Trajectory-User Linking via Variational AutoEncoder

Fan Zhou[1†], **Qiang Gao**[1], **Goce Trajcevski**[2], **Kunpeng Zhang**[3], **Ting Zhong**[1], **Fengli Zhang**[1]
[1] School of Information and Software Engineering, University of Electronic Science and Technology of China. {fan.zhou, qianggao@std., zhongting@, fzhang@}uestc.edu.cn
[2] Iowa State University, Ames. gocet25@iastate.edu
[3] University of Maryland, College park. kpzhang@umd.edu

Abstract

Trajectory-User Linking (TUL) is an essential task in Geo-tagged social media (GTSM) applications, enabling personalized Point of Interest (POI) recommendation and activity identification. Existing works on mining mobility patterns often model trajectories using Markov Chains (MC) or recurrent neural networks (RNN) – either assuming independence between non-adjacent locations or following a shallow generation process. However, most of them ignore the fact that human trajectories are often *sparse*, *high-dimensional* and may contain *embedded hierarchical structures*. We tackle the TUL problem with a semi-supervised learning framework, called *TULVAE (TUL via Variational AutoEncoder)*, which learns the human mobility in a neural generative architecture with stochastic latent variables that span hidden states in RNN. TULVAE alleviates the data sparsity problem by leveraging large-scale unlabeled data and represents the hierarchical and structural semantics of trajectories with high-dimensional latent variables. Our experiments demonstrate that TULVAE improves efficiency and linking performance in real GTSM datasets, in comparison to existing methods.

1 Introduction

Geo-tagged social media (GTSM) data, such as ones generated by Instagram, Foursquare and Twitter (to name but a few sources), provides an opportunity for better understanding and use of motion patterns in various applications [Zheng, 2015], such as POI recommendation [Yang *et al.*, 2017a]; next visit-location [Liu *et al.*, 2016]; user interest inference and individual activity modeling [Li *et al.*, 2017a]; etc.

An important aspect, and often an initial component, of many applications based on GTSM data is the *Trajectory-User Linking* (TUL), which links *trajectories* to *users* who generate them. For example, ride-sharing (bike, car) apps generate large volumes of trajectories – but the user identities are unknown – for the sake of privacy. However, correlating such trajectories with corresponding users seems desirable in many real world applications: it might lead to better, more personalized and/or precise recommendations; it may help in identifying terrorists/criminals from sparse spatio-temporal data such as transient check-ins [Gao *et al.*, 2017].

Common approaches for modeling human trajectories rely on Markov Chain (MC) or Recurrent Neural Networks (RNN) to model human mobility based on their historical check-ins. MC based methods [Zhang *et al.*, 2016], e.g., Ranking based Markov chain (FPMC) and ranking based MC transition, are implemented under a strong independence assumption among non-adjacent locations, which limits their performance on capturing long term dependency of locations. RNN, on the other hand, is a special neural network that is able to handle variable-length input and output. It predicts the next output in a sequence, given all the previous outputs, by modeling the joint probability distribution over sequences. It has been successfully used in many spatio-temporal location predictions [Liu *et al.*, 2016] and recently in identifying human mobility patterns in TUL context [Gao *et al.*, 2017]. However, applying RNN directly in modeling check-in sequences in GTSM confronts the following challenges: (1) *data sparsity*: the density of check-ins such as those from Foursquare and Yelp are usually around 0.1% [Li *et al.*, 2017a]; (2) *structural dependency*: strong and complex dependencies among check-ins or trajectories exist at different time-steps [Chung *et al.*, 2015]; and (3) *shallow generation*: model variability (or stochasticity) occurs only when an output (e.g., location in POI sequence) is sampled [Serban *et al.*, 2017].

In this paper, we tackle the above challenges in modeling human trajectories and linking trajectories to their generating-users with a novel method – *TULVAE (TUL via Variational AutoEncoder)*. The main benefits of TULVAE are:
(1) It alleviates the data sparsity problem by adapting semi-supervised variational autoencoder [Kingma *et al.*, 2014] to utilize a large volume of unlabeled data to improve the performance of the TUL task.
(2) It instantiates an architecture for learning the distributions of latent random variables to model the variability observed in trajectories. By incorporating variational inference into the generative model with latent variables, TULVAE exposes an interpretable representation of the complex distribution and long-term dependencies over POI sequences.
(3) By exploiting the practical fact that users' mobility tra-

[†]Corresponding author.

jectories exhibit high spatio-temporal regularity (e.g., more than 90% of nighttime mobility records are generated in the same POI [Xu *et al.*, 2017a]), TULVAE is able to capture the semantics of subtrajectories inherently representing the uniqueness of individual's motion. Furthermore, users' mobility trajectories exhibit hierarchical properties – e.g., frequent POIs within a subtrajectory, and some implicit patterns (e.g., the meaning of destinations) encoded across trajectories. This motivates us to exploit hierarchical mobility patterns for improving human mobility identification, as well as performance. As our main contributions:

- We take a first step towards addressing the data sparsity problem in GTSM with semi-supervised learning, especially via incorporating unlabeled data.
- We propose an optimization-based approach for modeling and inferring latent variables in human mobility, which, to our best knowledge, is the first variational trajectory inference model and opens up a new perspective for spatial data mining.
- We tackle the problem of hierarchical structures and complex dependencies of mobility trajectories by modeling both within- and across-trajectory semantics.
- We provide experimental evaluations illustrating the improvements enabled by TULVAE, using three publicly available GTSM datasets and comparing with several existing models.

In the rest of the paper, we review related work in Section 2 and introduce preliminaries in Section 3. Section 4 introduces the technical detail of TULVAE, followed by experimental observations presented in Section 5 and concluding remarks in Section 6.

2 Related Work

Mining human mobility behavior is a trending research topic in AI [Zhuang *et al.*, 2017], GIS [Zheng *et al.*, 2008] and recommendation systems [Yang *et al.*, 2017a] and trajectory classification (i.e., categorizing trajectories into different motion patterns) is one of the central tasks in understanding mobility patterns [Zheng *et al.*, 2008]. TUL was recently introduced [Gao *et al.*, 2017] for correlating (unlabeled) trajectories to their generating-users in GTSM, using RNN based models to learn the mobility patterns and classify trajectories by users. However, the standard RNN based supervised trajectory models suffer from lacking of understanding hierarchical semantics of human mobility and fail to leverage the unlabeled data which embeds rich and unique individual mobility patterns. Trajectory recovery problem was studied in a similar manner in [Xu *et al.*, 2017a], inferring individual's identity using trajectory statistics – essentially an equivalent TUL problem. The performance is greatly limited by the extreme sparsity issue in GTSM data: as observed in [Li *et al.*, 2017a] the density of check-ins in Foursquare and Yelp data is around 0.1%, and that of the Gowalla data is around 0.04% [Yang *et al.*, 2017a].

Deep generative models, such as Generative Adversarial Networks (GAN) [Goodfellow *et al.*, 2014], autoregressive models [van den Oord *et al.*, 2016] and Variational AutoEncoder (VAE) [Kingma and Welling, 2014], have been successful in image generation and natural language modeling [Yang *et al.*, 2017b]. Semi-supervised generative learning, utilizing both labeled and unlabeled data for modeling complex high dimensional latent variables, has recently attracted increasing attention [Kingma *et al.*, 2014; Hu *et al.*, 2017]. VAE has shown promising performance on text classification [Xu *et al.*, 2017b] and language generation tasks [Serban *et al.*, 2017], however, its application in modeling human mobility has not been well investigated and previous VAE based models cannot be applied to TUL due to the data sparsity and complex semantic structures underlying humans' check-in sequences. TULVAE proposed in this paper differs from earlier works in: (1) tackling the latent variable inference problem in human mobility trajectories; (2) learning hierarchical semantics of human check-in sequences; and (3) incorporating the unlabeled data for identifying individual mobility pattern and for solving the TUL problem in the manner of semi-supervised learning.

Recent research in Moving Objects Databases (MOD) community has tackled problems related to managing spatio-textual trajectories (cf. [Issa and Damiani, 2016]) – however, most of the works pertain to efficient query processing and are complementary to the problems investigated in this paper.

3 Preliminaries

We now introduce the TUL problem and basics of VAE.

Trajectory-User Linking: Let $\mathbf{t}_{u_i} = \{c_{i1}, c_{i2}, ..., c_{in}\}$ denote a trajectory generated by the user u_i during a time interval, where c_{ij} ($j \in [1, n]$) is a location at time t_j for the user u_i, in a suitable coordinate system (e.g., longitude + latitude, or some (x_{ij}, y_{ij}). We refer to c_{ij} as a *check-in* in this paper. A trajectory $\overline{\mathbf{t}}_i = \{c_1, c_2, ..., c_m\}$ for which we do not know the user who generated it, is called *unlinked*. The TUL problem is accordingly defined as: suppose we have a number of unlinked trajectories $\mathcal{T} = \{\overline{\mathbf{t}}_1, ..., \overline{\mathbf{t}}_m\}$ produced by a set of users $\mathcal{U} = \{u_1, ..., u_n\}$ ($m \gg n$). TUL learns a mapping function that links unlinked trajectories to users: $\mathcal{T} \mapsto \mathcal{U}$ [Gao *et al.*, 2017].

Variational Autoencoders: Similar in spirit to text modeling (cf. [Yang *et al.*, 2017b]), we consider a dataset consisting of pairs $(\mathbf{t}_{u_1}, u_1)$, $\cdots$, $(\mathbf{t}_{u_m}, u_m)$, with the i-th trajectory $\mathbf{t}_{u_i} \in \mathcal{T}$ and the corresponding user (label) $u_i \in \mathcal{U}$. We assume that an observed trajectory is generated by a latent variable z_i. Following [Kingma and Welling, 2014], we omit the index i whenever it is clear that we are referring to terms associated with a single data point – i.e., a trajectory. The empirical distribution over the labeled and unlabeled subsets are respectively denoted by $\tilde{p}_l(\mathbf{t}, u)$ and $\tilde{p}_u(\mathbf{t})$.

We aim at maximizing the probability of each trajectory $\mathbf{t}$ in the training set under the generative model, according to $p_\theta(\mathbf{t}) = \int_z p_\theta(\mathbf{t}|z)p_\theta(z)\mathrm{d}z$, where: $p_\theta(\mathbf{t}|z)$ refers to a *generative model* or *decoder*; $p_\theta(z)$ is the prior distribution of the random latent variable z, e.g., an isotropic multivariate Gaussian; $p_\theta(z) = \mathcal{N}(0, \mathbf{I})$ ($\mathbf{I}$ is the identity matrix); and θ is the generative parameters of the model. Typically, to estimate the generative parameters θ, the evidence lower bound (ELOB)

$\mathcal{E}^d(\mathbf{t})$ (a.k.a. the negative free energy) on the marginal likelihood of a single trajectory is used as an objective [Kingma and Welling, 2014]:

$$\log p_\theta(\mathbf{t}) = \log p_\theta(\mathbf{t}) \int_z q_\phi(z|\mathbf{t})\mathrm{d}z \geq -\mathcal{E}^d(\mathbf{t})$$
$$= \mathbb{E}_{z\sim q_\phi(z|\mathbf{t})}[\log p_\theta(\mathbf{t}|z)] - \mathbb{KL}\left[q_\phi(z|\mathbf{t})||p_\theta(z)\right] \quad (1)$$

where $q_\phi(z|\mathbf{t})$ is an approximation to the true posterior $p_\theta(z|\mathbf{t})$ (a.k.a *recognition model* or *encoder*) parameterized by ϕ. $\mathbb{KL}\left[q_\phi(\cdot)||p_\theta(\cdot)\right]$ is the Kullback-Leibler (KL) divergence between the learned latent posterior distribution $q(z|\mathbf{t})$ and the prior $p(z)$ (for brevity, we will omit the parameters ϕ and θ in subsequent formulas). Since the objective (of the model) is to minimize the KL divergence between $q(z|\mathbf{t})$ and the true distribution $p(z|\mathbf{t})$ – we can alternatively maximize ELOB $\mathcal{E}^d(\mathbf{t})$ of $\log p(\mathbf{t})$ w.r.t. both θ and ϕ, which are jointly trained with separate neural networks such as multilayer perceptrons. We refer to [Kingma and Welling, 2014; Kingma *et al.*, 2014] regarding the details of VAE and reparameterization approaches used for stochastic gradient descent training.

4 TUL via Variational AutoEncoder (TULVAE)

We now describe the details of the proposed TULVAE model consists of four RNN-based components: *encoder RNN*, *intermediate RNN*, *decoder RNN* and *classifier*, as illustrated in Figure 1.

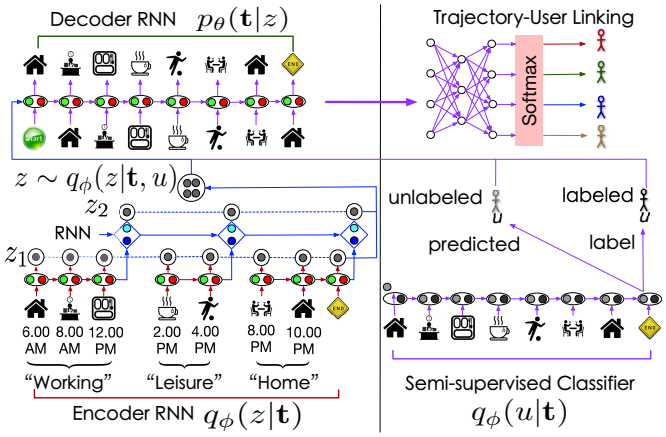


Figure 1: Overview of the approach: TULVAE first uses trajectories to learn all check-in embeddings (low-dimension representation) $\mathbf{T} \in \mathbb{R}^{|C|\times d}$. *Bottom Left*: Encoder and intermediate RNNs are employed to learn the hierarchical structures of check-in sequences, with two-layer of latent variables concatenated to represent the latent space. *Top Left*: A sample z from the posterior $q_\phi(z|\mathbf{t}, u)$ and user u are passed to the generative network to estimate the probability $p_\theta(\mathbf{t}|u, z)$. *Bottom Right*: The unlabeled data is used to train a classifier $q_\phi(z|\mathbf{t})$ to characterize label-distribution. *Top Right*: A user for a given unlinked trajectory is predicted by the deep neural networks.

4.1 Semantic Trajectory Segmentation

The trajectories of a user u_i are originally separated by days, i.e., the original trajectory data $\mathbf{T}_{u_i}$ is segmented into k consecutive sub-sequences $\mathbf{t}^1_{u_i}, ..., \mathbf{t}^k_{u_i}$, where k is the number of days that u_i has check-ins. We consider two semantic factors: (1) *Temporal influence*: Following [Gao *et al.*, 2017], each daily trajectory $\mathbf{t}^j_{u_i}$ ($j \in [1, k]$) is split into 4 consecutive subsequences $\mathbf{t}^{j,1}_{u_i}, ..., \mathbf{t}^{j,4}_{u_i}$ based on time intervals of 6 hours. (2) *Spatial influence*: In Gowalla and Foursquare datasets, 90% of users' transition distances are less than 50km [Xu *et al.*, 2017a] – indicating that users tend to visit nearby POIs. Thus, we further split a subtrajectory $\mathbf{t}^{j,m}_{u_i}$ ($m \in [1, 4]$) if the continuous POI distance is more than 50 km.

4.2 Hierarchical Trajectory Encoding

Inspired by the hierarchical text models - e.g., in document-level classification [Zhang *et al.*, 2018], and utterance-level dialogue generation [Serban *et al.*, 2017], we model daily trajectories in a two-level structural hierarchy: a trajectory consists of subtrajectories that encode spatio-temporal moving patterns of an individual, while a subtrajectory is composed of sequential POIs:

$$p_\theta(\mathbf{s}_1, \cdots, \mathbf{s}_N) = \prod_{n=1}^{N}\prod_{m=1}^{M_n} p_\theta(s_{n,m}|s_{n,1:m-1}, \mathbf{s}_{1:n-1}) \quad (2)$$

where $\mathbf{s}_n$ is the n^{th} subtrajectory in a daily check-in sequence, $s_{n,m}$ is the m^{th} POI in the n^{th} subtrajectory, and M_n is the number of POIs in the m^{th} subtrajectory.

At the POI level, a POI Long Short-Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997] (or a Gated Recurrent Unit (GRU) [Chung *et al.*, 2014]) is used to encode the POIs to form an implicit topic of a subtrajectory (e.g., "working", "Leisure" or "Home" in Figure 1). The last hidden states of the POI level encoder are fed into the intermediate LSTM (or GRU), where the internal hidden states are encoded into a vector to reflect the structured characteristics of daily mobility. The internal state of the hierarchical trajectory encoding is mathematically described as:

$$\begin{cases} \mathbf{h}^{\text{POI}}_{n,0} = 0, \mathbf{h}^{\text{POI}}_{n,m} = \text{LSTM}(\mathbf{h}^{\text{POI}}_{n,m-1}, s_{n,m}), & \text{POI RNN} \\ \mathbf{h}^{\text{INT}}_{0} = 0, \mathbf{h}^{\text{INT}}_{n} = \text{LSTM}(\mathbf{h}^{\text{INT}}_{n-1}, \mathbf{h}^{\text{INT}}_{n,M_n}), & \text{Intermediate RNN} \end{cases}$$

where respective LSTM$(\cdot)$ is a "vanilla" LSTM function:

$$\begin{aligned} \mathbf{i_t} &= \sigma(\mathbf{W_i v_t} + \mathbf{U_i h_{t-1}} + \mathbf{b_i}) \\ \mathbf{f_t} &= \sigma(\mathbf{W_f v_t} + \mathbf{U_f h_{t-1}} + \mathbf{b_f}) \\ \mathbf{o_t} &= \sigma(\mathbf{W_o v_t} + \mathbf{U_o h_{t-1}} + \mathbf{b_o}) \\ \tilde{\mathbf{c}}_t &= \tanh(\mathbf{W_c v_t} + \mathbf{U_c h_{t-1}} + \mathbf{b_c}) \\ \mathbf{c_t} &= \mathbf{f_t} \odot \mathbf{c_{t-1}} + \mathbf{i_t} \odot \tilde{\mathbf{c}}_\mathbf{t} \\ \mathbf{E_{v_t}} &= \mathbf{h_t} = \mathbf{o_t} \odot \tanh(\mathbf{c_t}) \end{aligned} \quad (3)$$

where $\mathbf{i_t}$, $\mathbf{f_t}$, $\mathbf{o_t}$ and $\mathbf{b}_*$ are respectively the input gate, forget gate, output gate and bias vectors; σ is the logistic sigmoid function; matrices $\mathbf{W}$ and $\mathbf{U}$ ($\in \mathbb{R}^{d\times d}$) are the different gate parameters; and $\mathbf{v_t}$ is the embedding vector of the POI c_t.

The memory cell $\mathbf{c_t}$ is updated by replacing the existing memory unit $\tilde{\mathbf{c}}_t$ with a new cell, where $\tanh(\cdot)$ is the hyperbolic tangent function, and $\odot$ is the component-wise multiplication. The output encoding vector $\mathbf{E_{v_t}}$ is the final hidden

layer. We refer the proposed hierarchical TUL learning as HTULERs, which alone improves the performance, in comparison with shallow POI-level learning methods (cf. Section 5).

4.3 TULVAE

To alleviate data sparsity problem, incorporating unlabeled data can improve the performance of identity linking – which may be partially addressed by the semi-supervised VAE [Kingma *et al.*, 2014]. However, this has limitations in modeling hierarchical structures of latent variables due to estimating the posterior with a single layer of encoder and may fail to capture abstract representations. Similarly, recursively stacking VAEs on top of each other [Sønderby *et al.*, 2016] is restricted by the overfitting of bottom generative layer – which alone learns sufficient representation to reconstruct the original data and renders the above layer(s) unnecessary [Zhao *et al.*, 2017].

Thus, we prefer to encode abstract topics of subtrajectories on certain part of the latent variables, and POI sequence patterns in others – learning the inference model in a "flat" semi-VAE framework. The basic idea of this implicit hierarchical generative model is inspired by the variational ladder autoencoder (VLAE) [Zhao *et al.*, 2017], where shallow networks are used to express low-level simple features. Deep networks, in turn, express high-level complex features, and inject Gaussian noise at different levels of the network. We note, though, that VLAE is designed for continuous latent feature discriminative learning, while TULVAE is a semi-supervised learning framework for discrete trajectory data.

More specifically, we consider a two-layer of latent variables z_1, z_2, respectively denoting the POI-level (bottom) and abstract level (top) layer. The prior $p(z) = p(z_1, z_2)$ is a standard Gaussian and the joint distribution $p(\mathbf{t}, z_1, z_2)$ can be thus factored as $p(\mathbf{t}, z_1, z_2) = p(\mathbf{t}|z_1, z_2)p(z_1|z_2)p(z_2)$. In the generative network, the latent variable z is a concatenation of two vectors $z = [\text{RNN}_m(z_2); \text{RNN}_n(z_1)]$, where z_2 and z_1 are respectively parameterized by the intermediate RNN and POI RNN. For the inference network, the approximate posterior $p(z|\mathbf{t})$ is a Gaussian $\mathcal{N}(\mu_i, \sigma_i)(i = 1, 2)$ parameterized by two-level RNNs.

Given the implicit representation of the hierarchy of latent variables, we now learn the trajectories in a semi-supervised manner and essentially adapt in a single-layer model. Thus, when the user u corresponding to a trajectory is observed (*labeled* data D_l), we have:

$$\begin{aligned}\log p(\mathbf{t}, u) &= \mathbb{E}_{z\sim q}[\log p(\mathbf{t}|u, z)] + \log p(u) \\ &\quad - \mathbb{KL}\left[q(z|\mathbf{t}, u)||p(z)\right] + \mathbb{KL}\left[q(z|\mathbf{t}, u)||p(z|\mathbf{t}, u)\right] \quad (4)\end{aligned}$$

Our goal is to approximate the true posterior $p(z|\mathbf{t}, u)$ with $q(z|\mathbf{t}, u)$, i.e., minimizing $\mathbb{KL}\left[q(z|\mathbf{t}, u)||p(z|\mathbf{t}, u)\right]$. Therefore, we obtain the following ELOB $\mathcal{E}_1^g(\mathbf{t}, u)$:

$$\mathcal{E}_1^g(\mathbf{t}, u) = \mathbb{E}_{z\sim q}[\log p(\mathbf{t}, u, z)] - \mathbb{KL}\left[q(z|\mathbf{t}, u)||p(z)\right] \quad (5)$$

where $\mathbb{KL}\left[q(z|\mathbf{t}, u)||p(z)\right]$ is the KL divergence between the latent posterior $q(z|\mathbf{t}, u)$ and the prior distribution $p(z)$, which measures how much information is lost when approximating a prior over z. The expectation term is the reconstruction error or expected negative log-likelihood of the data, which encourages the decoder to reconstruct the trajectory from the latent distribution.

In the case of *unlabeled* trajectory data D_u, the user identity u is predicted by performing posterior inference with a *classifier* $q_\phi(u|\mathbf{t})$. We now have following ELOB $\mathcal{E}_2^g(\mathbf{t})$, by considering u as another latent variable:

$$\begin{aligned}\log p(\mathbf{t}) &\geq \mathbb{E}_{z\sim q(u,z|\mathbf{t})}[\log p(\mathbf{t}, u, z) - \log q(u, z|\mathbf{t})] \\ &= \sum_u q(u|\mathbf{t})(\mathcal{E}_1^g(\mathbf{t}, u)) + \mathcal{H}(q(u|\mathbf{t})) = \mathcal{E}_2^g(\mathbf{t}) \quad (6)\end{aligned}$$

where $\mathcal{H}(q(u|\mathbf{t}))$ is the information entropy, and the loss of the classifier $q_\phi(u|\mathbf{t})$ during training is measured by L_2 reconstruction error between the predicted user and the real label. Thus, the ELOB $\mathcal{E}$ on the marginal likelihood for the entire dataset is:

$$\mathcal{E} = -\sum_{(\mathbf{t},u)\sim D_l} (\mathcal{E}_1^g(\mathbf{t}, u) + \alpha \log q(u|\mathbf{t})) - \sum_{\mathbf{t}\sim D_u} \mathcal{E}_2^g(\mathbf{t}) \quad (7)$$

where the first RHS term includes an additional classification loss of classifier $q_\phi(u|\mathbf{t})$ when learning from the labeled data, and hyper-parameter α controls the weight of labeled data learning.

We note that semi-VAE scales linearly in the number of classes in the data sets, which is raised by re-evaluating the generative likelihood for each class during training. For the task of text classification, the number of classes is small (e.g., 2 classes in IMDB and 4 classes in AG's News). However, the number of classes (i.e., users) in TUL is very large, which incurs heavy computation on evaluating classifier $q_\phi(u|\mathbf{t})$. To overcome this problem, we employ the Monte Carlo method suggested in [Xu *et al.*, 2017b] to estimate the expectation evaluations per class, where the baseline method from [Williams, 1992] is used to reduce the variance of the sampling-based gradient estimator: $\bigtriangledown_\phi \mathbb{E}_{z\sim q(z|\mathbf{t},u)}[q(u|\mathbf{t})(-\mathcal{E}_1^g(\mathbf{t}, u)]$. Since ELOB $\mathcal{E}_2^g(\mathbf{t})$ determines the magnitude of classifier $q_\phi(u|\mathbf{t})$, the baseline we use in TULVAE is the averaged $\mathcal{E}_2^g(\mathbf{t})$, following the choice in previous text classification works, i.e., [Xu *et al.*, 2017b].

4.4 Training

At the early stage of TULVAE training, the term $\mathbb{KL}\left[q(z|\mathbf{t}, u)||p(z)\right]$ in Eq.4 may discourage encoding interesting information into the latent variable z, thereby easily resulting in model collapse – largely because of the strong learning capability of autoregressive models such as RNN as observed in [Bowman *et al.*, 2016]. One efficient solution is to control the weight of KL-divergence term by gradually increasing its co-efficiency β (from 0 to 1), which is also called KL cost annealing [Bowman *et al.*, 2016].

The activation function used in the RNN models is softplus (i.e., $f(x) = \ln(1 + e^x)$), which applies nonlinearity to each component of its argument vector ensuring positive variances. In addition, we use the "bucket trick" to speed up the training process: we sort all trajectories by length, and put those with similar length into the same bucket, in which the data is padded to the same length and fed into the neural networks as a batch. This may yield efficiency – i.e., reduce the computation time since variable length data is one of the bottlenecks in RNN based training.

Finally, we use Bidirectional LSTMs in TULVAE which could jointly model the forward likelihood and the backward posterior for training the POI sequences, which could potentially capture richer representation of the mobility patterns and speed up the convergence of training.

Discussion: TULVAE formulates the solution to TUL problem in a semi-VAE framework combined with hierarchical trajectory modeling with RNNs. There exist several other choices in TULVAE – e.g., the decoder could be replaced with a CNN which has the potential to improve the efficiency since it only requires a convolution on one dimension for discrete data, such as dilation CNN decoder used in [Yang *et al.*, 2017b] for text modeling. Another modification is to incorporate the label information to the latent representation, using the disentangled variables instead of introducing a classifier $q_\phi(u|\mathbf{t})$ [Li *et al.*, 2017b]. By dividing the latent representation into disentangled variables and non-interpretable variables, the categorical information can be explicitly employed to regularize the disentangled representation. However, in this work we focus on inferring the distribution of variables in human check-ins and improving the performance of TUL. The investigation of alternatives for TULVAE are left for our future work.

5 Evaluation

In this section we present the evaluation of the benefits of TULVAE using three real-word GTSM datasets. To ease the reproduction of our results, we have made the source code of TULVAE publicly available[1].

Dataset	$\lvert\mathcal{U}\rvert$	$\lvert\mathcal{T}_n\rvert/\lvert\mathcal{T}_e\rvert$	$\lvert C\rvert$	$\overline{\mathcal{R}}$	$\mathcal{T}_r$
Gowalla	201	9,920/10,048	10,958	219	[1,131]
	112	4,928/4,992	6,683	191	[1,95]
Brightkite	92	9,920/9,984	2,123	471	[1,184]
	34	4,928/4,992	1,359	652	[1,44]
Foursquare	270	12,800/12,928	7,195	242	[1,35]
	109	5,312/5,376	4,227	246	[1,35]

Table 1: *Data description*: $|\mathcal{U}|$: the number of users; $|\mathcal{T}_n|/|\mathcal{T}_e|$: number of trajectories for training and testing; $|C|$: number of check-ins; $\overline{\mathcal{R}}$: average length of trajectories (before segmentation); $\mathcal{T}_r$: range of the trajectory length

Datasets: We conducted our experiments on three publicly available GTSM datasets: Gowalla[2], Brightkite[3] and Foursquare[4]. For Foursquare, we choose the most popular city – New York. We randomly select $|\mathcal{U}|$ users and their corresponding trajectories from the datasets for evaluation – for each dataset, we select two different numbers of those users (e.g., labels here) who generate varied trajectories for robustness check of model performance. Table 1 depicts the statistics of the three datasets.

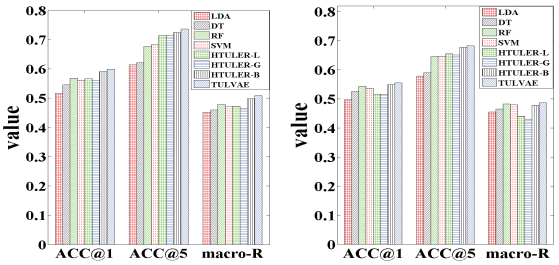
Baselines and Metrics: We compare TULVAE with several state-of-the-art approaches from the field of trajectory similarity measurement and deep learning based classification. We also implemented three hierarchical variations of

[1]https://github.com/AI-World/IJCAI-TULVAE

[2]http://snap.stanford.edu/data/loc-gowalla.html

[3]http://snap.stanford.edu/data/loc-brightkite.html

[4]https://sites.google.com/site/yangdingqi/home



(a) $|\mathcal{U}|$=109 (b) $|\mathcal{U}|$=270

Figure 2: Comparing to traditional methods on Foursquare.

TULER (TUL via Embedding and RNN) [Gao *et al.*, 2017] – namely: HTULER-L, HTULER-G and HTULER-B, respectively implemented with the hierarchical LSTM, GRU and Bi-directional LSTM but without variational inference. In our implementation, multivariate Gaussian distribution is used as the prior in TULVAE. The learning rate of all models is initialized with 0.001 and decays with rate of 0.9. The weight β (KL cost annealing) increases from 0.5 to 1; and the dropout rate is 0.5. We embed POIs in 250 dimensional vectors and used 300 units for classifier, 512 units for the encoder-decoder RNN and 100 units for latent variable z. Finally, the batch size is 64 for all RNN based models.

The baselines used for benchmarking can be broadly categorized as:

(a) **Traditional approaches**, including ***LDA*** (Linear Discriminant Analysis, with SVD as matrix solver), ***DT*** (Decision Tree), ***RF*** (Random Forest) and ***SVM*** (Support Vector Machine, with linear kernel), which are widely used for measuring mobility patterns and classifying trajectories in literatures [Zheng, 2015].

(b) **RNN based TUL**, including ***TULER-LSTM***, ***TULER-GRU***, ***TULER-LSTM-S***, ***TULER-GRU-S*** and ***Bi-TULER*** proposed in [Gao *et al.*, 2017], which are the state-of-the-art methods for trajectory-user linking.

We report the ACC@K, macro-P, macro-R and macro-F1 of all methods, which are common metrics in information retrieval area. Specifically, ACC@K is used to evaluate the trajectory-user linking accuracy as:

$$\text{ACC@}K = \frac{\textit{\# correctly identified trajectories @}K}{\textit{\# trajectories}}$$

and macro-F1 is the harmonic mean of the precision (*macro-P*) and recall (macro-R), averaged across all classes (users in TUL):

$$\text{macro-F1} = 2 \times \frac{\textit{macro-P} \times \textit{macro-R}}{\textit{macro-P} + \textit{macro-R}}$$

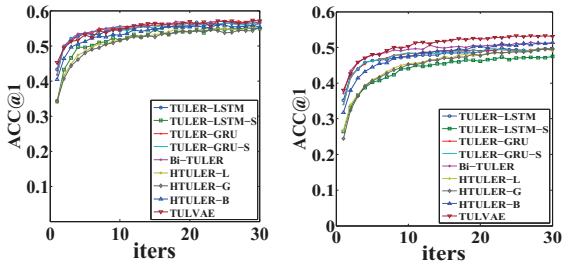
Performance comparison

Table 2 summarizes the performance comparisons among the proposed method and RNN based baselines on three datasets, where the best method is shown in **bold**, and the second best is shown as underlined. We demonstrate the comparison to traditional approaches in Figure 2 only for Foursquare dataset – because, to our knowledge, these methods have already been shown to be inferior to TULERs [Gao *et al.*, 2017] on Gowalla and Brightkite. From the results, we can see how the

Dataset	Method \ Metric	ACC@1	ACC@5	macro-P	macro-R	macro-F1	ACC@1	ACC@5	macro-P	macro-R	macro-F1
		$\|\mathcal{U}\|$=112					$\|\mathcal{U}\|$=201				
Gowalla	TULER-LSTM	41.79%	57.89%	33.61%	31.33%	32.43%	41.24%	56.88%	31.70%	28.60%	30.07%
	TULER-GRU	42.61%	57.95%	35.22%	32.69%	33.91%	40.85%	57.31%	29.52%	27.80%	28.64%
	TULER-LSTM-S	42.11%	58.01%	33.49%	31.97%	32.71%	41.22%	57.70%	29.34%	28.68%	29.01%
	TULER-GRU-S	41.35%	58.45%	32.51%	31.79%	32.15%	41.07%	57.49%	29.08%	27.17%	28.09%
	Bi-TULER	42.67%	59.54 %	37.55%	33.04%	35.15%	41.95%	57.58%	32.15%	31.66%	31.90%
	HTULER-L	43.89%	60.90%	35.95%	34.32%	35.12%	43.40%	60.25%	34.43%	33.63%	34.02%
	HTULER-G	43.33%	60.74%	37.71%	**34.47%**	36.01%	42.88%	59.41%	32.72%	32.54%	32.63%
	HTULER-B	44.21%	62.28%	36.48%	33.51%	34.93%	44.50%	60.93%	34.89%	34.46%	34.67%
	TULVAE	**44.35%**	**64.46%**	**40.28%**	32.89%	**36.21%**	**45.40%**	**62.39%**	**36.13%**	**34.71%**	**35.41%**
		$\|\mathcal{U}\|$=34					$\|\mathcal{U}\|$=92				
Brightkite	TULER-LSTM	48.26%	67.39%	49.90%	47.20%	48.51%	43.01%	59.84%	38.45%	35.81%	37.08%
	TULER-GRU	47.84%	67.42%	48.88%	46.87%	47.85%	44.03%	61.36%	38.86%	36.47%	37.62%
	TULER-LSTM-S	47.88%	67.38%	48.81%	47.03%	47.62%	44.23%	61.00%	38.02%	36.33%	37.16%
	TULER-GRU-S	48.08%	68.23%	48.87%	46.74%	47.78%	43.93%	61.85%	37.93%	36.01%	36.94%
	Bi-TULER	48.13%	68.17%	49.15%	47.06%	48.08%	43.54%	60.68%	38.20%	36.47%	37.31%
	HTULER-L	49.44%	71.13%	51.51%	47.31%	**49.32%**	45.26%	63.55%	41.61%	38.13%	39.79%
	HTULER-G	49.12%	70.81%	51.85%	46.88%	49.24%	44.50%	63.17%	41.10%	37.51%	39.22%
	HTULER-B	49.78%	70.69%	**52.45%**	**47.98%**	48.90%	45.30%	63.93%	41.82%	39.32%	38.60%
	TULVAE	**49.82%**	**71.71%**	51.26%	46.43%	48.72%	**45.98%**	**64.84%**	**43.15%**	**39.65%**	**41.32%**
		$\|\mathcal{U}\|$=109					$\|\mathcal{U}\|$=270				
Foursquare	TULER-LSTM	57.24%	69.27%	49.35%	47.61%	48.46%	50.69%	62.11%	46.27%	41.84%	43.95%
	TULER-GRU	56.85%	69.40%	49.05%	47.34%	48.18%	50.65%	62.68%	46.38%	41.65%	43.89%
	TULER-LSTM-S	57.14%	69.57%	48.48%	47.59%	48.03%	49.55%	62.65%	43.40%	42.11%	42.75%
	TULER-GRU-S	56.31%	69.56%	49.04%	46.98%	47.99%	50.21%	62.33%	46.17%	41.01%	43.44%
	Bi-TULER	58.31%	71.17%	50.84%	48.88%	49.84%	52.31%	64.03%	47.15%	44.95%	46.03%
	HTULER-L	56.66%	71.46%	48.33%	47.28%	47.80%	51.59%	65.53%	45.82%	44.06%	44.92%
	HTULER-G	55.92%	71.37%	48.10%	46.47%	47.27%	51.46%	65.15%	45.34%	43.03%	43.97%
	HTULER-B	59.10%	72.40%	51.37%	49.85%	50.03%	54.91%	67.76%	48.94%	47.82%	48.37%
	TULVAE	**59.91%**	**73.60%**	**53.59%**	**50.93%**	**52.23%**	**55.54%**	**68.27%**	**51.07%**	**48.63%**	**49.83%**

Table 2: Comparison among different TUL methods on three datasets.

hierarchical trajectory modeling combined with latent representation in exploring human mobility patterns yields performance improvements over the baseline(s). In summary:
(1) TULVAE performs the best on most of metrics. This superior result is due to its capability of learning the complicated latent distribution of trajectories and leveraging the unlabeled data. By modeling the distribution of trajectories in a probabilistic generative model (rather than point estimation in "vanilla" RNNs), TULVAE is able to capture underlying semantics of mobility patterns. In addition, by incorporating unlabeled data into the training, the semi-supervised classifier in TULVAE may ameliorate the data sparsity problem inherent to the GSTM data. However, the latent representation learned is often not effective enough, especially when the data size is small, e.g., $|\mathcal{U}| = 34$ in Brightkite. We conjecture that this is partially because of the *entangled* representation produced by the encoder, which results in difficulty on characterizing variations of relative small and sparse datasets.
(2) We note the improvements due to hierarchical trajectory modeling when focusing more specifically on comparison between HTULERs and TULERs. Although TULERs use variant RNNs, they suffer from the shallow generation in modeling check-in sequences. In contrast, HTULERs explore structural information of human mobility, which leads to a more robust performance – even superior on several metrics when the number of users is relative small.
(3) When it comes to the training process of various deep learning-based TUL methods, Figure 3 shows the results (ACC@1) on Foursquare and it shows that TULVAE exhibits fastest convergence. This demonstrates the effectiveness of inherent generative models in understanding human mobility. Similar results also hold for other metrics and other datasets but are omitted here due to the lack of space.



(a) $|\mathcal{U}|$=109 (b) $|\mathcal{U}|$=270

Figure 3: Training ACC@1 of various TUL methods on Foursquare.

6 Conclusions

We presented TULVAE, a generative model to mine human mobility patterns, which aims at learning the implicit hierarchical structures of trajectories and alleviating the data sparsity problem with the semi-supervised learning. TULVAE achieves a significant performance improvement for the TUL problem in comparison to existing methods. In addition, TULVAE can be augmented by incorporating other representative features such as spatial and temporal information in the latent space, which we leave for our future investigation.

Acknowledgements

This work was supported by National Natural Science Foundation of China (Grant No.61602097, No.61472064 and No.61502087), NSF grants III 1213038 and CNS 1646107, ONR grant N00014-14-10215 and HERE grant 30046005, and the Fundamental Research Funds for the Central Universities (No.ZYGX2015J072).

References

[Bowman *et al.*, 2016] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Józefowicz, and Samy Bengio. Generating Sentences from a Continuous Space. In *CoNLL*, 2016.

[Chung *et al.*, 2014] Junyoung Chung, Caglar Gulcehre, Kyung Hyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *Eprint Arxiv*, 2014.

[Chung *et al.*, 2015] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A Recurrent Latent Variable Model for Sequential Data. In *NIPS*, 2015.

[Gao *et al.*, 2017] Qiang Gao, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Xucheng Luo, and Fengli Zhang. Identifying Human Mobility via Trajectory Embeddings. *IJCAI*, 2017.

[Goodfellow *et al.*, 2014] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C Courville, and Yoshua Bengio. Generative Adversarial Nets. In *NIPS*, 2014.

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

[Hu *et al.*, 2017] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Toward Controlled Generation of Text. In *ICML*, 2017.

[Issa and Damiani, 2016] Hamza Issa and Maria Luisa Damiani. Efficient access to temporally overlaying spatial and textual trajectories. In *IEEE MDM*, 2016.

[Kingma and Welling, 2014] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *ICLR*, 2014.

[Kingma *et al.*, 2014] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised Learning with Deep Generative Models. In *NIPS*, 2014.

[Li *et al.*, 2017a] Huayu Li, Yong Ge, Defu Lian, and Hao Liu. Learning User's Intrinsic and Extrinsic Interests for Point-of-Interest Recommendation: A Unified Approach. In *IJCAI*, 2017.

[Li *et al.*, 2017b] Yang Li, Quan Pan, Suhang Wang, Haiyun Peng, Tao Yang, and Erik Cambria. Disentangled Variational Auto-Encoder for Semi-supervised Learning. *arxiv*, 2017.

[Liu *et al.*, 2016] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. Predicting the next location: a recurrent model with spatial and temporal contexts. In *AAAI*, 2016.

[Serban *et al.*, 2017] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. In *AAAI*, 2017.

[Sønderby *et al.*, 2016] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder Variational Autoencoders. In *NIPS*, 2016.

[van den Oord *et al.*, 2016] Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel Recurrent Neural Networks. In *ICML*, 2016.

[Williams, 1992] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.

[Xu *et al.*, 2017a] Fengli Xu, Zhen Tu, Yong Li, Pengyu Zhang, Xiaoming Fu, and Depeng Jin. Trajectory Recovery From Ash - User Privacy Is NOT Preserved in Aggregated Mobility Data. *WWW*, 2017.

[Xu *et al.*, 2017b] Weidi Xu, Haoze Sun, Chao Deng, and Ying Tan. Variational Autoencoder for Semi-Supervised Text Classification. In *AAAI*, 2017.

[Yang *et al.*, 2017a] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. Bridging Collaborative Filtering and Semi-Supervised Learning: A Neural Approach for POI Recommendation . In *SIGKDD*, 2017.

[Yang *et al.*, 2017b] Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. Improved Variational Autoencoders for Text Modeling using Dilated Convolutions. In *ICML*, 2017.

[Zhang *et al.*, 2016] Chao Zhang, Keyang Zhang, Quan Yuan, Luming Zhang, Tim Hanratty, and Jiawei Han. Gmove: Group-level mobility modeling using geo-tagged social media. In *ACM SIGKDD*, 2016.

[Zhang *et al.*, 2018] Tianyang Zhang, Minlie Huang, and Li Zhao. Learning Structured Representation for Text Classification via Reinforcement Learning. In *AAAI*, 2018.

[Zhao *et al.*, 2017] Shengjia Zhao, Jiaming Song, and Stefano Ermon. Learning Hierarchical Features from Deep Generative Models. *ICML*, 2017.

[Zheng *et al.*, 2008] Yu Zheng, Quannan Li, Yukun Chen, Xing Xie, and Wei Ying Ma. Understanding mobility based on gps data. In *UbiComp*, 2008.

[Zheng, 2015] Yu Zheng. Trajectory data mining: An overview. *Acm Transactions on Intelligent Systems & Technology*, 6(3):1–41, 2015.

[Zhuang *et al.*, 2017] Chenyi Zhuang, Nicholas Jing Yuan, Ruihua Song, Xing Xie, and Qiang Ma. Understanding People Lifestyles: Construction of Urban Movement Knowledge Graph from GPS Trajectory. In *IJCAI*, 2017.