\text{res}}}, l] = \text{TRAN}_{i_{\text{res}}}[l_{i_{\text{res}}}, l] + \text{Laplace}(\varepsilon), \tag{4}$$

where $\text{Laplace}(\varepsilon)$ is a Laplace noise that leads to ε-DP when sensitivity is 1. This approach results in a denser transition matrix, hence more informative and less noisy, which is useful in pre-training. Figure 1 left side shows a sample of such coarse transition matrix.

4.2 Pre-training

This section outlines the pre-training methodology using the DP transition matrix. During this phase, the core objective is to generate a probability distribution depicting the likelihood of transitions from a given region (a grid cell at resolution i_{res}) to the POIs. It is important to note that while both the hierarchical location encoding component and its scoring component undergo pre-training, the prefix encoding component (GRU) (see Figure 2) does not since the DP transition matrix does not contain prefix information.

4.2.1 The Architecture. The architecture for the pre-training phase incorporates a temporary substitute for GRU, necessitated by the absence of prefix information in the DP transition matrix. This temporary component is designed to mimic the function of the GRU, generating a query vector with dimensions identical to those of an encoded prefix. Conceptually, this query vector symbolizes the initial grid of the transition, as represented by the following equation:

$$h = \text{temp}(x), \tag{5}$$

where x is the input vector that represents the given region, with further details provided in the following section. Notably, this temporary component is exclusive to the pre-training phase and is replaced with GRU during model training.

Aside from this modification, the architecture during pre-training remains consistent with the HRNet model. The scoring component processes the vector h output by the temporary component, in tandem with POI encodings obtained from the hierarchical encoding component. The query vector is computed as $\text{query} = f_{\text{query}}(h)$, and the key vector for a potential POI l is computed as $\text{key}_l = f_{\text{key}}(\text{hiencode}_d(l))$. The score for POI l is then calculated using a dot product $\text{score}_l = \text{query} \cdot \text{key}_l$. The application of a softmax function to these computed scores results in a probability distribution over L.

4.2.2 The Objective Function. To enhance learning, we propose a data augmentation approach to augment the transition matrix, achieved through proportional blending of regions (grid cells at resolution i_{res}). A mixing ratio vector is defined as: $\mathbf{r} = (r_1, \ldots, r_{|L_{i_{\text{res}}}|}) \in [0,1]^{|L_{i_{\text{res}}}|}$, subject to the constraint $\sum_{i=1}^{|L_{i_{\text{res}}}|} r_i = 1$. Here, each r_i denotes the proportion of region i in the initial state, forming a prior distribution as a multinomial distribution with probabilities $\mathbf{r}$. Then, the ground truth probability distribution given $\mathbf{r}$ is calculated as:

$$\Pr(l_{\text{next}}|\mathbf{r}) = \sum_{l \in L_{i_{\text{res}}}} \text{Mult}(l_{\text{initial}} = l|\mathbf{r}) \Pr(l_{\text{next}}|l_{\text{initial}} = l),$$

where $\text{Mult}(\cdot|\mathbf{r})$ is the multinomial distribution with probabilities $(r_1, \ldots, r_{|L_{i_{\text{res}}}|})$ and $\Pr(l_{\text{next}}|l_{\text{initial}} = l) = \text{DPTRAN}_{i_{\text{res}}}[l]$. This essentially models the probability distribution for the next POI, given a stay at l_{initial} with the probability $\Pr(l_{\text{initial}}|\mathbf{r})$. The mixing ratio is generated by sampling from a Dirichlet distribution $\mathbf{r} \sim \text{Dirichlet}(\mathbf{a})$, where $\mathbf{a} = (1, 1, \ldots, 1)$ and $|\mathbf{a}| = |L_{i_{\text{res}}}|$. Note that the augmented transition matrix is still DP due to the post processing property.

For the model derived probability distribution, we incorporate the mixing ratio $\mathbf{r}$ into the location encoding as follows:

$$x = \sum_{l \in L_{i_{\text{res}}}} \text{Mult}(l_{\text{initial}} = l|\mathbf{r}) \text{hiencode}_{i_{\text{res}}}(l).$$

This is the weighted sum of the encoding of region $l \in L_{i_{\text{res}}}$ with ratio $\mathbf{r}$. Then, we use the mixed encoding x as input to the temporal component (i.e., Equation (5)) to derive the key vector and then the output distribution. Finally, the loss is

$$\text{loss} = \text{KL}(\Pr(l_{\text{next}}|\mathbf{r}) || \hat{y}|\mathbf{r}), \tag{6}$$

where $\hat{y}|\mathbf{r}$ is the output distribution derived from the key vector. With this objective function, our model not only learns the probability distributions of DPTRAN but also learns from a continuous representation among these probability distributions.

4.2.3 Discussion. In summary, HRNet is pretrained using DPTRAN (Equation 4) based on the KL loss (Equation 6). Here, we further explain the rationale behind this choice from two perspectives:

Warm Start. One challenge of DP-SGD is that it requires many iterations due to starting from a cold state. A warm start helps to mitigate this issue [3], and we anticipate this benefit in our approach. This is because when the KL loss during pre-training is optimized, it also helps to optimize the cross-entropy loss during main training due to the multi-task learning. It is important to note that optimizing cross-entropy loss is essentially the same as optimizing KL loss. Although marginalization to generate DPTRAN means the KL loss is not exactly the original cross-entropy loss, we expect it to be similar enough to provide a warm start.

Differential Privacy Efficiency. There are several alternatives for pretraining beyond the first-order transition matrix DPTRAN, such as second-order transitions and transitions that include time information. Additionally, we could use a sophisticated and nonuniform grid based on density, similar to privtree [66], for the initial grid of DPTRAN. As shown by Zhou et al. [70], using public data to find the gradient subspace addresses the issue of a large number of parameters and improves noise stability. Therefore, a more precise DPTRAN as mentioned above could potentially enhance the main training results. However, these methods typically require additional privacy budget. From our experience, we found that using first-order transition and a fixed resolution ($i_{\text{res}} = 2$) is more beneficial than these complex methods. First-order transitions are denser

than second-order transitions and those with time information, making the perturbed information more useful for pre-training. A fixed resolution allows us to allocate a larger portion of the privacy budget to perturb the transition matrix, which provides better information for pre-training. Moreover, a fixed resolution enables algorithmic privacy budget allocation (see Section 4.3.1) without seeing raw data.

Although the selection of first-order transition and ($i_{\text{res}} = 2$) was heuristically made based on the principles mentioned above, it is not theoretically clear why this approach is effective. Understanding what type of information or loss contributes to the main task is challenging, and we believe this study provides some insights. Investigating the types of information that are beneficial for pre-training is an interesting direction for future research.

4.3 Privacy Analysis of HRNet with Pre-training

We analyze the privacy guarantees of HRNet when pre-trained using the DP transition matrix.

Theorem 4.1. *Given a dataset D, privacy parameters $\varepsilon_1, \varepsilon_2 \in \mathbb{R}^+$ and $\delta \in [0, 1]$, and initial parameters of HRNet θ, the pre-training and training process for HRNet is as follows:*

$$\theta' = \text{pretrain}(\theta, \text{dptran}(D, \varepsilon_2)). \tag{7}$$

$$\theta'' = \text{DPSGD}(D, \theta', \varepsilon_1, \delta). \tag{8}$$

Here, $\text{dptran}(D, \varepsilon_2)$ *refers to the computation of Equation (4) and* pretrain *is the standard training with loss function (6).* DPSGD *is the private training algorithm described in Section 2.1, utilizing the allocated privacy parameters (ε_1, δ), and initial parameter θ'. Deriving the trained parameters for HRNet (i.e., θ'') satisfies $(\varepsilon_1 + \varepsilon_2, \delta)$-DP.*

Proof. We can see that the entire computation is the sequential composition of Equation (7) and Equation (8). For Equation (7) (i.e., the pre-training phase), the raw data D is employed solely for generating DPTRAN (i.e., Equation (4)). The application of the Laplace mechanism, which is known to satisfy ε_2-DP [17], ensures that the pre-training phase is ε_2-DP. According to the post-processing theorem [18], computing θ' satisfies ε_2-DP since it does not further utilize D. For Equation (8) (i.e., the training phase), the raw data D is used for DPSGD and DPSGD satisfies (ε_1, δ)-DP (see Section 2.1 for detail). By applying the composition theorem of DP [18], deriving θ'' satisfies $(\varepsilon_1 + \varepsilon_2, \delta)$-DP. □

4.3.1 Privacy Budget Allocation. Based on the analysis above, we have demonstrated that HRNet with pre-training satisfies $(\varepsilon_1+\varepsilon_2, \delta)$-DP. A critical challenge arises in optimally allocating the given privacy budget ε between ε_1 and ε_2 due to the difficulty in hyperparameter selection, such as cross-validation, under the DP constraint. An excessive allocation to ε_2 (thus reducing ε_1) may lead to premature depletion of the privacy budget for DP-SGD during model training, potentially hindering the model's convergence and impairing its final performance. Conversely, a small ε_2 risks yielding an insufficient pre-trained model, thereby diminishing its effectiveness in supporting DP-SGD. Striking a balance in privacy budget distribution is therefore crucial for enhancing the overall performance of the model. This section proposes a heuristic solution to this budget allocation challenge.

Solution. Assume that we use the i_{res}th resolution for the initial region of the DP transition matrix. We propose the following allocation formula:

$$\varepsilon_2 = \min\left(\frac{cw^2 4^{i_{\text{res}}} \log(w)}{|D|}, \varepsilon\right), \varepsilon_1 = \varepsilon - \varepsilon_2. \tag{9}$$

This solution is derived (see Appendix in the full version [53]) in order to maintain a constant signal-to-noise ratio (SNR) in the DP transition matrix, based on the dataset's meta-information (the number of records $|D|$ and the parameter that decides the number of grid cells w) which is not sensitive, thus publicly available. Our aim is to consistently ensure a minimum quality of the DP transition matrix for effective pre-training.

5 EXPERIMENTS

Our experiments aim to evaluate the efficacy of HRNet in generating human mobility data under DP constraints. We first compare the privacy-utility trade-off of our approach with existing state-of-the-art methods. Additionally, we conduct an ablation study to assess the impact of our novel components: hierarchical location encoding, multi-task learning, and private pre-training.

5.1 Setup

5.1.1 datasets: We utilize five datasets, broadly classified into three categories: human mobility, taxi trajectory (road network), and synthetic data. Human mobility encompasses a wide range of unrestricted mobility patterns, while road network data is constrained by road networks. The datasets employed are Geolife [69] (human mobility), Peopleflow[3] (human mobility), Didi in Chengdu [60] (taxi trajectory), and two synthetic datasets. The synthetic datasets are used in the ablation study and discussed in detail in Section 5.4. We use randomly sampled 10,000 trajectories from each dataset as the training dataset. See details of the datasets in Appendix in the full version [53].

5.1.2 Preprocessing: In the preprocessing phase, our approach centers on the concept of *stay points*, as defined by Li et al. [34]. A stay point is identified when an individual remains within a radius of m meters for a duration exceeding t minutes. To prepare our dataset, we first process each trajectory to identify these stay points. The identified stay points are then regarded as critical waypoints, representing significant stops or areas of interest in the movement patterns of individuals. By focusing on these waypoints, we can distill the essence of each trajectory, thereby capturing the most relevant and informative aspects of human mobility. Note that our preprocessing inevitably leads to a loss of detailed information, such as minor route variations and brief stops.

5.1.3 Competitors. We compare our method with a baseline and three state-of-the-art methods in distinct categories as outlined in Introduction:

- **Baseline**: implements the generator as described in Section 2.3 with DP-SGD.

[3] http://pflow.csis.u-tokyo.ac.jp/home/

- **PrivTrace [58]**: a state-of-the-art Markov-based approach leveraging Markov models for mobility data analysis.
- **Clustering [41]**: a generative method with mobility data clustering using k-means.
- **MTNet [60]**: a deep learning method specifically tailored for road network data.

We excluded tree-based methods such as DPT [24] from our comparison. This decision was based on reports [41, 58] from the aforementioned studies, which indicated superior performance of the Markov and the clustering-based methods over tree-based approaches. For our implementation, we relied on publicly available open-source codes for PrivTrace and MTNet.

5.1.4 Evaluation metrics: To compare the quality of generated mobility data, we evaluate the discrepancies between generated and original stay-point trajectories (denoted as $\mathbf{v}$) across various metrics. We employ the Jensen-Shannon (JS) divergence and average relative error (ARE) for measuring discrepancy (both the lower the better), with details provided in the Appendix in the full version [53]. The key metrics considered are as follows which are all probability distributions. **Waypoint** captures which waypoints are traversed given a starting point. **Destination** captures which location serves as the final destination given a starting point. **Transition** captures where the trajectory moves next from a starting point. **Travel distance** represents the total length of a trajectory. **Diameter** measures the maximum distance between any two points along a trajectory. **Route** captures the route taken from a starting point. **Density at** t is the probability of a trajectory being at a specific location at time t. **Trajectory density** is the probability of a trajectory passing through a specific area. **Trajectory pattern** captures the most frequent transition patterns.

5.2 Comparison with Baseline, PrivTrace, and Clustering

In this section, we present a comparative evaluation of HRNet against existing approaches for human mobility not constrained by road networks. The competitors in this analysis include the Baseline, PrivTrace, and Clustering methods. The used datasets are the Geolife dataset and the Peopleflow dataset.

5.2.1 Main result. Table 1 showcases the main result, where we assess the discrepancy across four metrics with a fixed privacy budget ($\varepsilon = 2$). Notably, our proposed method demonstrates significant superiority across all metrics compared to the other approaches. The statistical-based methods, such as Clustering and PrivTrace, inherently struggle with a high signal-to-noise ratio due to the need to add noise to sparse data. Conversely, deep learning methods have the potential to uncover hidden characteristics of POIs, moving beyond mere sparse information. The Baseline method experiences a decrease in utility, largely due to its incompatibility with DP-SGD, as it still relies on learning sparse information through an embedding matrix. Our approach, incorporating a deconvolutional network, multi-task training, and private pre-training, effectively learns dense, hidden characteristics in a manner that harmonizes with DP-SGD.

5.2.2 Utility-privacy trade-off. This subsection delves into the utility-privacy trade-off of HRNet in comparison to other competitors, achieved by varying the privacy budget. The results depicted in Figures 3 and 4 illustrate this analysis. It is observed that at lower privacy budgets (e.g., $\varepsilon < 1$), PrivTrace and clustering sometimes exhibit better performance. This is attributed to their direct computation of transitions.

In contrast, deep learning methods generally require extensive training epochs to learn the hidden features of human mobility. However, our method enhances the learning process through pre-training and multi-task learning, effectively reducing the need for a larger privacy budget, unlike the baseline method. Furthermore, as epsilon increases, deep learning approaches increasingly outperform their counterparts. This advantage stems from their direct learning of raw data, avoiding the utility loss associated with sensitivity bounding, a common issue in statistical-based methods like clustering and PrivTrace.

5.3 Comparison with MTNet

MTNet [60] represents a leading deep learning methodology designed specifically for generating short trips within road networks. The architecture of MTNet necessitates that trajectories be confined to connected road networks. This requirement is incompatible with datasets like Geolife and Peopleflow, which encompass mobility patterns extending beyond road networks, including activities like train travel or park traversing. Hence, for a comparative analysis with MTNet, we utilize the Didi dataset.

Our results, as illustrated in Figures 5 and 6, focus on evaluating destination and route discrepancies. For both metrics, MTNet initially exhibits superior performance, primarily because it generates trajectories using road network metadata. However, as epsilon increases, our method demonstrates improved results. This discrepancy arises from the differing objectives of the training targets. Our method is tailored to learn semantically significant next waypoints. Conversely, MTNet is designed to directly learn the next road segment. As a result, MTNet is required to learn a longer sequence which is difficult to learn under DP constraints. Hence, our method has better convergence in the case where DP-SGD is applied.

5.4 Ablation study

In this section, we conduct an ablation study to elucidate the individual contributions of the key components of HRNet: the hierarchical location encoding via deconvolutional network, multi-resolution multi-task learning, and private pre-training. Our analysis begins with the Geolife dataset, followed by an examination using synthetic datasets, each with distinct characteristics.

Random Dataset: The Random dataset comprises of trajectories, each with a fixed length 2. The initial and second locations in these trajectories are randomly selected from all POIs, forming a total of $10,000$ trajectories. This dataset is characterized by completely random empirical transition distributions, effectively disregarding the first law of geography.

Straight Dataset: Each trajectory in the Straight dataset is of fixed length 3. The first location is chosen from even columns on a grid (i.e., $L_1 = \{l \in [w * w] || \lfloor l/w \rfloor / 2 = 0\}$). Subsequently, the second and third locations are determined as the next two rows above in the same column ($l_2 = l_1 + w$ and $l_3 = l_2 + w$). This study creates $10,000$ trajectories.

Table 1: The discrepancy of the generated dataset to the real dataset (Geolife / Peopleflow) with $w = 32$ when $\varepsilon = 2$ and $\delta = 10^{-5}$.

	JS Divergence ↓						ARE ↓	
	waypoint	destination	transition	travel distance	diameter	density_t	traj density	traj pattern
Clutering	9.92 / 12.9	0.438 / 0.650	0.422 / 0.542	0.0227 / 0.154	0.0921 / 0.133	0.123 / 0.0813	0.233 / 0.852	0.701 / 0.772
PrivTrace	12.6 / 11.0	0.274 / 0.649	0.245 / 0.431	0.0476 / 0.0181	0.0712 / 0.0531	-	0.212 / 0.835	0.712 / 0.781
Baseline	6.68 / 10.1	0.397 / 0.670	0.397 / 0.486	0.0275 / 0.151	0.0932 / 0.303	0.115 / 0.0512	0.382 / 0.811	0.771 / 0.912
HRNet	5.67 / 6.17	0.192 / 0.586	0.212 / 0.309	0.00871 / 0.0569	0.0681 / 0.183	0.0523 / 0.0425	0.184 / 0.562	0.671 / 0.743

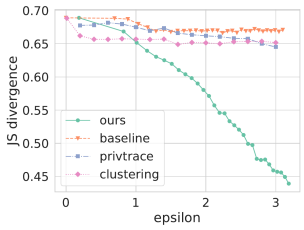

Figure 3: The discrepancy of destination on Peopleflow dataset ($w = 64$) for each $\varepsilon \in [0, 3.0]$.

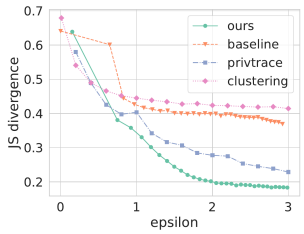

Figure 4: The discrepancy of transition on Geolife dataset ($w = 32$) for each $\varepsilon \in [0, 3.0]$.

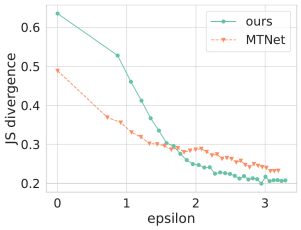

Figure 5: The discrepancy of destination on Didi dataset ($w = 32$) for each ε and $\delta = 10^{-5}$.

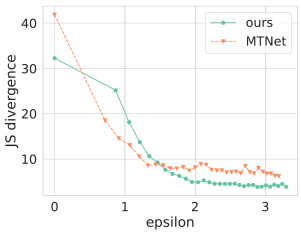

Figure 6: The discrepancy of route on Didi dataset ($w = 32$) for each ε and $\delta = 10^{-5}$.

The ablation study is structured around six configurations, combining the elements of deconvolutional network, multi-task training, and private pre-training:

- **Baseline:** baseline generator with DP-SGD.
- **Pre-training:** baseline + private pre-training (Section 4). Unlike our complete model, the location encoding component is not pre-trained here due to the absence of coarse region embeddings.
- **Deconvolutional Network:** the location encoding component of the baseline is replaced with a 2D quad deconvolutional network.
- **Deconvolutional Network + Pre-training:** this combines private pre-training with the deconvolutional network setup.
- **Deconvolutional Network + Multi-task Training:** this integrates multi-task training (Section 3.4) with the deconvolutional network framework.
- **Deconvolutional Network + Multi-task Training + Pre-training (HRNet):** the complete solution.

Through this study, we aim to isolate and understand the contribution of each component to the overall efficacy of our method in capturing and replicating human mobility patterns.

5.4.1 Real data. The evaluation on the Geolife dataset with a grid size of $w = 64$ is depicted in Figure 7. The results indicate that the incorporation of each component – the deconvolutional network, multi-task learning, and pre-training – not only enhances the accuracy in terms of discrepancy reduction but also accelerates convergence.

The deconvolutional network, in particular, contributes to improvement of discrepancy by offering an efficient parameterization. Its architecture supports the integration of multi-task learning and

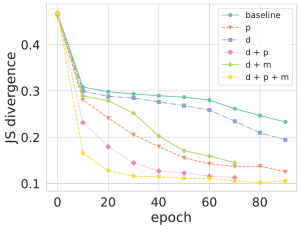

Figure 7: The discrepancy of transition on Geolife ($w = 64$). p, d, and m represent pre-raining, deconvolutional network, and multi-task learning, respectively.

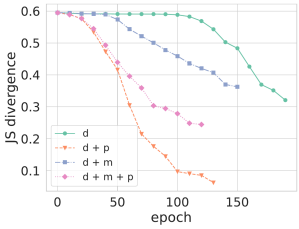

Figure 8: The discrepancy of transition on Random dataset ($w = 32$). d, m, and p mean deconvolutional network, multi-task learning, and pre-training, respectively.

Table 2: The number of parameters with varying grid size w.

	$w = 8$	$w = 16$	$w = 32$	$w = 64$
baseline	14, 054	26, 534	76, 454	276, 134
ours	24, 710	28, 934	41, 606	47, 942

pre-training while providing benefits even when deployed independently. Multi-task learning improves the convergence speed with DP-SGD by incorporating simpler tasks as observed in the study by Dockhorn et al.[15]. Pre-training, on the other hand, offers a substantial boost in convergence speed through a warm start and enhances utility, aligning with findings by Amid et al. [3].

5.4.2 Deconvolutional Network. The deconvolutional network not only enables multi-task learning and private pre-training but also

Table 3: Results for next location prediction and trajectory classification. We used the same privacy budget for the baseline model and HRNet ($\varepsilon = 3.0, \delta = 10^{-5}$).

		Acc@1	Acc@5	F-score	AUROC
next loc pred	baseline w/o DP	29.7	45.1	40.9	84.5
	baseline	13.3	30.6	25.2	60.3
	HRNet	23.1	37.7	30.2	70.2
traj class	Random Forest	78.2	-	74.3	92.6
	baseline w/o DP	73.4	-	70.1	91.3
	baseline	48.3	-	47.8	72.1
	HRNet	58.3	-	55.9	78.7

reduces the number of parameters. As mentioned in Section 2.4, it is known that the lower bound of empirical risk increases as the number of parameters increases [4]. Therefore, the reduced number of parameters is a crucial advantage of the deconvolutional network, and we explore this aspect here. Table 2 compares the number of parameters with the baseline architecture (i.e., matrix embedding). The number of parameters in HRNet depends logarithmically on the number of POIs, in contrast to the baseline mechanism, which depends linearly. As a result, even if w becomes larger, the number of parameters remains stable. This difference theoretically improves stability, and this is supported by the experimental results shown in Figure 7.

5.4.3 Multi-task learning. HRNet benefits from the efficient utilization of inferences derived from simpler tasks, which align with the first law of geography. It is important to note that bias based on the first law of geography can also have a negative impact. To further substantiate this point, we investigate a scenario where the first law of geography is not applicable, utilizing the Random dataset for this purpose. The results of this investigation are presented in Figure 8. Contrary to previous observations, the addition of multi-task learning in this context results in a decrease in utility. This decline can be attributed to the inductive bias towards the first law of geography, which, in the case of the Random dataset, leads to a misalignment with the actual data characteristics. Consequently, the multi-task learning framework, while beneficial under certain conditions, may impact the results negatively when the underlying data does not adhere to the geographic principles it assumes.

Even for datasets that generally follow the first law of geography, it is possible for some grid cells to not satisfy this law. For example, a grid cell near the border of a coarser cell tends to differ from the characteristics of the coarser cell because people in this grid cell often move to the adjacent coarser cell, whereas people in grid cells near the center of the coarser cell tend to stay within the same coarser cell. It is important to note that some cells may suffer due to this bias, even though it globally improves performance. Solving this aspect further is an interesting direction for future research.

5.5 Application Case Study

5.5.1 Next location prediction. Next location prediction or recommendation is crucial for various applications such as smart cities, tourism, and advertising. In this section, we evaluate our model's ability to predict the next location. To ensure a more realistic setting, we preprocess the Geolife dataset with the notion of significant locations [26, 46, 64] to make scatterd POIs, instead of a grid-based approach. Details on embedding scattered POIs instead of grid cells can be found in Section 3.2.2. This preprocessing results in 734 POIs and 9,816 trajectories, which are split into a training set and a test set with a ratio of 0.9 to 0.1. We trained generative models on the training dataset and performed next location inference on the test dataset. Table 3 presents the results. Acc@k represents the accuracy when the correct answer is included in the top-k predictions, and AUROC is the area under the receiver operating characteristic curve. The baseline w/o DP is trained using SGD instead of DP-SGD. We spent the same privacy budget ($\varepsilon = 3, \delta = 10^{-5}$) to both the baseline and HRNet. The results show that while the baseline model significantly loses utility due to noise, HRNet outperforms the baseline, demonstrating its robustness to noise in the case of scattered POIs.

5.5.2 Trajectory classification. Trajectory classification involves training a model to categorize trajectories into different types. Here, we consider classification of transportation modes, a well-known problem in trajectory classification [40, 61, 68]. Some of trajectories in the Geolife dataset are labeled as bike, walk, car, bus, and train. We create 5,980 trajectories using grid-based discretization with $w = 32$, splitting them into a training set and a test set with a 0.9 to 0.1 ratio. Using the training set, we first train a generative model (HRNet or the baseline) to synthesize trajectories. Here, the label information is incorporated into the generative model by extending Equation 2 as follows: $\text{encode}(v_i) = [M_{\text{time}}[t_i], M_{\text{POI}}[l_i], M_{\text{label}}[s]]$, where M_{label} is the embedding matrix for labels and s represents the label index. This encoding method generates trajectories considering the label information. With this generated labeled data, we extract features following the method proposed by Liao et al. [36] and train a random forest classifier using these features. The lower part of Table 3 shows the classification results on the test dataset. The Random Forest results represent the model trained with raw data. Compared to the baseline, our model performs better, but there is still a significant performance drop. This is because trajectory classification requires maintaining spatio-temporal correlations, making it more challenging than next location prediction. Additionally, our model uses discretized time information with the embedding matrix, and the preprocessing required for embedding inherently leads to some information loss.

5.5.3 Commuting distance analysis. Analyzing commuting patterns is a real-world application, as conducted by the U.S. Census Bureau [45]. In this study, similar to the work by Machanavajjhala [45], we conducted a commuting distance analysis on the generated data. We utilized trajectories labeled as "commuting" from the peopleflow data. We created 20,000 trajectories using grid-based discretization with $w = 32$ for these commuting trajectories. We then trained a generative model (HRNet or the baseline model) to generate synthetic commuting trajectories, and analyzed them. Figures 9 show the results. Each point in the figures corresponds to a specific destination cell. The x-axis shows the number of trajectories ending in that destination cell, and the y-axis shows the average commute distance to that destination block. For the baseline model (the left figure), many average commute distances are longer than

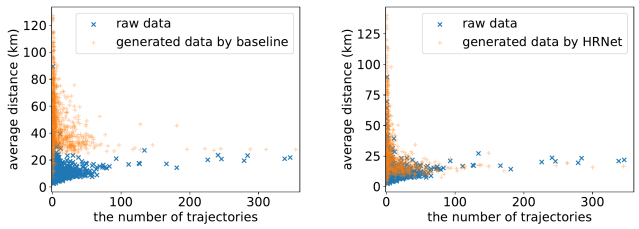

Figure 9: Average commuting distance for each destination by the baseline model (left) and HRNet (right).

the original average commute distances, especially for destinations with a smaller number of trajectories. In contrast, our generated trajectories (the right figure) result in average commute distances that are similar to the original ones, even for destinations with a smaller number of trajectories.

6 RELATED WORK

6.1 Non-deep Learning Models

Clustering. Clustering-based methods [27, 32] first cluster the locations to reduce the number of locations as a preprocessing step. Then, they count the transitions in the smaller location domain (i.e., classes) to create a probabilistic model with the Laplace mechanism. Many clustering-based methods [27, 32] have been found to cause unexpected privacy leakage in the data-dependent preprocessing due to flaws in the privacy proof [47]. Instead, data-dependent preprocessing and the use of the Laplace mechanism, which is very effective with respect to privacy-utility trade-off, can sometimes perform well with small ε values (see Figure 3). However, clustering based preprocessing leads to substantial information loss, and even with large ε values, accuracy cannot be significantly improved.

Tree and Markov models. Tree [10, 24] and Markov [23, 58] methods first construct a data structure (e.g., prefix tree) for trajectories. Then, they count elements (e.g., nodes) to create the probabilistic model. Essentially, tree and Markov methods face two problems due to the high uniqueness of patterns in trajectory: how to define locations (e.g., determining the grid size w) and how to resolve sparsity. Since the initial study by Chen et al. [10], subsequent studies have attempted to address these issues. In the work [11], substrings are considered for building an exploration tree based on a Markov assumption, resulting in higher leaf counts and better utility. DPT [24] addresses the sparsity problem by restricting movement to adjacent cells using a hierarchical structure, effectively capturing short trajectory features. However, this method's limitation to adjacent cell movement prevents conversion to stay point trajectories, resulting in longer sequence lengths. Given DP's characteristics, performance significantly degrades with longer sequence lengths, making it unsuitable for capturing broad patterns such as daily human movement. PrivTrace [58] addresses the sparsity problem heuristically, yielding good results on some datasets, but there remains the problem of determining constants used in heuristics. Our method addresses them using hierarchical networks and multi-task learning.

6.2 Deep Learning Models

As argued in this study, when using DP-SGD, sparsity emerges as a significant issue in deep learning, and thus not many studies propose using DP-SGD. Ahuja et al. [2] propose using negative sampling and skip-gram to solve this sparsity issue. Note that such techniques could also be applicable to our study. Wang et al. [60] address this problem similarly to DPT by restricting movement to adjacent road network segments. Hence, as demonstrated in Figure 5, they effectively capture short trajectory features and perform well with small ε due to prior road network information. However, the same problem as DPT arises, making it unsuitable for capturing broad daily human movement patterns. Without the formal guarantees like DP, many deep learning approaches [22, 48, 51] rely on obfuscation caused by the generation itself. However, without any formal guarantee, it is unclear if privacy is genuinely preserved [52]. Even without privacy-aware considerations, sparsity remains a problem, and there are studies proposing models similar to ours. Lim et al. [37] address sparsity by employing multi-task learning with multiple resolutions, but their naive creation of embedding matrices for each resolution results in a large number of parameters ($O(n^2)$), which is not suitable to DP-SGD. Lian et al. [35] adopt a hierarchical location encoding using a quadtree similar to our study, but instead of using a deconvolutional network, they treat quadtree paths as sequences and perform sequence embedding with the attention architecture [57]. Consequently, encoding a single location requires processing a sequence length corresponding to the quadtree depth. Longer sequence lengths degrade DP-SGD performance, which is not ideal. Moreover, the application of multi-task learning to such methods is not trivial. Many deep learning methods for capturing trajectory features have been considered [20, 38, 44, 63]. However, directly encoding the tuple of latitude and longitude is fundamentally challenging [35], leading to the use of embedding matrices. Using embedding matrices poses the same issues as the baseline.

7 CONCLUSION

In this paper, we introduced HRNet, a novel framework designed to effectively learn and generate human mobility data under DP constraints. HRNet incorporates three novel components: a hierarchical location encoding component, multi-resolution multi-task learning, and private pre-training. The concept of a hierarchical network, as employed in HRNet, has intrinsic value beyond its application in location encoding. The hierarchical nature of data is not unique to locations but is also evident in language for example, where words possess semantic layers. This observation opens up possibilities for applying our hierarchical approach to improve language models under DP constraints, suggesting a promising avenue for future research in the field of privacy-preserving deep learning.

ACKNOWLEDGMENTS

This work was supported by the support of JST SICORP JPMJSC2107, JST CREST JPMJCR21M2, JST PRESTO JPMJPR23P5, JSPS KAKENHI JP22H03595, JP23K24851, JP21K19767, NSF CNS-2125530, CNS-2124104, IIS-2302968, and CDC 1NU38FT000001-01-00.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. 308–318.

[2] Ritesh Ahuja, Gabriel Ghinita, and Cyrus Shahabi. 2020. Differentially-private next-location prediction with neural networks. In Advances in database technology. 121–132.

[3] Ehsan Amid, Arun Ganesh, Rajiv Mathews, Swaroop Ramaswamy, Shuang Song, Thomas Steinke, Vinith M Suriyakumar, Om Thakkar, and Abhradeep Thakurta. 2022. Public data-assisted mirror descent for private model training. In International Conference on Machine Learning. PMLR, 517–535.

[4] Raef Bassily, Vitaly Feldman, Cristóbal Guzmán, and Kunal Talwar. 2020. Stability of stochastic gradient descent on nonsmooth convex losses. Advances in Neural Information Processing Systems 33 (2020), 4381–4391.

[5] Raef Bassily, Adam Smith, and Abhradeep Thakurta. 2014. Private empirical risk minimization: Efficient algorithms and tight error bounds. In 2014 IEEE 55th annual symposium on foundations of computer science. IEEE, 464–473.

[6] Claire McKay Bowen and Fang Liu. 2021. Comparative study of differentially private data synthesis methods. Journal of Pivacy and Confidentiality 11-1 (2021), 280–307.

[7] USC Bureau. 2020. On the map: Longitudinal employer-household dynamics, https://lehd.ces.census.gov/applications/help/onthemap.html#confidentiality_protection.

[8] Sujin Cai, Xin Lyu, Xin Li, Duohan Ban, and Tao Zeng. 2021. A trajectory released scheme for the Internet of Vehicles based on differential privacy. IEEE Transactions on Intelligent Transportation Systems 23, 9 (2021), 16534–16547.

[9] Serina Chang, Emma Pierson, Pang Wei Koh, Jaline Gerardin, Beth Redbird, David Grusky, and Jure Leskovec. 2021. Mobility network models of COVID-19 explain inequities and inform reopening. Nature 589, 7840 (2021), 82–87.

[10] Rui Chen, Gergely Acs, and Claude Castelluccia. 2012. Differentially private sequential data publication via variable-length n-grams. In Proceedings of the 2012 ACM conference on Computer and communications security. 638–649.

[11] Rui Chen, Benjamin CM Fung, Noman Mohammed, Bipin C Desai, and Ke Wang. 2013. Privacy-preserving trajectory data publishing by local suppression. Information Sciences 231 (2013), 83–97.

[12] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). Association for Computational Linguistics, Doha, Qatar, 1724–1734. https://doi.org/10.3115/v1/D14-1179

[13] DF Crouse. 2016. On implementing 2D rectangular assignment algorithms. Transactions on Aerospace and Electronic Systems 52, 4 (2016), 1679–1696.

[14] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. 2017. Collecting telemetry data privately. In Advances in Neural Information Processing Systems. 3571–3580.

[15] Tim Dockhorn, Tianshi Cao, Arash Vahdat, and Karsten Kreis. 2022. Differentially private diffusion models. arXiv preprint arXiv:2210.09929 (2022).

[16] Cynthia Dwork. 2006. Differential privacy. In Proceedings of the 33rd international conference on Automata, Languages and Programming-Volume Part II. Springer-Verlag, 1–12.

[17] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In Theory of cryptography conference. Springer, 265–284.

[18] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. Foundations and Trends in Theoretical Computer Science 9, 3-4 (2014), 211–407.

[19] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In Proceedings of the 2014 ACM SIGSAC conference on computer and communications security. 1054–1067.

[20] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. 2018. Deepmove: Predicting human mobility with attentional recurrent networks. In Proceedings of the 2018 world wide web conference. 1459–1468.

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

[22] Ivan Fontana, Marc Langheinrich, and Martin Gjoreski. 2023. GANs for Privacy-Aware Mobility Modeling. IEEE Access 11 (2023), 29250–29262.

[23] Mehmet Emre Gursoy, Ling Liu, Stacey Truex, and Lei Yu. 2018. Differentially private and utility preserving publication of trajectory data. IEEE Transactions on Mobile Computing 18, 10 (2018), 2315–2329.

[24] Xi He, Graham Cormode, Ashwin Machanavajjhala, Cecilia Procopiuc, and Divesh Srivastava. 2015. DPT: differentially private trajectory synthesis using hierarchical reference systems. Proceedings of the VLDB Endowment 8, 11 (2015), 1154–1165.

[25] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. Neural computation 9, 8 (1997), 1735–1780.

[26] Ye Hong, Yatao Zhang, Konrad Schindler, and Martin Raubal. 2023. Context-aware multi-head self-attentional neural network model for next location prediction. Transportation Research Part C: Emerging Technologies 156 (2023), 104315.

[27] Jingyu Hua, Yue Gao, and Sheng Zhong. 2015. Differentially private publication of general time-serial trajectory data. In 2015 IEEE Conference on Computer Communications (INFOCOM). IEEE, 549–557.

[28] Noah Johnson, Joseph P Near, and Dawn Song. 2018. Towards practical differential privacy for SQL queries. Proceedings of the VLDB Endowment 11, 5 (2018), 526–539.

[29] Peter Kairouz, Monica Ribero Diaz, Keith Rush, and Abhradeep Thakurta. 2021. (Nearly) Dimension Independent Private ERM with AdaGrad Rates via Publicly Estimated Subspaces. In Conference on Learning Theory. PMLR, 2717–2746.

[30] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. 2015. The composition theorem for differential privacy. In International conference on machine learning. PMLR, 1376–1385.

[31] Antti Koskela, Joonas Jälkö, and Antti Honkela. 2020. Computing tight differential privacy guarantees using fft. In International Conference on Artificial Intelligence and Statistics. PMLR, 2560–2569.

[32] Meng Li, Liehuang Zhu, Zijian Zhang, and Rixin Xu. 2017. Achieving differential privacy of trajectory data publishing in participatory sensing. Information Sciences 400 (2017), 1–13.

[33] Ninghui Li, Zhikun Zhang, and Tianhao Wang. 2021. DPSyn: Experiences in the NIST Differential Privacy Data Synthesis Challenges. Journal of Privacy and Confidentiality 11-2 (2021).

[34] Quannan Li, Yu Zheng, Xing Xie, Yukun Chen, Wenyu Liu, and Wei-Ying Ma. 2008. Mining User Similarity Based on Location History. In Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (Irvine, California) (GIS '08). Association for Computing Machinery, New York, NY, USA, Article 34, 10 pages. https://doi.org/10.1145/1463434.1463477

[35] Defu Lian, Yongji Wu, Yong Ge, Xing Xie, and Enhong Chen. 2020. Geography-aware sequential location recommendation. In Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. 2009–2019.

[36] Lin Liao, Donald J Patterson, Dieter Fox, and Henry Kautz. 2007. Learning and inferring transportation routines. Artificial intelligence 171, 5-6 (2007), 311–331.

[37] Nicholas Lim, Bryan Hooi, See-Kiong Ng, Yong Liang Goh, Renrong Weng, and Rui Tan. 2022. Hierarchical multi-task graph recurrent network for next poi recommendation. In Proceedings of the 45th international ACM SIGIR conference on Research and development in Information Retrieval. 1133–1143.

[38] Nicholas Lim, Bryan Hooi, See-Kiong Ng, Xueou Wang, Yong Liang Goh, Renrong Weng, and Jagannadan Varadarajan. 2020. STP-UDGAT: Spatial-temporal-preference user dimensional graph attention network for next POI recommendation. In Proceedings of the 29th ACM International conference on information & knowledge management. 845–854.

[39] Ziqian Lin, Jie Feng, Ziyang Lu, Yong Li, and Depeng Jin. 2019. Deepstn+: Context-aware spatial-temporal neural network for crowd flow prediction in metropolis. In Proceedings of the AAAI conference on artificial intelligence, Vol. 33. 1020–1027.

[40] Hongbin Liu, Hao Wu, Weiwei Sun, and Ickjai Lee. 2019. Spatio-temporal GRU for trajectory classification. In 2019 IEEE International Conference on Data Mining (ICDM). IEEE, 1228–1233.

[41] Qi Liu, Juan Yu, Jianmin Han, and Xin Yao. 2021. Differentially private and utility-aware publication of trajectory data. Expert Systems with Applications 180 (2021), 115120.

[42] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition. 3431–3440.

[43] Massimiliano Luca, Gianni Barlacchi, Bruno Lepri, and Luca Pappalardo. 2023. A Survey on Deep Learning for Human Mobility. ACM Comput. Surv. 55, 2 (2023), 7:1–7:44. https://doi.org/10.1145/3485125

[44] Yingtao Luo, Qiang Liu, and Zhaocheng Liu. 2021. STAN: Spatio-Temporal Attention Network for next Point-of-Interest Recommendation. WWW. 2177&2185 (2021).

[45] Ashwin Machanavajjhala, Daniel Kifer, John Abowd, Johannes Gehrke, and Lars Vilhuber. 2008. Privacy: Theory meets practice on the map. In 2008 IEEE 24th international conference on data engineering. IEEE, 277–286.

[46] Henry Martin, Ye Hong, Nina Wiedemann, Dominik Bucher, and Martin Raubal. 2023. Trackintel: An open-source Python library for human mobility analysis. Computers, Environment and Urban Systems 101 (2023), 101938. https://doi.org/10.1016/j.compenvurbsys.2023.101938

[47] Àlex Miranda-Pascual, Patricia Guerra-Balboa, Javier Parra-Arnau, Jordi Forné, and Thorsten Strufe. 2023. SoK: Differentially private publication of trajectory data. Proceedings on Privacy Enhancing Technologies (2023).

[48] Ren Ozeki, Haruki Yonekura, Hamada Rizk, and Hirozumi Yamaguchi. 2023. Balancing privacy and utility of spatio-temporal data for taxi-demand prediction. In 2023 24th IEEE International Conference on Mobile Data Management (MDM). IEEE, 215–220.

[49] Roberto Pellungrini, Luca Pappalardo, Francesca Pratesi, and Anna Monreale. 2018. Analyzing privacy risk in human mobility data. In Software Technologies: Applications and Foundations: STAF 2018 Collocated Workshops, Toulouse, France, June 25-29, 2018, Revised Selected Papers. Springer, 114–129.

[50] David Sommer, Sebastian Meiser, and Esfandiar Mohammadi. 2018. Privacy loss classes: The central limit theorem in differential privacy. Cryptology ePrint Archive (2018).

[51] Yeji Song, Jihwan Shin, Jinhyun Ahn, Taewhi Lee, and Dong-Hyuk Im. 2023. Except-Condition Generative Adversarial Network for Generating Trajectory Data. In International Conference on Database and Expert Systems Applications. Springer, 289–294.

[52] Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. 2022. Synthetic data–anonymisation groundhog day. In 31st USENIX Security Symposium (USENIX Security 22). 1451–1468.

[53] Shun Takagi, Li Xiong, Fumiyuki Kato, Yang Cao, and Masatoshi Yoshikawa. 2024. HRNet: Differentially Private Hierarchical and Multi-Resolution Network for Human Mobility Data Synthesization. arXiv:2405.08043 (2024). https://arxiv.org/abs/2308.12210

[54] A. D. P. Team. 2017. Learning with privacy at scale.

[55] Waldo R Tobler. 1970. A computer movie simulating urban growth in the Detroit region. Economic geography 46, sup1 (1970), 234–240.

[56] Zhen Tu, Fengli Xu, Yong Li, Pengyu Zhang, and Depeng Jin. 2018. A new privacy breach: User trajectory recovery from aggregated mobility data. IEEE/ACM Transactions on Networking 26, 3 (2018), 1446–1459.

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. Advances in neural information processing systems 30 (2017).

[58] Haiming Wang, Zhikun Zhang, Tianhao Wang, Shibo He, Michael Backes, Jiming Chen, and Yang Zhang. 2023. PrivTrace: Differentially Private Trajectory Synthesis by Adaptive Markov Model. In USENIX Security Symposium 2023.

[59] Senzhang Wang, Jiannong Cao, and S Yu Philip. 2020. Deep learning for spatio-temporal data mining: A survey. IEEE transactions on knowledge and data engineering 34, 8 (2020), 3681–3700.

[60] Yong Wang, Guoliang Li, Kaiyu Li, and Haitao Yuan. 2022. A Deep Generative Model for Trajectory Modeling and Utilization. Proceedings of the VLDB Endowment 16, 4 (2022), 973–985.

[61] Zhibin Xiao, Yang Wang, Kun Fu, and Fan Wu. 2017. Identifying different transportation modes from trajectory data using tree-based ensemble classifiers. ISPRS International Journal of Geo-Information 6, 2 (2017), 57.

[62] Xiaodong Yan, Tengwei Song, Yifeng Jiao, Jianshan He, Jiaotuan Wang, Ruopeng Li, and Wei Chu. 2023. Spatio-Temporal Hypergraph Learning for Next POI Recommendation. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. 403–412.

[63] Dingqi Yang, Benjamin Fankhauser, Paolo Rosso, and Philippe Cudre-Mauroux. 2020. Location prediction over sparse user mobility traces using rnns. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence. 2184–2190.

[64] Jie Yang, Jian Xu, Ming Xu, Ning Zheng, and Yu Chen. 2014. Predicting next location using a variable order Markov model. In Proceedings of the 5th ACM SIGSPATIAL International Workshop on GeoStreaming. 37–42.

[65] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. 2017. Privbayes: Private data release via bayesian networks. ACM Transactions on Database Systems (TODS) 42, 4 (2017), 25.

[66] Jun Zhang, Xiaokui Xiao, and Xing Xie. 2016. Privtree: A differentially private algorithm for hierarchical decompositions. In Proceedings of the 2016 international conference on management of data. 155–170.

[67] Minxing Zhang, Haowen Lin, Shun Takagi, Yang Cao, Cyrus Shahabi, and Li Xiong. 2023. CSGAN: Modality-Aware Trajectory Generation via Clustering-based Sequence GAN. In 2023 24th IEEE International Conference on Mobile Data Management (MDM). IEEE, 148–157.

[68] Yu Zheng, Yukun Chen, Quannan Li, Xing Xie, and Wei-Ying Ma. 2010. Understanding transportation modes based on GPS data for web applications. ACM Transactions on the Web (TWEB) 4, 1 (2010), 1–36.

[69] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. 2009. Mining interesting locations and travel sequences from GPS trajectories. In Proceedings of the 18th international conference on World wide web. 791–800.

[70] Yingxue Zhou, Zhiwei Steven Wu, and Arindam Banerjee. 2021. Bypassing the Ambient Dimension: Private SGD with Gradient Subspace Identification. ICLR abs/2007.03813 (2021). https://api.semanticscholar.org/CorpusID:220404588