Pre-training Context and Time Aware Location Embeddings from Spatial-Temporal Trajectories for User Next Location Prediction

Yan Lin,[1,2] **Huaiyu Wan,**[1,2,3] **Shengnan Guo,**[1,2,4] **Youfang Lin**[1,2,3,4*]

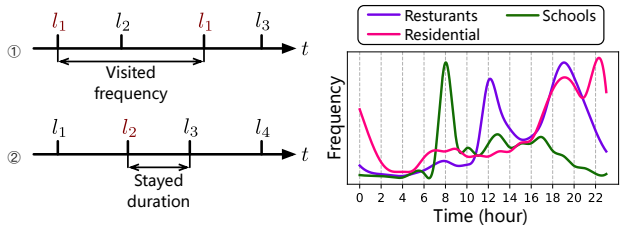
[1]School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China
[2]Beijing Key Laboratory of Traffic Data Analysis and Mining, Beijing, China
[3]CAAC Key Laboratory of Intelligent Passenger Service of Civil Aviation, Beijing, China
[4]Key Laboratory of Transport Industry of Big Data Appalication Technologies for Comprehensive Transport, Beijing, China
{ylincs, hywan, guoshn, yflin}@bjtu.edu.cn

Abstract

Pre-training location embeddings from spatial-temporal trajectories is a fundamental procedure and very beneficial for user next location prediction. In the real world, a location usually has variable functionalities under different contextual environments. If the exact functions of a location in the trajectory can be reflected in its embedding, the accuracy of user next location prediction should be improved. Yet, existing location embeddings pre-trained on trajectories are mostly based on distributed word representations, which mix a location's various functionalities into one latent representation vector. To address this problem, we propose a Context and Time aware Location Embedding (CTLE) model, which calculates a location's representation vector with consideration of its specific contextual neighbors in trajectories. In this way, the multi-functional properties of locations can be properly tackled. Furthermore, in order to incorporate temporal information in trajectories into location embeddings, we propose a subtle temporal encoding module and a novel pre-training objective, which further improve the quality of location embeddings. We evaluate our proposed model on two real-world mobile user trajectory datasets. The experimental results demonstrate that, compared with the existing embedding methods, our CTLE model can pre-train higher quality location embeddings and significantly improve the performance of downstream user location prediction models.

Introduction

In recent years, the increasing availability of location-based service (LBS) data, such as GPS trajectories, check-ins at point-of-interests (POIs) and cellular signaling records, have led to a burst of studies focused on mining spatial-temporal data. Among them, user location prediction has received much attention (Liu et al. 2016; Xiao et al. 2017; Kong and Wu 2018), and can be utilized in multiple applications, including modeling users' mobility behaviors (Fu and Lee 2020), or recommending locations (POIs) for users (Zhao et al. 2019). Among these researches, learning embedding vectors for locations is a very fundamental and critical problem, due to the fact that accurate user location prediction requires high quality embedding vectors for better discrimination between locations. Compared to the widely used fully-connected embedding layers in most prediction models, pre-training location embeddings using un-supervised or self-supervised objectives can incorporate more general and comprehensive information of locations. In this way, they can be shared across a wide range of downstream models to improve the computation efficiency, while also promoting the overall prediction performance.



(a) Visited time differences (b) Visited time distributions

Figure 1: Two aspects of temporal information contained in human trajectories. (a) illustrates that in a trajectory, ① visited time difference between the same location can reveal its visited frequency, and ② the difference between two consecutive visiting records can be seen as the former location's stayed duration. (b) displays the variable visited time distributions of locations with different functionalities.

The idea of capturing location characteristics from sequential trajectories is akin to language modelings in natural language processing (NLP). In fact, most existing pre-trained location embedding methods are based on distributed word representations, which are widely used in NLP systems. For example, DeepMove (Zhou and Huang 2018) implements Word2Vec (Mikolov et al. 2013) on trajectory data to extract location function information. However, besides sharing many similar characteristics with sentences, trajectories contain unique temporal information that shouldn't be ignored. As illustrated in Figure 1(a), in a trajectory, the relative visited time differences between locations can reflect their properties such as visited frequency and stayed duration. On the other hand, time will affect peoples' preferences on locations, so the visited time of a location can be an indication of its functionalities, as shown in Figure 1(b). Geo-

*Corresponding author: yflin@bjtu.edu.cn

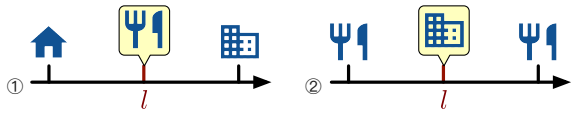


Figure 2: One location may undertake different functionalities in dissimilar contextual neighbors. In trajectory ①, the user visited location l for breakfast after leaving home, and then headed for work; in trajectory ②, the user stayed at the same location l for a business meeting between two meals.

Teaser (Zhao et al. 2017), TALE (Wan et al. 2019) and HIER (Shimizu, Yabe, and Tsubouchi 2020) extract temporal information from absolute visited time of locations on top of modeling sequential correlations. Yet, they don't explicitly take relative visited time differences into account.

On the other hand, multi-functional locations are common in the real world. For instance, a shopping mall may contain restaurants or cinemas, and an office building may contain entertainment places or gyms. People visit the same location for different purposes in dissimilar contextual neighbors, as shown in Figure 2. Incorporating the specific context of a location into its embedding vector can be beneficial for location prediction, for the provided information will be more accurate. Yet, current location embedding methods based on distributed representations assign a single latent vector for each location. This approach can be viewed as a simple mixing of various contextual environments, and is unable to handle variable location functionalities under different contexts.

In order to tackle the above discussed problems lie in existing location embedding models, we aim to build a pre-training model which is able to adaptively generate embedding vectors for locations based on their specific contextual neighbors. In this paper, we propose a novel *Context and Time aware Location Embedding* (**CTLE**) model. Instead of assigning one latent vector as each location's final representation, a target location's embedding vector is calculated by a parameterized mapping function of its contextual locations' encoding vectors. In this way, we are able to incorporate locations' context-specific functionalities into their embedding vectors. We implement the mapping function using bidirectional Transformer encoder (Vaswani et al. 2017), and employ Masked Language Model (MLM) pre-training objective introduced in BERT (Devlin et al. 2019) to model the sequential correlations in user trajectories. Furthermore, in order to incorporate the unique temporal information in trajectories, we design a subtle temporal encoding module to model relative visited time differences between locations. And we also propose a novel Masked Hour (MH) pre-training objective to extract locations' functionalities from their visited time distributions. Our model's effectiveness is verified on two real-world mobile user trajectory datasets and three downstream models for user next location prediction. The experimental results demonstrate that our CTLE model can obviously help downstream models to improve the prediction accuracy on dense trajectories.

The main contributions of this paper are summarized as follows:

- We propose a contextual location embedding model CTLE, which is able to adaptively generate embedding vector for a target location based on its specific context. Therefore, a location's variable functionalities under different contextual environments can be discriminated.
- A subtle temporal encoding module and a novel pre-training objective are proposed to model two aspects of temporal information in trajectories. This will improve our model's understanding of locations' characteristics.
- We employ our CTLE model into three downstream models for user next location prediction, and conduct extensive experiments on two real-world mobile user trajectory datasets. The experimental results show that the prediction performances are significantly improved, which demonstrates our model's superiority.

Related Work

Most user location prediction models are feature based, and require locations being represented by latent embedding vectors. One of the most straight forward and widely applied strategies is to use a fully-connected embedding layer. This layer randomly initializes one latent vector for each location, which is then trained with task-specific objectives (Kong and Wu 2018; Zhao et al. 2019). Yet, the embedding vectors generated in this way are hard to migrate to other models and tasks. Embedding layers also suffer from over-fitting problems, and are unable to incorporate comprehensive information of locations, which will hurt prediction performance.

Pre-training embedding vectors with un-supervised or self-supervised objectives is a common practice in natural language processing (Mikolov et al. 2013; Devlin et al. 2019) and computer vision (Szegedy et al. 2015). In recent years, this topic has also attracted much attention in spatial-temporal data mining. Actually, as a kind of sequential data, user mobility trajectories share many common characteristics with sentences in natural language. Thus, by modeling the co-occurrence of target locations and their contexts in trajectories, we can extract locations' functionality information. Inspired by this idea, most existing location embedding models are based on word embedding models in natural language processing. For example, DeepMove (Zhou and Huang 2018) implements Skip-gram (Mikolov et al. 2013) to model human movements between locations, and (Yao et al. 2018) learns latent representations through an N-gram model (Pauls and Klein 2011). These pre-training models only require un-labeled trajectory data, which are often abundant. The quality of result embedding vectors are validated by incorporating them into downstream tasks, like user location prediction (Zhao et al. 2017) and land usage classification (Zhang et al. 2020).

However, user trajectories also include some unique properties, and temporal correlation is one of them. Locations' functionalities can be more accurately extracted from the temporal information, so it should be considered by embedding models. Geo-teaser (Zhao et al. 2017) discriminates locations which are visited during weekdays and weekends through expanding the output embeddings in Skip-gram. TALE (Wan et al. 2019) incorporates visited time information by designing a novel tree structure for soft-max calculation in CBOW (Mikolov et al. 2013). HIER

(Shimizu, Yabe, and Tsubouchi 2020) combines visited time and stayed duration information by feeding an RNN-based N-gram model with extra embedding vectors. Experimental results show temporal information is indeed helpful for pre-training higher quality location embedding vectors.

It is common that a location is multi-functional, and people may visit the same location for various purposes under dissimilar contextual environments. Correspondingly, it is possible to indicate a location's specific functionality from its context, i.e., nearby visited locations in the trajectory. Yet, existing location embedding methods only assign a single latent vector for each location, which means they are unable to differentiate a location's variable functionalities. Inspired by the recent advances of contextual embedding in natural language processing (Devlin et al. 2019; Yang et al. 2019), we propose a context-aware location embedding model that dynamically calculate latent embeddings for locations based on their contextual neighbors. In this way, the specific functionality of a location is able to be well modeled.

Preliminaries

Definition. ***Spatial-Temporal Trajectory.*** In location-based services, a user's movements during a certain period can be represented by a trajectory s consisting of sequential visiting records. A visiting record (u, l, t) indicates that user u visited location l at time t. We denote the set of user trajectories as S, the set of all locations appear in the dataset as L, and the set of all users as U.

Problem Statement. ***Pre-training Contextual Embedding Vectors for Locations.*** Given a set of user spatial-temporal trajectories S, we aim to pre-train a parameterized mapping function f to generate a contextual embedding vector $\boldsymbol{z}(l)$ for a target location l given its context $C(l)$. The function should be trained in a self-supervised manner with no task-specific objectives needed.

The CTLE Model

Figure 3 illustrates the architecture of our CTLE model. It mainly consists of three components: 1) the encoding layer, which fuses temporal encoding vectors with locations' encoding vectors to extract information from relative visited time differences; 2) the bidirectional Transformer encoder, acting as a mapping function for calculating a target location's embedding, given its specific contextual neighbors; 3) the pre-training objective, which incorporates locations' characteristic information into their embeddings through modeling the co-occurrence probability between target locations and contexts, and the absolute visited time to the targets. The construction of our CTLE model is explained in detail in this section.

Pre-training Bidirectional Transformer for Context-aware Location Embedding

Multi-functional locations are very common in the real world. Usually, a user visits one location with a certain purpose, but if the location is multi-functional, a single visiting record (u, l, t) cannot directly indicate the location's functionality under that specific circumstance. Contextual environment, i.e., visiting records nearby the target location in a trajectory, can more explicitly indicate a certain functionality of the target location, as shown in Figure 2.

Based on the above observation, we propose a context-aware location embedding model, which generates a location's latent representation with consideration of its contextual neighbors. Specifically, given a target location l and its context $C(l)$, we calculate its embedding vector $\boldsymbol{z}(l)$ via a parameterized mapping function f, denoted as the following abstract form:

$$\boldsymbol{z}(l) = f(l, C(l)). \tag{1}$$

In this way, locations' embedding vectors are dependent on their contextual environments, which makes it possible to discriminate a location's specific functionality. To implement function f, we utilize a bidirectional Transformer encoder (Vaswani et al. 2017), for two reasons. First, locations in trajectories are related with contexts from both left and right sides. Bidirectional Transformers are able to capture contextual information from both sides, while also consider their interactions. Second, Transformer architecture has been proven by many studies (Dai et al. 2019) to have higher expressive power than traditional sequential models like LSTM (Hochreiter and Schmidhuber 1997).

Now we introduce the calculation of contextual embeddings in detail. Given a trajectory $s = \{(u, l_1, t_1), (u, l_2, t_2), \ldots, (u, l_n, t_n)\}$, we fetch an input latent vector $\boldsymbol{z}'(l)$ for each location l from the location encoding module, denoted as:

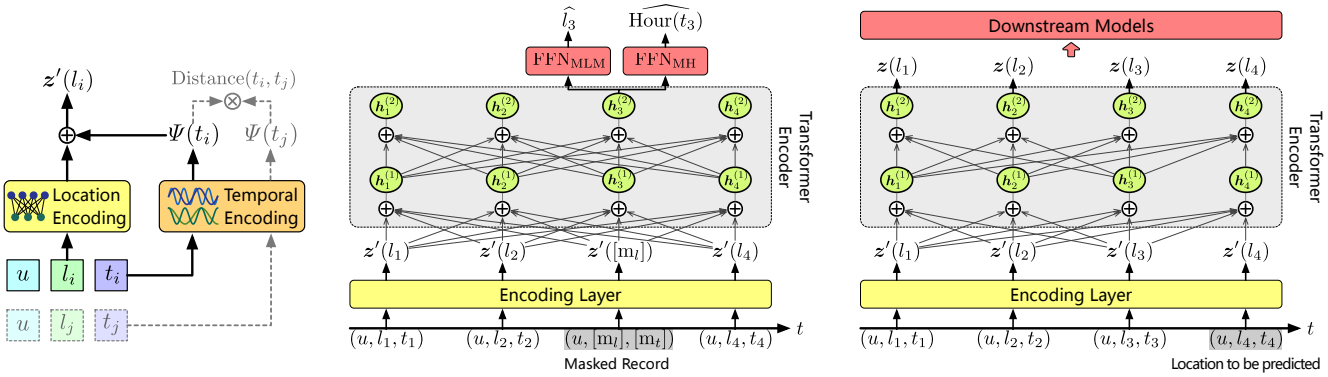
$$\boldsymbol{z}'(l) = \Omega(l), \tag{2}$$

thus forming an input sequence $\{\boldsymbol{z}'(l_1), \boldsymbol{z}'(l_2), \ldots, \boldsymbol{z}'(l_n)\}$. Ω denotes the location encoding module, which is implemented using a fully-connected embedding layer. The input sequence is then fed into the stack of Transformer encoder layers, where each layer is composed of a multi-head self-attention module and a position-wise fully connected feed-forward network. Formally, we have:

$$\begin{aligned} &\{\boldsymbol{h}_1^{(k)}, \boldsymbol{h}_2^{(k)}, \ldots, \boldsymbol{h}_n^{(k)}\} \\ =&\mathrm{TransEncLayer}(\{\boldsymbol{h}_1^{(k-1)}, \boldsymbol{h}_2^{(k-1)}, \ldots, \boldsymbol{h}_n^{(k-1)}\}), \\ &\{\boldsymbol{h}_1^{(0)}, \boldsymbol{h}_2^{(0)}, \ldots, \boldsymbol{h}_n^{(0)}\} = \{\boldsymbol{z}'(l_1), \boldsymbol{z}'(l_2), \ldots, \boldsymbol{z}'(l_n)\}, \end{aligned} \tag{3}$$

where $\mathrm{TransEncLayer}$ represents a Transformer encoder layer. $\{\boldsymbol{h}_1^{(k)}, \boldsymbol{h}_2^{(k)}, \ldots, \boldsymbol{h}_n^{(k)}\}$ is the output memory sequence of the k-th layer, and the input sequence of the $(k+1)$-th layer. Suppose the model contains a total of N Transformer encoder layers. We regard the i-th item of the last layer's output memory sequence, $\boldsymbol{h}_i^{(N)}$, as the contextual embedding vector of location l_i in trajectory s. If we denote all locations in s except l_i as its context $C(l_i)$, the calculation equation of $\boldsymbol{z}(l_i)$ can be written as:

$$\begin{aligned} \boldsymbol{z}(l_i) = \boldsymbol{h}_i^{(N)} &= \mathrm{TransEnc}(\{\boldsymbol{z}'(l_1), \boldsymbol{z}'(l_2), \ldots, \boldsymbol{z}'(l_n)\})_i \\ &= \mathrm{TransEnc}(\Omega(\{l_1, l_2, \ldots, l_n\}))_i \\ &= f(\{l_1, l_2, \ldots, l_n\})_i \\ &= f(l_i, C(l_i))_i, \end{aligned} \tag{4}$$



(a) Encoding layer (b) Pre-training (c) Calculating contextual embeddings

Figure 3: The model architecture of CTLE. (a) illustrates the encoding layer of CTLE. (b) displays the pre-training process of CTLE, where two fully-connected feed-forward networks aims to predict a masked record's original location and visited hour index, respectively. (c) represents utilizing CTLE to calculate contextual embeddings for user next location prediction. The full trajectory is used as input, and a special attention mask is applied to prevent information leaking. Note that both (b) and (c) just demonstrate a Transformer encoder with two layers. In reality, the number of layers is adjustable.

which corresponds to the abstract form presented in Equation (1). TransEnc represents the stack of Transformer encoder layers.

The mapping function f should learn to understand the correlation between target locations and their corresponding contexts, so that the locations' characteristic information can be incorporated into their embedding vectors. Inspired by recent advances of language understanding in NLP models, we implement the Masked Language Model (MLM) proposed in BERT (Devlin et al. 2019) to construct a self-supervised training objective. Given a trajectory s, we randomly select 20% records as masked records, and replace their corresponding locations and visited time with special tokens $[\text{m}_l]$ and $[\text{m}_t]$. The result trajectory $\widetilde{s}$ is regarded as the input of the function f. For a masked location l_m, we aim to predict its original token using the output of f:

$$\widehat{l_m} = \text{FFN}_{\text{MLM}}(\boldsymbol{h}_m^{(N)}) = \text{FFN}_{\text{MLM}}(f(\widetilde{s})_m), \tag{5}$$

where FFN_{MLM} is a fully-connected feed-forward network for casting Transformer encoder's output memory vectors into the prediction of location tokens. MLM objective can then be constructed through maximizing the prediction accuracy. If we regard the masked locations as targets, and the un-masked locations as their contexts, the pre-training objective of MLM can be described as:

$$\begin{aligned} O_{\text{MLM}} &= \arg\max_{\theta} \textstyle\sum_{l_m \in \Gamma} p(l_m | \text{FFN}_{\text{MLM}}(f(\widetilde{s})_m)) \\ &= \arg\max_{\theta} \textstyle\sum_{l_m \in \Gamma} p(l_m | \text{FFN}_{\text{MLM}}(f(C(l_m))_m), \end{aligned} \tag{6}$$

where Γ denotes the set of all masked records in the training set, θ denotes the set of all learnable parameters of the embedding model. By maximizing the co-occurrence probability of targets and their contexts, locations with similar contextual environment will have closer embedding vectors, thus the functionality information is incorporated. This idea is akin to that in Word2Vec (Mikolov et al. 2013). Yet, instead of directly optimizing a set of static embedding vectors, our model optimizes the parameters of a mapping function. In this way, the result embedding vectors will be context-dependent, and are able to incorporate locations' specific characteristic information in different contextual neighbors.

Incorporating Temporal Information

Temporal information is essential for spatial-temporal data mining. For trajectory data, temporal information is mainly carried by the visited time of locations, and can be viewed from two aspects: relative visited time difference and absolute visited time. Locations' visited frequencies or stayed duration can be revealed from relative visited time differences, which implies functionality information, as illustrated by Figure 1(a). On the other hand, human activities are usually time-regulated, and the functionalities a location carried can determine the time users arrive at it, as shown in Figure 1(b). Correspondingly, we can extract characteristic information from locations' absolute visited time.

Based on the above analysis, we incorporate two aspects of temporal information into our model via a subtle temporal encoding module and a novel pre-training objective.

Temporal Encoding in Encoding Layer In order to explicitly model the relative visited time differences between locations in trajectories, we propose the temporal encoding module. In the original Transformer (Vaswani et al. 2017), positional encoding is introduced for the model to discriminate the order of tokens in the input sequence. Its calculation equation can be written as:

$$\begin{aligned} \Phi(o) &= [\cos(\omega_1 o), \sin(\omega_1 o), \ldots, \cos(\omega_d o), \sin(\omega_d o)], \\ \omega_k &= 1/10000^{2k/d}, \end{aligned} \tag{7}$$

where function Φ represents the positional encoding module, o is the index of a position, and $2d$ is the dimension of positional encoding vector.

Positional encoding can only represent simple sequential orders. Yet, for trajectories, the visiting records unevenly ar-

range on the temporal axis, and the relative visited time differences between locations contain important information, like locations' visited frequencies or stayed durations, as shown in Figure 1(a). Inspired by (Xu et al. 2020), our proposed temporal encoding tackles this problem by making two simple adjustments to the positional encoding: replacing the index of position o with an absolute visited timestamp t, and setting the parameters $\{\omega_1, \omega_2, \ldots, \omega_d\}$ to be trainable. Formally, we have:

$$\Psi(t) = [\cos(\omega_1 t), \sin(\omega_1 t), \ldots, \cos(\omega_d t), \sin(\omega_d t)], \quad (8)$$

where Ψ is our temporal encoding module. The dot-product of $\Psi(t)$ and $\Psi(t+\delta)$ are calculated as:

$$\Psi(t) \cdot \Psi(t+\delta) = \cos(\omega_1 \delta) + \cos(\omega_2 \delta) + \cdots + \cos(\omega_d \delta), \quad (9)$$

which means the distance (measured by dot-product) between $\Psi(t)$ and $\Psi(t+\delta)$ is learnable, and is independent of t. There are no lack of dot-product operations in a Transformer encoder, so this distance information is destined to be captured by the model.

Then, we introduce the temporal encoding into our model by combing it with the location encoding. Specifically, we modify Equation 2 for fetching the input latent vector of record (u, l, t) as follows:

$$\boldsymbol{z}'(l) = \Omega(l) + \Psi(t). \quad (10)$$

In this way, our model is able to incorporate the temporal information embedded in relative visited time differences into the pre-training and embedding generation process, which results in higher quality embedding vectors. The overall architecture of CTLE's encoding layer combined with temporal encoding is shown in Figure 3(a).

Masked Hour Objective in Pre-training In order to further extract information from the absolute visited time, we propose the Masked Hour (MH) pre-training objective. For a masked visited time t_m in a masked trajectory $\widetilde{s}$, we aim to predict its original visited hour index:

$$\widehat{\mathrm{Hour}(t_m)} = \mathrm{FFN}_{\mathrm{MH}}(f(\widetilde{s})_m), \quad (11)$$

where $\mathrm{Hour}(t)$ denotes the hour index of time t. $\mathrm{FFN}_{\mathrm{MH}}$ is a fully-connected network for casting the Transformer encoder's output memory vector into the prediction of hour index. During pre-training, the MH is combined with MLM to form a multi-task learning objective. The MH objective and the overall pre-training objective can be described as:

$$O_{\mathrm{MH}} = \arg\max_{\theta} \textstyle\sum_{t_m \in \Gamma} p(\mathrm{Hour}(t_m) | \mathrm{FFN}_{\mathrm{MH}}(f(\widetilde{s})_m)), \quad (12)$$

$$O = O_{\mathrm{MLM}} + O_{\mathrm{MH}}, \quad (13)$$

where θ denotes the set of all learnable parameters of embedding model. We can give an abstract form of the above pre-training objective as $\arg\max_\theta p(l_m, t_m | C(l_m))$. By maximizing the probability that a target location will be visited during a certain period given its contexts, we incorporate the information implied in absolute visited time into the model. This enables the model's ability to extract locations' functionalities from temporal information, thus improving the quality of generated embedding vectors. The overall model architecture of CTLE during the pre-training phase is shown in Figure 3(b).

Scope of Application

To make the contextual embedding idea of CTLE effective, the trajectories need to be relatively dense, so that target locations have adequate strength of correlations with their contextual environments. In other words, the contexts need to be complete enough to reveal the targets' functionalities. That is what sparse trajectories such as most of check-in records lack. Besides, locations in check-in data are often manually divided and classified into POIs with certain purposes, which means their multi-functional property is not obvious. In summary, when applied on very sparse trajectories like check-in data, our CTLE will lose its superiority over the existing distributed embedding-based methods.

Experiments

In order to evaluate the quality of contextual embedding vectors generated by our model, we incorporate these vectors into three downstream user location prediction models, and compare the results with other location embedding methods.

Datasets

The experiments are carried out on two real-world datasets which contain mobile signaling data in Beijing and Shenyang in China, denoted as Mobile-PEK and Mobile-SHE, respectively. They record the switching events between telecommunication base stations of anonymous mobile users. Each base station is responsible for providing signal to its surrounding area, and users' entering to the area will be recorded by the station. In this paper, we treat base stations as locations. Unlike some public available check-in datasets, mobile signaling datasets contain dense trajectories, making them more proper for evaluating the effectiveness of CTLE.

Technically, signaling data can be representative of a user's full moving trajectories. Yet, not all trajectory points reflect the user's visiting to locations. For example, people's traveling from one place to another will also generate a sequence of trajectory points, but these points lack semantic information. In order to filter out such points that users just passed by, we firstly remove the base stations and users whose related records have an average duration time below 30 minutes, for most signaling records generated by them are just passing-points. Then we remove all the trajectory points with a duration time below 5 minutes. After this processing, all remaining signaling records can be seen as staying points. A user's staying point sequence within one day sorted by ascending timestamp is regarded as a trajectory. Finally, we include the trajectories with more than 5 staying points and users with more than 4 trajectories in our datasets. The statistics of datasets are shown in Table 1.

Dataset	#Users	#Locations	#Records	Time span
Mobile-PEK	12,691	7,279	1,383,422	5 days
Mobile-SHE	10,564	7,201	607,581	11 days

Table 1: Statistics of datasets.

Baseline Location Embedding Methods

To prove the superiority of our model, we include two classic distributed embedding models which are implemented by many location embedding methods, and also some state-of-the-art location embedding models for comparison.

- **FC Layer**: a widely used strategy for latent representation, which allocates a randomly initialized embedding vector for each location. These vectors are then trained together with other parameters of the whole models.
- **Skip-gram** (Mikolov et al. 2013): a variant of the Word2Vec model, and has been utilized in (Liu, Liu, and Li 2016) for modeling mobility trajectories.
- **POI2Vec** (Feng et al. 2017): an embedding method based on Word2Vec, which models spatial correlations through distributing locations into a geographical binary tree.
- **Geo-Teaser** (Zhao et al. 2017): Geo-Temporal Sequential Embedding Rank fuses temporal and spatial information through expanding the output vector, and modifying Skip-gram's negative sampling strategy.
- **TALE** (Wan et al. 2019): Time-Aware Location Embedding incorporates temporal information through designing a novel temporal tree structure for hierarchical softmax calculation.
- **HIER** (Shimizu, Yabe, and Tsubouchi 2020): Hierarchical Fine Grained Place Embedding utilizes an N-gram architecture (Pauls and Klein 2011) based on LSTM (Hochreiter and Schmidhuber 1997).

User Next Location Prediction Models

Downstream prediction models can be utilized to evaluate the effectiveness of embedding methods, for the prediction performance relies on the quality of location embedding vectors. In this paper, we employ three popular user next location prediction methods as the downstream models:

- **ST-RNN** (Liu et al. 2016): A Spatial-Temporal Recurrent Neural Network model for user next location prediction. It incorporates spatial and temporal correlations via time-specific and distance-specific transition matrices in recurrent propagation.
- **ERPP** (Xiao et al. 2017): Event Recurrent Point Process regards temporal event sequences as point processes. In this paper, we consider a visit to a location as an event, and the arrival time as its timestamp.
- **ST-LSTM** (Kong and Wu 2018): A Spatial-Temporal Long-Short Term Memory model for user next location prediction. It expands the input of LSTM by embedding time and distance values into latent vectors.

Give a sequential trajectory $s = \{(u, l_1, t_1), (u, l_2, t_2), \ldots, (u, l_n, t_n)\}$, we feed the prediction models with a sequence of corresponding embedding vectors $\{\boldsymbol{z}(l_1), \boldsymbol{z}(l_2), \ldots, \boldsymbol{z}(l_i)\}$. It's worth noting that when applying CTLE to the location prediction models, we use a special attention mask to prevent the information of the locations to be predicted from leaking to the historical sequence, as illustrated in Figure 3(c). All embedding methods except FC Layer do not participate in backward propagation during location prediction.

Settings

For the Mobile-PEK dataset, we choose trajectories in the first 3 days for training, the 4th day for evaluation and the last day for test. For the Mobile-SHE dataset, we choose trajectories in the first 7 days for training, the last 2 days for test and the other days for evaluation. In the location prediction task, we select the last three points in every trajectory as prediction targets, and the rest of points as source sequence.

Both pre-trained location embedding models and downstream prediction models are trained with the training sets, and validated on the evaluation sets to get the optimum hyperparameters. Note that the pre-training embedding models themselves don't have a reliable metric for the performance evaluation, so we adjust the hyperparameters with the help of the ST-LSTM model. The location prediction models are trained with Cross Entropy loss, and evaluated with Accuracy, macro-Recall and macro-F1 score.

We implement all baseline models and our CTLE model in PyTorch (Paszke et al. 2019). All embedding models' dimensions of result embedding vectors are set to 128. We implement ERPP and ST-LSTM based on 2-layer LSTM networks with a hidden size of 512. The unique spatial-temporal context structure of ST-RNN prevents us from implementing it in a multi-layer manner, so it is based on a single-layer RNN with a hidden size of 512. For our CTLE model, we stack 4 layers of Transformer encoders, with 8 attention heads, and the hidden sizes of feed-forward layers are set to 512. The CTLE is pre-trained on the training sets for 200 epochs, and all prediction models are trained with the early-stopping mechanism to obtain the best-performing epochs on the evaluation sets. We choose Adam optimizer and an initial learning rate of 0.0001 across the board. All experiments were ran on Intel Xeon Silver 4210 CPUs, and NVIDIA RTX2080ti GPUs.

Experimental Results

Table 2 shows the performance comparison of different models for user next location prediction. All baseline embedding methods are fused with multiple downstream prediction models to calculate the results. Our CTLE consistently outperforms other location embedding methods except on the ST-RNN prediction model, Mobile-SHE dataset.

Randomly initialized fully-connected embedding layers only fit on the specific prediction objectives, and are unable to incorporate general information, like the functionalities of locations. Skip-gram only utilizes the co-occurrence probabilities of target locations and their contexts, ignoring some unique properties of spatial-temporal trajectories, like the visited time and geographical positions of locations. As we can see from Table 2, these methods generally result in worse prediction performance than the newer, more comprehensive location embedding methods.

Geo-Teaser and POI2Vec introduce spatial information into the embeddings through different approaches. Yet, geographical correlation between locations in a city is not very evident. On the one hand, people's visiting preferences are not confined within small areas in the city; on the other hand, locations' properties don't have a strong correlation with their geographical positions, as close-up locations might

Prediction Model		ST-RNN			ERPP			ST-LSTM		
Metric		Accuracy (%)	macro-Recall (%)	macro-F1 (%)	Accuracy (%)	macro-Recall (%)	macro-F1 (%)	Accuracy (%)	macro-Recall (%)	macro-F1 (%)
Dataset	Embedding Method									
Mobile-PEK	FC Layer *	3.744±0.10	1.739±0.06	1.449±0.19	4.373±0.14	2.017±0.04	1.595±0.05	4.542±0.15	2.092±0.09	1.689±0.07
	Skip-gram	3.671±0.11	1.777±0.11	1.423±0.05	4.611±0.01	2.368±0.07	1.779±0.04	4.877±0.05	2.586±0.06	1.947±0.04
	POI2Vec	3.992±0.08	2.281±0.08	1.838±0.06	5.024±0.08	2.595±0.07	2.035±0.06	5.163±0.10	2.682±0.08	2.077±0.10
	Geo-Teaser	3.998±0.13	2.166±0.07	1.796±0.07	5.159±0.05	2.671±0.08	2.039±0.03	5.305±0.05	2.739±0.04	2.084±0.02
	TALE	4.199±0.05	2.240±0.07	1.815±0.06	5.457±0.03	3.237±0.07	2.587±0.03	5.511±0.05	3.152±0.10	2.486±0.13
	HIER	4.339±0.04	2.440±0.07	1.862±0.08	5.607±0.09	2.870±0.08	2.176±0.04	5.589±0.15	2.839±0.11	2.165±0.02
	CTLE (ours)	**5.068±0.05**	**2.890±0.11**	**2.312±0.02**	**6.481±0.05**	**4.002±0.04**	**3.066±0.06**	**6.473±0.09**	**4.072±0.13**	**3.097±0.13**
Mobile-SHE	FC Layer *	3.674±0.07	2.408±0.07	1.946±0.05	4.343±0.18	2.454±0.10	2.037±0.09	4.416±0.20	2.450±0.14	2.005±0.11
	Skip-gram	3.646±0.05	2.278±0.08	1.809±0.05	4.405±0.06	2.459±0.06	1.974±0.05	4.508±0.05	2.507±0.07	1.998±0.07
	POI2Vec	3.936±0.04	2.605±0.04	2.084±0.03	4.923±0.06	2.992±0.02	2.408±0.02	4.930±0.07	2.890±0.10	2.305±0.08
	Geo-Teaser	4.006±0.05	2.455±0.03	1.897±0.02	4.932±0.12	2.895±0.02	2.410±0.07	5.130±0.15	2.754±0.07	2.245±0.06
	TALE	4.689±0.10	**3.444±0.09**	**2.761±0.08**	5.179±0.09	3.446±0.06	2.883±0.04	5.204±0.06	3.399±0.11	2.787±0.11
	HIER	4.539±0.22	3.117±0.15	2.521±0.09	5.624±0.16	3.273±0.17	2.708±0.18	5.672±0.09	3.252±0.07	2.680±0.05
	CTLE (ours)	**5.124±0.20**	3.392±0.11	2.720±0.07	**6.311±0.04**	**3.984±0.05**	**3.340±0.07**	**6.325±0.08**	**3.950±0.11**	**3.291±0.06**

* The FC Layer is the originally used location embedding method in the downstream prediction models, so the results in the corresponding rows are the original prediction performances of these models.

Table 2: User next location prediction performance comparison of different approaches.

also have diverse functionalities.

Geo-Teaser also incorporates temporal information into location embeddings, by taking absolute visited time of locations into consideration. Yet, it only discriminates visiting records happened during weekdays and weekends, ignoring finer-grained temporal information, like visiting hours. TALE and HIER both utilize visiting hour indices, which can imply users' preferences to locations. Thus, they can generate location embeddings that are more beneficial for prediction tasks. Yet, none of the aforementioned location embedding methods considers relative visited time differences, which are capable of indicating relationship between locations. More importantly, they assign only one latent vector to represent each location, which makes them unable to reflect the variable functionality property of locations in resulting embedding vectors. This can seriously hurt the prediction performance of downstream models. Unlike check-in datasets, locations in mobile signaling data are not manually divided and classified into POIs with certain purposes, so their multi-functional property is extra obvious.

Our CTLE model dynamically generates locations' embeddings with regards to their specific context, so that the exact functionality of a location in a certain contextual environment can be discriminated. In addition, CTLE takes both relative visited time differences between locations and fine-grained absolute visited time into consideration. As demonstrated in Table 2, these designs result in higher quality location embeddings, and can help downstream location prediction models to achieve better performance.

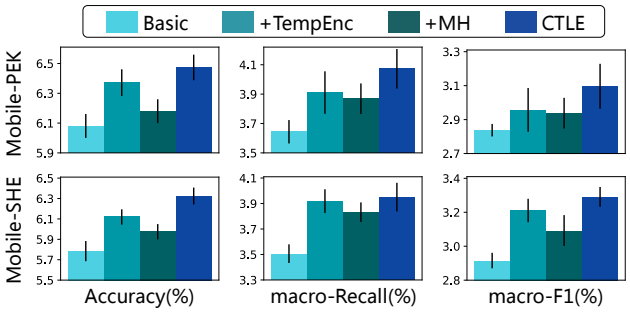


Figure 4: Component analysis of CTLE.

Component Analysis

To further investigate the effectiveness of each component of CTLE, we design three variants of the CTLE model, as listed below:

- Basic: This model uses the original positional encoding from Transformer (Vaswani et al. 2017), and only utilizes the MLM objective for pre-training.
- +TempEnc: This model replaces the positional encoding in the basic model with our proposed temporal encoding.
- +MH: This model still uses positional encoding, but combines the Masked Hour objective with the MLM objective during pre-training.

We compare these three variants with the CTLE model on the ST-LSTM downstream model. Figure 4 shows the results. The basic model already have some performance advantages compared to the location embedding methods that are based on distributed word embeddings. This indicates that learning context-aware location embedding vectors can be beneficial for the location prediction task.

Temporal encoding and the Masked Hour objective incorporates temporal information embedded in trajectories into the model. As illustrated in Figure 4, they can both improve the prediction performance over the basic model. Because these two modules focus on different aspects of temporal influence, our final model combines them and gain the optimum results.

Conclusion

In this paper, we propose a novel context and time aware location embedding method CTLE. It dynamically generates embedding vectors of target locations given their contexts, so the specific functionality of a location under certain contextual environment can be incorporated. A subtle temporal encoding module and a novel Masked Hour pre-training objective further incorporate multiple aspects of temporal information in trajectories into the embeddings. Experiments are conducted on the user next location prediction downstream task on two real-world mobile user trajectory datasets and three popular downstream models, and the results prove the effectiveness of our model for learning accurate and comprehensive location embeddings on dense trajectories.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities (Grant No. 2019JBM024).

References

Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J. G.; Le, Q. V.; and Salakhutdinov, R. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *57th Conference of the Association for Computational Linguistics*, 2978–2988.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 4171–4186.

Feng, S.; Cong, G.; An, B.; and Chee, Y. M. 2017. POI2Vec: Geographical Latent Representation for Predicting Future Visitors. In *31th AAAI Conference on Artificial Intelligence*, 102–108.

Fu, T.-Y.; and Lee, W.-C. 2020. Trembr: Exploring Road Networks for Trajectory Representation Learning. *ACM Transactions on Intelligent Systems and Technology* 11(1): 1–25.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8): 1735–1780.

Kong, D.; and Wu, F. 2018. HST-LSTM: A Hierarchical Spatial-Temporal Long-Short Term Memory Network for Location Prediction. In *27th International Joint Conference on Artificial Intelligence*, volume 18, 2341–2347.

Liu, Q.; Wu, S.; Wang, L.; and Tan, T. 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. In *30th AAAI conference on artificial intelligence*, 194–200.

Liu, X.; Liu, Y.; and Li, X. 2016. Exploring the context of locations for personalized location recommendations. In *25th International Joint Conference on Artificial Intelligence*, 1188–1194.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representation*.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *32nd Advances in Neural Information Processing Systems*, 8024–8035.

Pauls, A.; and Klein, D. 2011. Faster and Smaller N-Gram Language Models. In *49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 258–267.

Shimizu, T.; Yabe, T.; and Tsubouchi, K. 2020. Learning Fine Grained Place Embeddings with Spatial Hierarchy from Human Mobility Trajectories. *arXiv preprint arXiv:2002.02058* .

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *25th IEEE Conference on Computer Vision and Pattern Recognition*, 1–9.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *30th Advances in Neural Information Processing systems*, 5998–6008.

Wan, H.; Li, F.; Guo, S.; Cao, Z.; and Lin, Y. 2019. Learning Time-Aware Distributed Representations of Locations from Spatio-Temporal Trajectories. In *24th International Conference on Database Systems for Advanced Applications*, 268–272.

Xiao, S.; Yan, J.; Yang, X.; Zha, H.; and Chu, S. M. 2017. Modeling the Intensity Function of Point Process Via Recurrent Neural Networks. In *31st AAAI Conference on Artificial Intelligence*, 1597–1603.

Xu, D.; Ruan, C.; Körpeoglu, E.; Kumar, S.; and Achan, K. 2020. Inductive representation learning on temporal graphs. In *8th International Conference on Learning Representations*.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J. G.; Salakhutdinov, R.; and Le, Q. V. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *32nd Advances in Neural Information Processing Systems*, 5754–5764.

Yao, Z.; Fu, Y.; Liu, B.; Hu, W.; and Xiong, H. 2018. Representing Urban Functions through Zone Embedding with Human Mobility Patterns. In *27th International Joint Conference on Artificial Intelligence*, 3919–3925.

Zhang, M.; Li, T.; Li, Y.; and Hui, P. 2020. Multi-View Joint Graph Representation Learning for Urban Region Embedding. In *29th International Joint Conference on Artificial Intelligence.*

Zhao, P.; Zhu, H.; Liu, Y.; Xu, J.; Li, Z.; Zhuang, F.; Sheng, V. S.; and Zhou, X. 2019. Where to go next: A spatio-temporal gated network for next POI recommendation. In *33rd AAAI Conference on Artificial Intelligence*, volume 33, 5877–5884.

Zhao, S.; Zhao, T.; King, I.; and Lyu, M. R. 2017. Geo-teaser: Geo-temporal sequential embedding rank for point-of-interest recommendation. In *26th International Conference on World Wide Web Companion*, 153–162.

Zhou, Y.; and Huang, Y. 2018. Deepmove: Learning place representations through large scale movement data. In *6th IEEE International Conference on Big Data*, 2403–2412.