Context-aware Variational Trajectory Encoding and Human Mobility Inference

Fan Zhou
University of Electronic Science and Technology of China
fan.zhou@uestc.edu.cn

Xiaoli Yue
University of Electronic Science and Technology of China
yue.xiaol16@gmail.com

Goce Trajcevski
Iowa State University, Ames
gocet25@iastate.edu

Ting Zhong*
University of Electronic Science and Technology of China
zhongting@uestc.edu.cn

Kunpeng Zhang
University of Maryland, College park
kzhang@rhsmith.umd.edu

ABSTRACT

Unveiling human mobility patterns is an important task for many downstream applications like point-of-interest (POI) recommendation and personalized trip planning. Compelling results exist in various sequential modeling methods and representation techniques. However, discovering and exploiting the *context* of trajectories in terms of abstract topics associated with the motion can provide a more comprehensive understanding of the dynamics of patterns. We propose a new paradigm for moving pattern mining based on learning trajectory context, and a method – *Context-Aware Variational Trajectory Encoding and Human Mobility Inference (CATHI)* – for learning user trajectory representation via a framework consisting of: (1) a variational encoder and a recurrent encoder; (2) a variational attention layer; (3) two decoders. We simultaneously tackle two subtasks: (T1) recovering user routes (*trajectory reconstruction*); and (T2) predicting the trip that the user would travel (*trajectory prediction*). We show that the encoded contextual trajectory vectors efficiently characterize the hierarchical mobility semantics, from which one can decode the implicit meanings of trajectories. We evaluate our method on several public datasets and demonstrate that the proposed CATHI can efficiently improve the performance of both subtasks, compared to state-of-the-art approaches.

KEYWORDS

human mobility, variational inference, encoder-decoder

ACM Reference Format:
Fan Zhou, Xiaoli Yue, Goce Trajcevski, Ting Zhong, and Kunpeng Zhang. 2019. Context-aware Variational Trajectory Encoding and Human Mobility Inference. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313608

1 INTRODUCTION

The past decade has witnessed a rapid growth of both academic and practical interest on mining human mobility patterns from location-based social networks (LBSN) such as Twitter and Foursquare. Availability of large volumes of LBSN data has spurred research in studying user behavior and movement patterns [13, 31], point-of-interest (POI) recommendation [16], trip planning [22], along with various privacy-preserving issues [28].

Majority of the existing studies focus on modeling the sequential patterns at *location-* or *POI*-level. Various techniques such as leveraging Markov Chain (MC), Matrix Factorization (MF), Recurrent Neural Networks (RNN), or the word2vec framework have been developed towards various application tasks. For example, in POI recommendation, the state-of-the-art methods are MF [20] and RNN [26] based models that learn user preference over POIs and include the contextual influence of neighboring POIs for generating implicit user-location feedback matrices [25]. More recently, the representation learning such as word2vec [27] has been introduced for embedding POI (and associated features) into latent space [12, 13, 40]. These approaches operate at a fine-grained POI level – which is also the case for trip recommendations, where a sequence of POIs are organized as an itinerary, with various other features (e.g., location, time, distance, user profile and social interactions, etc.) incorporated. In addition, it is expected to learn the POI transitions with MC model [6] or maximize the reward of the selected POIs, e.g., through Monte Carlo Tree Search (MCTS) [21].

Complementary to these, some trajectory mining tasks aim at uncovering implicit mobility patterns, clustering or classifying trajectories – which are typically reduced to location-level learning. For example, sequential and periodic pattern [9, 35] mining requires measuring location or trajectory (a sequence of locations) similarity with various time-series metrics [8], while trajectory clustering groups together similar ones. However, the *Trajectory Context Learning (TCL)* focuses on understanding and encoding the semantics of a trajectory at *trajectory level.* As a higher level of mobility semantic learning, discovering the context of trajectories may enable more comprehensive understanding of the human mobility patterns and provide additional benefits for downstream applications such as POI/trip recommendation.

In this paper, we take initial steps towards learning trajectory context by adapting the problem to an encoder-decoder framework,

*Corresponding author.

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313608

inspired by recent progress of representation learning in natural languages processing (NLP). Specifically, we investigate the TCL problem by simultaneously addressing two (sub)tasks: (T1) recovering the user's route by (*Trajectory Reconstruction*) and (T2) predicting the trip that she would travel (*Trajectory Prediction*). To accomplish this, we propose the *Context-Aware Variational Trajectory Encoding and Human Mobility Inference (CATHI)* method for embedding the contexts of trajectories in an unsupervised manner within a seq2seq framework which consists of two encoders and two decoders, tuned with a novel variational attention mechanism. The two sub-tasks of TCL problem are then trained in parallel with the proposed model. In summary, our main contributions are:

- We address the TCL problem in LBSN applications as a novel learning paradigm for analyzing human mobility patterns. We propose an approach for encoding the semantics of trajectories and inferring the trajectory context, which is the first context-aware trajectory learning model and opens a new perspective for understanding user check-in behaviors.
- The proposed CATHI method leverages two variational latent factors which are respectively the traditional latent variables and the variational hidden state attention, for learning trajectory-level context. This novel variational architecture is not only capable of estimating the probability density and optimizing the lower bound of the data likelihood, but also captures the sequential and temporal characteristics of human mobility.
- We address the trajectory prediction and reconstruction simultaneously using the proposed CATHI method, which is evaluated on several public LBSN datasets and compared to competing models, demonstrating that it achieves state-of-the-art performance, which also reflects upon the efficiency of TCL.

2 OUR PROPOSED METHOD: CATHI

We now formally define the TCL problem and introduce the CATHI method consisting of two subtasks – (T1): trajectory reconstruction and (T2): trajectory prediction.

Given a set of POIs $\{l_1, l_2, ..., l_m\}$ and a set of trajectories $\mathbf{T} = \{T_1, T_2, ..., T_n\}$, where T_i is a sequence of POIs generated by a particular user, let $\mathbf{T}^r$ denote the training data consisting of a set of variable-length trajectories generated by certain users in a particular area (e.g., in a city or a scenic spot). We denote the t^{th} POI for trajectory T_i as l_i^t and its embedding vector (with word2vec [27]) as $\mathbf{v}_i^t \in \mathbb{R}^d$, where d is the POI embedding dimensionality.

2.1 Overview

The main objective of TCL is to produce two surrounding trajectories $\mathbf{T}_i^s = \{T_{i-1}, T_{i+1}\}$ for a given trajectory $T_i \in \mathbf{T}^r$. Generally, given a tuple $< T_{i-1}, T_i, T_{i+1} >$, where T_i is the i^{th} trajectory in $\mathbf{T}^r$, we are interested in coupling the reconstruction of T_{i-1} and prediction of T_{i+1}. That is, let $T_i = \{l_i^1, l_i^2, ..., l_i^N\}$ be a trajectory with N POIs, TCL tries to construct the previous trajectory $T_{i-1} = \{l_{i-1}^1, l_{i-1}^2, ..., l_{i-1}^Q\}$ and predict the next trajectory $T_{i+1} = \{l_{i+1}^1, l_{i+1}^2, ..., l_{i+1}^M\}$, respectively.

To this end, our CATHI model consists of four components (shown in Figure 1): (1) *recurrent encoder*, (2) *variational encoder*, (3) *variational attention layer*, and (4) *two decoders*. More specifically, a recurrent encoder maps a sequence of POIs to a set of hidden states $\{\mathbf{h}_i^1, \mathbf{h}_i^2, ..., \mathbf{h}_i^N\}$. Meanwhile, a variational encoder learns the distribution of the latent variable z. Subsequently, a variational attention on the hidden states of recurrent encoder learns the distribution of another latent variable $\mathbf{c}$. Finally, two decoders, conditioned on two latent variables z and $\mathbf{c}$, are employed to generate the previous trajectory and the next one respectively. Both decoders have a two-layer architecture in which the first-layer decoder $\texttt{Dec}_1$ alone can be employed to reconstruct the surrounding trajectories, and the second-layer decoder $\texttt{Dec}_2$ is added to refine the results of $\texttt{Dec}_1$.

2.2 Trajectory Context Encoding

Throughout this work, the Trajectory Context Encoding (TCE) relies on variational autoencoder [19] and Long Short-Term Memory (LSTM) [18] – one spatio-temporal LSTM for the encoder and two variants of LSTM for the decoders. Details will be presented below.

2.2.1 Recurrent Trajectory Encoder. The recurrent encoder is a bi-directional LSTM. At each time step t, one can obtain an annotation for each POI by concatenating the forward and backward hidden states, i.e., $\mathbf{h}_i^t = [\overrightarrow{\mathbf{h}}_i^t; \overleftarrow{\mathbf{h}}_i^t]$, interpreted as the representation of the trajectory $l_i^1, l_i^2, ..., l_i^t$. The last hidden state $\mathbf{h}_i^N$ thus represents the full trajectory T_i.

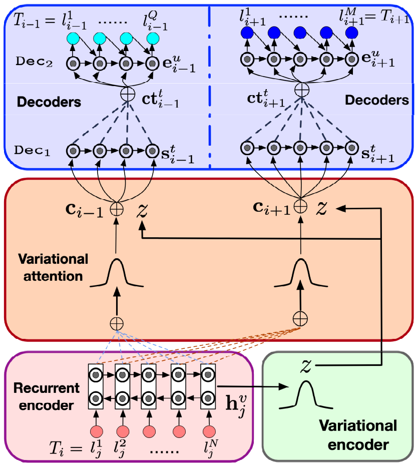


Figure 1: Trajectory context encoding with two encoders and two decoders with each having two layers.

Previous analysis shows that POIs' contextual information, such as geographical location and social information, has influence over user check-in behavior. Integrating them into the model might be helpful for better representation of users. However, vanilla LSTM does not consider the contextual information associated with POIs. To address this limitation, we present an extension of LSTM that is capable of encoding the spatio-temporal features in check-ins.

Given a sequence of check-ins $l_i^1, \cdots, l_i^t, \cdots, l_i^N$, each l_i^t associated with a check-in time τ_i^t and a geo-location g_i^t, we compute both time interval and distance between adjacent check-ins as $\Delta_{\tau_i^t} = \tau_i^t - \tau_i^{t-1}$ and $\Delta_{g_i^t} = g_i^t - g_i^{t-1}$, $t \in [1, N]$. Then we add a time gate $\mathbf{t}_i^t$ and a geographical gate $\mathbf{g}_i^t$ and modify the candidate gate $\tilde{\mathbf{h}}_i^t$ as

$$\begin{aligned}
\mathbf{t}_i^t &= \sigma(W_t\mathbf{v}_i^t + \sigma([\Delta_{\tau_i^t};\tau_i^t]W_t) + b_t) \\
\mathbf{g}_i^t &= \sigma(W_g\mathbf{v}_i^t + \sigma(\Delta_{g_i^t}W_g) + b_g) \\
[\mathbf{i}_i^t, \mathbf{f}_i^t, \mathbf{o}_i^t] &= \sigma(W\mathbf{v}_i^t + U\mathbf{h}_i^{t-1}) \\
\tilde{\mathbf{h}}_i^t &= \tanh(W'\mathbf{v}_i^t + U'(\mathbf{t}_i^t \odot \mathbf{g}_i^t \odot \mathbf{h}_i^{t-1})) \qquad (1) \\
\mathbf{m}_i^t &= \mathbf{f}_i^t\mathbf{m}_{i-1}^t + \mathbf{i}_i^t\tilde{\mathbf{h}}_i^t \\
\mathbf{h}_i^t &= \tanh(\mathbf{m}_i^t) \odot \mathbf{o}_i^t
\end{aligned}$$

where $\mathbf{v}_i^t$ is the embedding vector of the POI l_i^t, $[\Delta_{\tau_t^t};\tau_t^t]$ is a concatenation of time interval and the current check-in time; $\mathbf{i}_i^t, \mathbf{f}_i^t, \mathbf{o}_i^t$ are input, forget and output gates in vanilla LSTM; W and U are parameter matrices w.r.t different gates. Now the two new gates capture spatio-temporal preference of users, which is used to control the influence of previous hidden state $\mathbf{h}_i^{t-1}$ in Eq.(1).

Note that above contextual LSTM is employed in the recurrent encoder. The hidden state $\mathbf{h}_i^t$ contains the information from both the preceding POIs and the following POIs in T_i. Due to the tendency of RNNs to better represent recent inputs, the hidden state $\mathbf{h}_i^t$ will focus more on the POIs around current check-in l_i^t (cf. [1]). Instead of only attention to different words in NLP encoder such as [5], we follow the typical assumption in trajectory modeling that adjacent POIs have more (related) information than long-distance ones, both temporally and geographically. Such constraints are achieved by the proposed spatio-temporal gates. This sequence of hidden states, combined with the following variational variables, can later be used to compute context vectors in the decoder.

2.2.2 Variational Trajectory Encoder. We use a variational encoder [19] for encoding the trajectory as a latent variable z, which is then used to reconstruct the original trajectory under the generative model: given a trajectory T_i, its likelihood is

$$\begin{aligned}
&\log p_\theta(T_i) = \log p_\theta(T_i)\int_z q_\phi(z|T_i)\mathrm{d}z \geq\triangleq \mathcal{L}(T_i) \\
=&\mathbb{E}_{z\sim q_\phi(z|T_i)}[\log p_\theta(T_i|z)] - \mathbb{KL}\left[q_\phi(z|T_i)||p_\theta(z)\right] \qquad (2)
\end{aligned}$$

where $q_\phi(z|T_i)$ is an approximation to the true posterior $p_\theta(z|T_i)$ (a.k.a *recognition model* or *encoder*) parameterized by ϕ. $\mathbb{KL}\left[q_\phi(\cdot)||p_\theta(\cdot)\right]$ is the KL divergence between the learned latent posterior distribution $q(z|T_i)$ 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 the proposal $q(z|T_i)$ and the true distribution $p(z|T_i)$ – we can alternatively maximize the evidence lower bound (ELBO) $\mathcal{L}(T_i)$ w.r.t. both θ and ϕ, which are jointly trained with separate neural networks such as multi-layer perceptrons.

2.2.3 Variational attention. Traditional attention mechanisms [1] are used for dynamically aligning the input and output sequences. Specifically, we compute a probabilistic distribution by

$$\gamma_{vt} = \frac{\exp(\mathbf{s}_{i\pm1}^v W^\top \mathbf{h}_i^t)}{\sum_{t'=1}^N \exp(\mathbf{s}_{i\pm1}^v W^\top \mathbf{h}_i^{t'})} \qquad (3)$$

where $\mathbf{s}_{i\pm1}^v$ denotes the v^{th} hidden state of previous (next) trajectory, $\mathbf{h}_i^t$ is t^{th} hidden hidden state of recurrent encoder, and W is the parameter matrix needed to be learneded. Then, the attention vector $\mathbf{c}_{i\pm1}$ is calculated by summing the weighted input:

$$\mathbf{c}_{i\pm1}^v = \sum_{t=1}^N \gamma_{vt}\mathbf{h}_i^t \qquad (4)$$

which is fed into the decoder at v^{th} step. We will denote the attention vector as $\mathbf{c}$ for simplicity.

However, if we directly use above attention vector $\mathbf{c}$, this deterministic attention is powerful enough to reconstruct the input and may eliminate the influence of the variational encoder – a.k.a. "bypassing" phenomena in combining VAE and seq2seq models [3]. Therefore, we introduce a variational attention on the hidden states of the recurrent encoder inspired by [2]. That is, the attention vector $\mathbf{c}$ is also treated as a latent factor as z, both of which are combined to reconstruct the input data by maximizing the ELBO:

$$\begin{aligned}
\mathcal{L}(T_i) &= \mathbb{E}[\log p(T_i|z,\mathbf{c})] - \mathbb{KL}\left[q(z,\mathbf{c}|T_i)||p(z,\mathbf{c})\right] \\
&= \mathbb{E}[\log p(T_i|z,\mathbf{c})] - \rho\mathbb{KL}\left[q(z|T_i)||p(z)\right] \\
&\quad -\psi\mathbb{KL}\left[q(\mathbf{c}|T_i)||p(\mathbf{c})\right] \qquad (5)
\end{aligned}$$

which is derived based on the fact that $\mathbf{c}$ and z are conditionally independent given T_i. The prior $p(\mathbf{c})$ is initialized via a Gaussian $p(\mathbf{c}) = \mathcal{N}(\bar{\mathbf{h}}_i, \mathbf{I})$ with $\bar{\mathbf{h}}_i$ as the mean of the hidden states of the recurrent encoder. At the early stage of variational training, the terms $\mathbb{KL}\left[q(\cdot|T_i)||p(\cdot)\right]$ in Eq.(5) may discourage encoding interesting information into the latent variables z and $\mathbf{c}$, thereby easily resulting in model collapse – largely because of the strong learning capability of autoregressive models such as RNN as observed in [3]. One efficient solution is to control the weight of KL-divergence term by gradually increasing its co-efficiencies ρ and ψ (from 0 to 1), which is also called KL cost annealing [3].

2.2.4 Trajectory Decoders. As mentioned above, there are two decoders in CATHI: two LSTMs trained to reconstruct the previous trajectory T_{i-1} and predict the next trajectory T_{i+1}, respectively.
– **Decoder 1.** In the case of trajectory T_{i+1} decoding, the first decoder $\mathtt{Dec}_1$ predict next POI l_{i+1}^t given: (1) all the previously predicted POIs $\{l_{i+1}^1, l_{i+1}^2, ..., l_{i+1}^{t-1}\}$; (2) the variational attention vector $\mathbf{c}$; and (3) the variational latent variable z. For example, the decoder is trained to maximize the probability of $\{l_{i+1}^1, l_{i+1}^2, ..., l_{i+1}^M\}$ given trajectory T_i which can be decomposed into the ordered conditional probabilities:

$$p(\{l_{i+1}^1, l_{i+1}^2, ..., l_{i+1}^M\}) = \prod_{t=1}^M p(l_{i+1}^t|l_{i+1}^{<t}, \mathbf{c}, z) \qquad (6)$$

where $l_*^{<t}$ denote previous $t-1$ POIs in the next trajectory. Each conditional probability $p(l_{i+1}^t|l_{i+1}^{<t}, \mathbf{c}, z)$ depends on the context vectors $\mathbf{c}$, z and all the previous POIs $\{l_{i+1}^1, l_{i+1}^2, ..., l_{i+1}^{t-1}\}$. With the first LSTM decoder, the conditional probability can be modeled as:

$$p(l_{i+1}^t|l_{i+1}^{<t}, \mathbf{c}, z) = \mathrm{LSTM}(\mathbf{v}_{i+1}^{t-1}, \mathbf{c}, z, \mathbf{s}_{i+1}^{t-1}) \qquad (7)$$

where $\mathbf{s}_{i+1}^{t-1}$ is the hidden state of the decoder LSTM.

Therefore, given a tuple $< T_{i-1}, T_i, T_{i+1} >$, the objective of trajectory context encoding is to maximize the log-probabilities for the previous and next trajectories of T_i:

$$\mathcal{J} = \log\prod_t p(l_{i+1}^t|l_{i+1}^{<t}, \mathbf{c}_{i+1}, z)p(l_{i-1}^t|l_{i-1}^{<t}, \mathbf{c}_{i-1}, z) \qquad (8)$$

where the two decoders (for T_{i+1} and T_{i-1}) focus on different POIs in T_i with distinct context vectors $\mathbf{c}_{i\pm1}$, iterating through all the tuples in the dataset during each epoch of the TCL training.

– **Decoder 2.** Intuitively, the first decoder(s) already can be employed to reconstruct the surrounding trajectories of T_i. However, the generated trajectories (both previous and next one) suffer the "myopic" posterior in the sense that the POIs generated previously at time step t are not informed about the whole future in the sequence – recall that at time step t, only the POIs $l_*^{<t}$ are used in Equation (8) without considering possible POIs $l_*^{>t}$ thereafter. We now introduce a refinement process to CATHI when inferring the around trajectories.

Specifically, we add a second decoder $\mathtt{Dec}_2$ upon the first decoder $\mathtt{Dec}_1$ to refine the final trajectory. In contrast to $\mathtt{Dec}_1$, $\mathtt{Dec}_2$ exploits the information regarding all the POIs of the generated trajectories by $\mathtt{Dec}_1$ at each time step. In addition to the previous hidden state and previously generated POIs, $\mathtt{Dec}_2$ leverages the context vector $\mathbf{ct}_{i+1}^t$, the weighted sum of the hidden states of $\mathtt{Dec}_1$, to "polish" the generated trajectory, i.e., T_{i-1} and T_{i+1}. That is, $\mathtt{Dec}_2$ updates the context vector $\mathbf{ct}_{i+1}^t$ at time step t as:

$$\mathbf{ct}_{i+1}^t = \sum_{u=1}^{M} \beta_{ut} \mathbf{s}_{i+1}^u$$

This is actually another attention vector on the first decoder weighted by a probabilistic distribution

$$\beta_{ut} \propto \exp\{\mathbf{e}_{i+1}^u W^\intercal \mathbf{s}_{i+1}^t\} \tag{9}$$

where $\mathbf{e}_{i+1}^u$ is the hidden state of $\mathtt{Dec}_2$ and $\mathbf{s}_{i+1}^t$ is the hidden state of $\mathtt{Dec}_1$.

2.2.5 Inference. After describing how to learn the context of trajectories, we now turn our attention from training to trajectory inference for addressing the two subtasks – trajectory reconstruction and trajectory prediction. In many seq2seq based tasks such as neural machine translation [1] and dialogue generation [3], beam search is the *de facto* decoding method at test time, due to the shrunk search space and reduced computational complexity. Beam search maintains a set of B highest-scoring beams which are prefixes of trajectories. At each time step, every possible POI is added to every existing beam, yielding a total of $B \times D$ beams where D is the number of all possible POIs. Then we discard all but the B most likely beams according to the log-likelihood of the trained model. This procedure is repeated until a segment ends with the end-of-trajectory token or reaches the preset maximum length of trajectories. The segment that ends with end-of-trajectory token will then be moved to the set of complete hypotheses. Finally, we choose the one with highest probability from the hypotheses.

3 EVALUATION

We now present the results of our evaluation regarding the performance of TCL solutions with the two context-aware trajectory inference tasks: trajectory reconstruction and prediction.

Datasets. We conducted our experiments on three datasets shown in Table 1. The trajectories extracted from Flickr photos and videos are typically used for trip recommendation [6, 21, 22]. **Toronto, Osaka, Glasgow and Edinburgh** data were built by Lim et al. [22]; **Melbourne** data were built by [6], and **Cali.Adv, Hollywood, Epcot, DisneyLand and MagicKing** data were built by [21]. **Foursquare** dataset [34] is often used for POI recommendation and contains 573,703 check-ins in Tokyo collected for approximately 10 months (from 12 April 2012 to 16 February 2013). We randomly selected 400 users and their historical trajectories for evaluation. **Geolife** dataset [38] was collected in Microsoft Geolife project in a period of over five years (from April 2007 to August 2012). Since the trajectories in original Geolife dataset contain only GPS points (longitude and latitude), we cluster all points based on their locations to obtain 4,796 POIs – that is, we save the two digits after the decimal point of longitude and latitude. For all the datasets, we randomly chose 90% trajectories for training and the rest for testing.

Table 1: Descriptives of datasets.

Dataset	#POI	#Trajectory	#User
Flickr@Edinburgh	29	5,028	1,454
Flickr@Glasgow	29	2,227	601
Flickr@Melbourne	87	5,106	1,000
Flickr@Osaka	29	1,115	450
Flickr@Toronto	30	6,057	1,395
Flickr@Cali.Adv	25	6,907	2,593
Flickr@Hollywood	13	3,858	1,972
Flickr@Epcot	17	5,816	2,725
Flickr@Disneyland	31	11,758	3,704
Flickr@MagicKing	27	8,126	3,342
Foursquare@Tokyo	30	1,1254	400
Geolife	4,796	24,943	179

We note that the two tasks evaluated in our experiments are predicting and reconstructing a sequence, which are more complicated than location prediction and/or recommendation which requires predicting/recommending a point. Thus, for the two tasks, the datasets are large enough to evaluate the methods, which is quite different from the datasets used in venue recommendation/prediction [9, 26].

Baselines. Since TCL is a novel problem proposed in this work, we compare our method with several state-of-the-art approaches from the area of sequential pattern modeling and trajectory recommendation, with two metrics for measuring accuracy and visiting-order of POIs in each reconstructed and predicted trajectory. The baselines consist of:

- **Random**: A naive approach chooses at random to construct a trajectory with desired length.
- **POIPopu** [11]: It selects the most popular and unvisited POI at each time.
- **POIRank** [6]: It plans a trajectory by first ranking POIs with rankSVM and then connects them according to ranking scores.
- **Markov and MRank** [6]: Markov considers the POI-POI transition probabilities and plans a trajectory by maximizing the transition likelihood. Markov-Rank learns both POI ranking and Markov transition.
- **MPath and MPRank** [6]: MarkovPath (MPath) and MarkovPath-Rank (MPRank) eliminate sub-tours in Markov and Markov-Rank by finding the best path using an integer programming.
- **ST-RNN** [23]: is a RNN-based method incorporating spatial and temporal context for predicting the next location.

Table 2: Trajectory Reconstruction Performance Among Different Algorithms on Flickr Data.

	Edinburgh	Glasgow	Melbourne	Osaka	Toronto	Cali.Adv.	Hollywood	Epcot	Disneyland	MagicKing
F_1										
Random	0.474 ± 0.071	0.502 ± 0.048	0.442 ± 0.098	0.497 ± 0.051	0.492 ± 0.058	0.458 ± 0.076	0.504 ± 0.043	0.488 ± 0.057	0.438 ± 0.086	0.440 ± 0.080
POIPopu	0.581 ± 0.214	0.589 ± 0.202	0.518 ± 0.174	0.569 ± 0.165	0.534 ± 0.147	0.512 ± 0.159	0.593 ± 0.189	0.618 ± 0.241	0.531 ± 0.187	0.565 ± 0.221
POIRank	0.583 ± 0.213	0.594 ± 0.208	0.531 ± 0.187	0.551 ± 0.170	0.591 ± 0.204	0.517 ± 0.167	0.596 ± 0.198	0.649 ± 0.232	0.537 ± 0.194	0.564 ± 0.220
Markov	0.550 ± 0.188	0.577 ± 0.194	0.455 ± 0.142	0.536 ± 0.173	0.566 ± 0.194	0.509 ± 0.173	0.500 ± 0.174	0.469 ± 0.165	0.500 ± 0.155	0.460 ± 0.150
MPath	0.559 ± 0.170	0.585 ± 0.193	0.486 ± 0.137	0.540 ± 0.197	0.568 ± 0.186	0.486 ± 0.145	0.536 ± 0.157	0.476 ± 0.159	0.496 ± 0.155	0.449 ± 0.130
MRank	0.601 ± 0.225	0.602 ± 0.232	0.490 ± 0.187	0.555 ± 0.201	0.590 ± 0.248	0.517 ± 0.183	0.580 ± 0.193	0.554 ± 0.208	0.512 ± 0.187	0.521 ± 0.194
MPRank	0.578 ± 0.215	0.611 ± 0.231	0.496 ± 0.168	0.574 ± 0.217	0.599 ± 0.236	0.528 ± 0.162	0.573 ± 0.173	0.583 ± 0.200	0.539 ± 0.201	0.543 ± 0.185
ST-RNN	0.665 ± 0.147	0.678 ± 0.171	0.679 ± 0.142	0.703 ± 0.159	0.708 ± 0.123	0.648 ± 0.142	0.645 ± 0.152	0.637 ± 0.168	0.611 ± 0.157	0.627 ± 0.160
DeepMove	0.679 ± 0.153	0.687 ± 0.161	0.688 ± 0.132	0.700 ± 0.152	0.726 ± 0.119	0.645 ± 0.128	0.672 ± 0.102	0.643 ± 0.170	0.618 ± 0.155	0.662 ± 0.158
CARA	0.689 ± 0.155	0.709 ± 0.112	0.669 ± 0.143	0.745 ± 0.144	0.722 ± 0.122	0.688 ± 0.130	0.645 ± 0.126	0.673 ± 0.148	0.631 ± 0.157	0.673 ± 0.150
CATHI	**0.778 ± 0.128**	**0.807 ± 0.121**	**0.715 ± 0.125**	**0.913 ± 0.079**	**0.843 ± 0.115**	**0.755 ± 0.127**	**0.796 ± 0.123**	**0.810 ± 0.136**	**0.742 ± 0.147**	**0.731 ± 0.155**
pairs-F_1										
Random	0.102 ± 0.120	0.052 ± 0.144	0.015 ± 0.073	0.074 ± 0.160	0.088 ± 0.123	0.053 ± 0.119	0.095 ± 0.158	0.074 ± 0.143	0.040 ± 0.098	0.054 ± 0.112
POIPopu	0.207 ± 0.366	0.196 ± 0.382	0.117 ± 0.280	0.152 ± 0.316	0.100 ± 0.266	0.113 ± 0.239	0.217 ± 0.333	0.308 ± 0.409	0.150 ± 0.274	0.237 ± 0.354
POIRank	0.212 ± 0.368	0.208 ± 0.389	0.138 ± 0.305	0.130 ± 0.315	0.209 ± 0.380	0.123 ± 0.254	0.225 ± 0.346	0.345 ± 0.415	0.165± 0.292	0.235 ± 0.354
Markov	0.169 ± 0.326	0.180 ± 0.367	0.037 ± 0.169	0.116 ± 0.313	0.172 ± 0.355	0.119 ± 0.280	0.122 ± 0.280	0.153 ± 0.460	0.146 ± 0.493	0.197 ± 0.314
MPath	0.130 ± 0.292	0.177 ± 0.370	0.049 ± 0.202	0.153 ± 0.228	0.155 ± 0.344	0.088 ± 0.283	0.144 ± 0.256	0.167 ± 0.440	0.149 ± 0.485	0.123 ± 0.375
MRank	0.244 ± 0.395	0.265 ± 0.412	0.114 ± 0.273	0.164 ± 0.345	0.263 ± 0.420	0.181 ± 0.284	0.221 ± 0.311	0.213 ± 0.319	0.187 ± 0.306	0.188 ± 0.305
MPRank	0.204 ± 0.369	0.265 ± 0.415	0.080 ± 0.256	0.220 ± 0.379	0.260 ± 0.411	0.192 ± 0.311	0.176 ± 0.265	0.228 ± 0.353	0.196 ± 0.317	0.215 ± 0.332
ST-RNN	0.436 ± 0.172	0.334 ± 0.075	0.355 ± 0.109	0.445 ± 0.188	0.418 ± 0.161	0.314 ± 0.132	0.347 ± 0.116	0.309 ± 0.159	0.351 ± 0.175	0.362 ± 0.196
DeepMove	0.448 ± 0.228	0.445 ± 0.198	0.440 ± 0.194	0.542 ± 0.202	0.582 ± 0.189	0.454 ± 0.223	0.439 ± 0.215	0.402 ± 0.234	0.412 ± 0.222	0.398 ± 0.266
CARA	0.440 ± 0.230	0.403 ± 0.199	0.402 ± 0.188	0.528 ± 0.204	0.574 ± 0.182	0.446 ± 0.210	0.427 ± 0.210	0.404 ± 0.212	0.408 ± 0.206	0.412 ± 0.288
CATHI	**0.556 ± 0.210**	**0.609 ± 0.187**	**0.474 ± 0.174**	**0.782 ± 0.201**	**0.617 ± 0.179**	**0.517 ± 0.205**	**0.565 ± 0.207**	**0.590 ± 0.204**	**0.493 ± 0.211**	**0.503 ± 0.240**

- **DeepMove** [9]: a most recent POI prediction method learning user periodical patterns with attention mechanism and the sequential patterns within the recurrent neural networks.
- **CARA** [26]: a most recent POI recommendation method jointly learns the user dynamic preference and contextual information associated with check-ins under the GRU [7] architecture.

Note that the last three methods (ST-RNN, DeepMove and CARA) are not targeting sequence learning. We extend them to infer a trajectory by predicting (or reconstruct) a sequence of POIs iteratively – e.g., after predicting POI l_*^{i+1}, we incorporate l_*^{i+1} into a LSTM for predicting l_*^{i+2}, and so on.

Metrics We use F_1 and pairs-F_1 score to evaluate the two subtasks in TCL following previous works [6, 21].

3.1 Experimental Observations

Overall performance: Table 2 and 3 respectively show the experimental results of trajectory reconstruction and trajectory prediction on 10 Flickr datasets, while Table 4 shows the results on Geolife and Foursquare (we exclude others due to the limited space). Note that the values before and after '±' are respectively the mean and the standard deviation.

Firstly, CATHI consistently yields the best performance on two subtasks for both metrics, while exhibiting significant improvement over the baselines. The effectiveness of CATHI lies in its trajectory-level embedding which simultaneously encodes the context information and captures the sequential patterns of trajectories (at both POI-level and trajectory-level). On the contrary, ST-RNN, DeepMove and CARA are deep learning based methods learning mobility at the POI-level and usually exhibit the secondary performance. In addition, one can also observe that CATHI achieves the lower value of standard deviation than other methods (except Random which samples around the mean and thus has minimum value of standard deviation), which means that the results of CATHI are more stable.

On the other hand, CATHI employs contextual recurrent networks as the encoders and decoders and proves their superior performance compared to MC-based models, especially in terms of trajectory ordering (measured by pairs-F_1). Note that the performance of MC-based models may even drop to the same level (if not worse) of random and greedy methods. This is because MC (or its variants) learns the transition patterns at the POI-level and works well when predicting the next one or previous one POI. However, both two subtasks in our work require to form a POI sequence.

Moreover, we observe that the performance of trajectory reconstruction is slightly better than the trajectory prediction, which is also true for other baselines. This phenomenon motivates a decomposition of the subtasks into two steps: (1) reconstruct the trajectories of individuals, and (2) incorporate the results to improve the prediction accuracy.

Validity of components: Recall that our CATHI model consists of several functional parts. In order to investigate the effectiveness of each component, we compare a series of variants of CATHI, including: (1) CATHI-1, which uses a basic seq2seq model, with contextual LSTM for trajectory context learning; (2) CATHI-2, which incorporates a deterministic attention mechanism into CATHI-1; (3) CATHI-3, which further incorporates variational encoder-decoder into CATHI-2; (4) CATHI-4, which considers attention vector as another variational variable; and (5) the CATHI, which can be considered as adding the second decoder on CATHI-4. Obviously, these methods gradually builds up our CATHI method. Thus, we can understand the rationale behind by scrutinizing the performance of each module.

Figure 2 plots the performance of the five methods on five Flickr datasets. From the trajectory prediction task, we can see that each component of CATHI contributes to the trajectory context learning. Among various modules, deterministic attention is the most influential one. We also observe that our variational attention works well across all comparisons which proves that variational encoder-decoder can be used to enhance seq2seq model, but should consider the attention vectors as variational variables. Finally, CATHI outperforming CATHI-4 demonstrates that the second decoder indeed improve the TCL performance by refining the results of using one decoder, especially for better learning visiting order – it largely improves the pairs-F_1 results.

4 RELATED WORK

POI-level Models. Modeling the patterns of user check-ins is a primary step in most application tasks such as POI recommendation

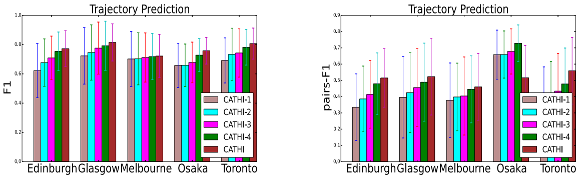
Table 3: Trajectory Prediction Performance Among Different Algorithms on Flickr Data.

	Edinburgh	Glasgow	Melbourne	Osaka	Toronto	Cali.Adv.	Hollywood	Epcot	Disneyland	MagicKing
					F_1					
Random	0.474 ± 0.071	0.503 ± 0.048	0.442 ± 0.098	0.500 ± 0.049	0.492 ± 0.058	0.424 ± 0.091	0.498 ± 0.043	0.438 ± 0.078	0.408 ± 0.099	0.406 ± 0.095
POIPopu	0.515 ± 0.154	0.542 ± 0.166	0.480 ± 0.151	0.587 ± 0.227	0.531 ± 0.167	0.494 ± 0.160	0.543 ± 0.192	0.543 ± 0.186	0.455 ± 0.166	0.465 ± 0.172
POIRank	0.518 ± 0.173	0.565 ± 0.205	0.511 ± 0.186	0.547 ± 0.173	0.568 ± 0.198	0.483 ± 0.156	0.516 ± 0.192	0.510 ± 0.157	0.453 ± 0.162	0.457 ± 0.153
Markov	0.520 ± 0.176	0.551 ± 0.169	0.457 ± 0.142	0.630 ± 0.228	0.557 ± 0.194	0.489 ± 0.172	0.524 ± 0.193	0.541 ± 0.191	0.456 ± 0.131	0.470 ± 0.159
MPath	0.550 ± 0.169	0.563 ± 0.182	0.488 ± 0.130	0.623 ± 0.189	0.575 ± 0.200	0.506 ± 0.169	0.541 ± 0.184	0.540 ± 0.182	0.469 ± 0.128	0.483 ± 0.171
MRank	0.541 ± 0.227	0.637 ± 0.244	0.465 ± 0.207	0.627 ± 0.248	0.612 ± 0.241	0.500 ± 0.195	0.531 ± 0.196	0.563 ± 0.193	0.464 ± 0.171	0.443 ± 0.147
MPRank	0.545 ± 0.248	0.642 ± 0.261	0.464 ± 0.222	0.610 ± 0.252	0.620 ± 0.261	0.492 ± 0.186	0.545 ± 0.170	0.564 ± 0.192	0.465 ± 0.108	0.467 ± 0.134
ST-RNN	0.613 ± 0.115	0.711 ± 0.086	0.669 ± 0.091	0.645 ± 0.140	0.689 ± 0.102	0.624 ± 0.113	0.643 ± 0.081	0.638 ± 0.108	0.642 ± 0.126	0.669 ± 0.121
DeepMove	0.602 ± 0.148	0.732 ± 0.162	0.701 ± 0.156	0.672 ± 0.136	0.712 ± 0.134	0.706 ± 0.149	0.682 ± 0.147	0.652 ± 0.183	0.656 ± 0.166	0.682 ± 0.155
CARA	0.616 ± 0.142	0.726 ± 0.158	0.704 ± 0.166	0.670 ± 0.140	0.704 ± 0.129	0.702 ± 0.146	0.680 ± 0.142	0.655 ± 0.115	0.650 ± 0.161	0.677 ± 0.167
CATHI	**0.772 ± 0.123**	**0.815 ± 0.127**	**0.722 ± 0.148**	**0.758 ± 0.091**	**0.807 ± 0.106**	**0.737 ± 0.129**	**0.777 ± 0.139**	**0.781 ± 0.141**	**0.737 ± 0.151**	**0.713 ± 0.152**
					pairs-F_1					
Random	0.041 ± 0.114	0.076 ± 0.116	0.012 ± 0.052	0.124 ± 0.136	0.046 ± 0.129	0.046 ± 0.103	0.095 ± 0.144	0.068 ± 0.126	0.042 ± 0.094	0.048 ± 0.100
POIPopu	0.097 ± 0.250	0.116 ± 0.301	0.064 ± 0.215	0.225 ± 0.412	0.109 ± 0.295	0.100 ± 0.232	0.169 ± 0.313	0.156 ± 0.299	0.080 ± 0.215	0.090 ± 0.232
POIRank	0.114 ± 0.283	0.174 ± 0.375	0.120 ± 0.296	0.130 ± 0.315	0.175 ± 0.362	0.091 ± 0.228	0.136 ± 0.298	0.108 ± 0.245	0.076 ± 0.199	0.072 ± 0.196
Markov	0.121 ± 0.287	0.129 ± 0.317	0.039 ± 0.174	0.283 ± 0.234	0.160 ± 0.348	0.108 ± 0.276	0.149 ± 0.302	0.176 ± 0.319	0.088 ± 0.341	0.086 ± 0.243
MPath	0.101 ± 0.285	0.144 ± 0.343	0.041 ± 0.188	0.185 ± 0.367	0.177 ± 0.374	0.105 ± 0.271	0.145 ± 0.297	0.145 ± 0.304	0.091 ± 0.334	0.087 ± 0.252
MRank	0.205 ± 0.348	0.319 ± 0.435	0.113 ± 0.280	0.313 ± 0.443	0.288 ± 0.411	0.123 ± 0.297	0.168 ± 0.322	0.221 ± 0.360	0.116 ± 0.243	0.142 ± 0.375
MPRank	0.229 ± 0.386	0.341 ± 0.460	0.124 ± 0.308	0.297 ± 0.439	0.325 ± 0.449	0.110 ± 0.275	0.166 ± 0.286	0.195 ± 0.327	0.104 ± 0.315	0.156 ± 0.324
ST-RNN	0.301 ± 0.147	0.392 ± 0.113	0.373 ± 0.122	0.305 ± 0.159	0.418 ± 0.172	0.347 ± 0.153	0.391 ± 0.109	0.303 ± 0.120	0.327 ± 0.176	0.328 ± 0.145
DeepMove	0.334 ± 0.152	0.398 ± 0.142	0.392 ± 0.156	0.362 ± 0.162	0.424 ± 0.188	0.352 ± 0.195	0.428 ± 0.198	0.402 ± 0.202	0.420 ± 0.199	0.368 ± 0.198
CARA	0.368± 0.198	0.340 ± 0.209	0.391 ± 0.183	0.394 ± 0.186	0.422 ± 0.119	0.360 ± 0.202	0.422 ± 0.201	0.416 ± 0.220	0.428 ± 0.212	0.385 ± 0.221
CATHI	**0.515 ± 0.180**	**0.523 ± 0.236**	**0.460 ± 0.205**	**0.516 ± 0.199**	**0.559 ± 0.205**	**0.431 ± 0.195**	**0.515 ± 0.201**	**0.533 ± 0.206**	**0.429 ± 0.209**	**0.422 ± 0.212**

Table 4: F_1 and pairs-F_1 Comparisons Among Different Algorithms on Geolife and Foursquare data.

		Trajectory Reconstruction		**Trajectory Prediction**	
	Methods	F1	Pairs-F1	F1	Pairs-F1
Geolife	Random	0.148±0.077	0.048±0.086	0.141±0.067	0.044±0.083
	POIPopu	0.190±0.138	0.003±0.026	0.163±0.122	0.114±0.092
	POIRank	0.188±0.133	0.002±0.025	0.157±0.124	0.128±0.122
	ST-RNN	0.502±0.111	0.158±0.088	0.509±0.107	0.223±0.138
	DeepMove	0.490±0.212	0.203±0.223	0.502±0.221	0.228±0.232
	CARA	0.502±0.111	0.158±0.088	0.509±0.107	0.223±0.138
	CATHI	**0.536±0.109**	**0.318±0.110**	**0.520±0.102**	**0.358±0.104**
Foursquare	Random	0.508±0.036	0.026±0.107	0.489±0.040	0.028±0.105
	POIPopu	0.555±0.158	0.117±0.307	0.532±0.170	0.109±0.305
	POIRank	0.583±0.189	0.172±0.365	0.550±0.189	0.142±0.342
	Markov	0.533±0.153	0.074±0.247	0.551±0.159	0.111±0.307
	MPath	0.527±0.143	0.065±0.236	0.599±0.124	0.147±0.209
	ST-RNN	0.609±0.153	0.333±0.210	0.642±0.116	0.342±0.163
	DeepMove	0.702±0.167	0.412±0.212	0.678±0.121	0.359±0.192
	CARA	0.719±0.169	0.425±0.228	0.684±0.119	0.365±0.166
	CATHI	**0.809±0.145**	**0.616±0.190**	**0.749±0.087**	**0.473±0.150**

and trip planning. The sequential patterns of check-ins are usually modeled by various machine learning techniques, such as Markov Chains (MC) [6, 17], matrix/tensor factorization [20], pairwise ranking model [16], Recurrent Neural Networks (RNN) [9, 13, 13, 23, 26] and distributed representation methods [10, 24, 32, 33, 37] (e.g. word2vec [27]). Trip planning leverages the spatio-temporal check-ins for recommending a sequence of POIs (POI-level modeling paradigm) and a use of MC to model the POI→POI transition by learning from historical behavior and trajectories have been discussed in [6, 21]. This line of researches focuses on modeling the sequential patterns of POIs (combined with various features of POIs and users) for solving different application tasks. However, they fail to incorporate the *context of trajectories* which is the objective of our work.



(a) F_1.

(b) pairs-F_1.

Figure 2: Comparison of modules in CATHI.

Trajectory Pattern Learning. Uncovering semantic patterns characterizing human trajectory has been studied extensively, and the results can be categorized into: (1) *statistical patterns learning*: measuring and quantifying models such as continuous-time random-walk [4] and Lévy flight [15], accounting for characteristics of individual human trajectories [29]. (2) *similarity mining*: measuring the similarity or distance between two trajectories, such as Dynamic Time Warping or Edit distance [8] – which may exploit the uniqueness and regularity of human mobility. (3) *periodical pattern mining*: finding (sub-)sequences and periodical motion patterns, enabling travel recommendation [6, 21], life pattern understanding [23, 30, 36], recovering trajectories associated with users [13, 39] and next location prediction [9, 14, 23]. The numerous works in semantic trajectory mining have not formally defined (and addressed) the *trajectory context learning (TCL)*, which is part of our study. Our proposed CATHI is also different from conventional approaches in that it captures higher level of semantics and provides more comprehensive understanding of human mobility by learning long-short term dependency at the trajectory level.

5 CONCLUSIONS

We introduced a novel problem – Trajectory Context Learning (TCL) – aiming at mining of high-level human motion patterns. Towards its solution, we relied on Trajectory Context Embedding and proposed CATHI – a method for jointly learning hierarchical and sequential patterns of trajectories, beneficial for many downstream tasks needing inference of human trajectories. As demonstrated, TCL provides valuable insights and promising guidelines for further investigations of intricacies of motion patterns. As part of our future work we plan to investigate how to leverage it for modeling the topics of sequential patterns, and its use in a wider range of applications, such as POI recommendation and road planning.

ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China (Grant No.61602097 and No.61472064), NSF grants III 1213038 and CNS 1646107, and ONR grant N00014-14-10215.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

[2] Hareesh Bahuleyan, Lili Mou, Olga Vechtomova, and Pascal Poupart. 2018. Variational Attention for Sequence-to-Sequence Models. In *COLING*.

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

[4] D Brockmann, L Hufnagel, T Geisel, and Roger M Whitaker. 2006. The scaling laws of human travel. *Nature* (2006).

[5] Ji Li Zhipeng Cai, Mingyuan Yan, and Yingshu Li. 2016. Sequential Short-Text Classification with Recurrent and Convolutional Neural Networks. In *NAACL-HLT*.

[6] Dawei Chen, Cheng Soon Ong, and Lexing Xie. 2016. Learning Points and Routes to Recommend Trajectories. In *CIKM*.

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[8] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn J. Keogh. 2008. Querying and mining of time series data: experimental comparison of representations and distance measures. In *PVLDB*.

[9] 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 *WWW*.

[10] Shanshan Feng, Gao Cong, Bo An, and Yeow Meng Chee. 2017. POI2Vec: Geographical Latent Representation for Predicting Future Visitors. *AAAI* (2017).

[11] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. 2015. Personalized Ranking Metric Embedding for Next New POI Recommendation. In *IJCAI*.

[12] Qiang Gao, Goce Trajcevski, Fan Zhou, Kunpeng Zhang, Ting Zhong, and Fengli Zhang. 2018. Trajectory-based Social Circle Inference. In *SIGSPATIAL*.

[13] Qiang Gao, Fan Zhou, Kunpeng Zhang, and Goce Trajcevski. 2017. Identifying Human Mobility via Trajectory Embeddings. In *IJCAI*.

[14] Qiang Gao, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Zhong Ting, and Fengli Zhang. 2019. Predicting Human Mobility via Variational Attention. In *WWW*.

[15] Marta C González, César A Hidalgo, Albert-László Barabási, A. L. BarabÃąsi, and Wang. 2008. Understanding individual human mobility patterns. *Nature* (2008).

[16] Jing He, Xin Li, and Lejian Liao. 2017. Category-aware Next Point-of-Interest Recommendation via Listwise Bayesian Personalized Ranking.. In *IJCAI*.

[17] Jing He, Xin Li, Lejian Liao, Dandan Song, and William K Cheung. 2016. Inferring a Personalized Next Point-of-Interest Recommendation Model with Latent Behavior Patterns. In *AAAI*.

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

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

[20] Xutao Li, Gao Cong, Xiao Li Li, Tuan Anh Nguyen Pham, and Shonali Krishnaswamy. 2015. Rank-GeoFM: A Ranking based Geographical Factorization Method for Point of Interest Recommendation. In *SIGIR*.

[21] Kwan Hui Lim, Jeffrey Chan, Shanika Karunasekera, and Christopher Leckie. 2017. Personalized Itinerary Recommendation with Queuing Time Awareness. *SIGIR* (2017).

[22] Kwan Hui Lim, Jeffrey Chan, Christopher Leckie, and Shanika Karunasekera. 2015. Personalized tour recommendation based on user interests and points of interest visit durations. In *IJCAI*.

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

[24] Xin Liu, Yong Liu, and Xiaoli Li. 2016. Exploring the Context of Locations for Personalized Location Recommendations. *IJCAI* (2016).

[25] Yiding Liu, Tuan-Anh Pham, Gao Cong, and Quan Yuan. 2017. An Experimental Evaluation of Point-of-interest Recommendation in Location-based Social Networks. *PVLDB* (2017).

[26] Jarana Manotumruksa, Craig Macdonald, and Iadh Ounis. 2018. A Contextual Attention Recurrent Architecture for Context-Aware Venue Recommendation. In *SIGIR*.

[27] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[28] Roman Schlegel, Chi-Yin Chow, Qiong Huang, and Duncan S. Wong. 2017. Privacy-Preserving Location Sharing Services for Social Networks. *IEEE Trans. Services Computing* 10, 5 (2017), 811–825.

[29] Chaoming Song, Tal Koren, Pu Wang, and Albert-László Barabási. 2010. Modelling the scaling properties of human mobility. *Nature Physics* (2010).

[30] Hongjian Wang and Zhenhui Li. 2017. Region representation learning via mobility flow. In *CIKM*.

[31] Fei Wu and Zhenhui Li. 2016. Where did you go: Personalized annotation of mobility records. In *CIKM*.

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

[33] Cheng Yang, Maosong Sun, Wayne Xin Zhao, Zhiyuan Liu, and Edward Y Chang. 2017. A Neural Network Approach to Jointly Modeling Social Networks and Mobile Trajectories. *TOIS* (2017).

[34] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu. 2015. Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 1 (2015), 129–142.

[35] Chao Zhang, Jiawei Han, Lidan Shou, Jiajun Lu, and Thomas La Porta. 2014. Splitter: Mining Fine-Grained Sequential Patterns in Semantic Trajectories. *PVLDB* (2014).

[36] Yuyu Zhang, Hanjun Dai, Chang Xu, Jun Feng, Taifeng Wang, Jiang Bian, Bin Wang, and Tie-Yan Liu. 2014. Sequential click prediction for sponsored search with recurrent neural networks. In *AAAI*.

[37] Shenglin Zhao, Tong Zhao, Irwin King, and Michael R Lyu. 2017. Geo-Teaser: Geo-Temporal Sequential Embedding Rank for Point-of-interest Recommendation. In *WWW*.

[38] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei Ying Ma. 2009. Mining interesting locations and travel sequences from GPS trajectories. In *WWW*.

[39] Fan Zhou, Qiang Gao, Kunpeng Zhang, Goce Trajcevski, Zhong Ting, and Fengli Zhang. 2018. Trajectory-User Linking via Variational AutoEncoder. In *IJCAI*.

[40] Fan Zhou, Bangying Wu, Yi Yang, Goce Trajcevski, Kunpeng Zhang, and Ting Zhong. 2018. vec2Link: Unifying Heterogeneous Data for Social Link Prediction. In *CIKM*.