STA-TCN: Spatial-temporal Attention over Temporal Convolutional Network for Next Point-of-interest Recommendation

JUNJIE OU, HAIMING JIN, XIAOCHENG WANG, HAO JIANG, and XINBING WANG,
Shanghai Jiao Tong University, China
CHENGHU ZHOU, Chinese Academy of Sciences, China

Recent years have witnessed a vastly increasing popularity of location-based social networks (LBSNs), which facilitates studies on the *next Point-of-Interest (POI) recommendation* problem. A user's POI visiting behavior shows the *sequential transition* correlation with previous successive check-ins and the *global spatial-temporal* correlation with those check-ins that happened a long time ago at a similar time of day and in geographically close areas. Although previous POI recommendation methods attempted to capture these two correlations, several limitations remain to be solved: (1) RNNs are widely adopted to capture the sequential transition correlation, whereas training an RNN is rather time-consuming given the long input check-in sequence. (2) The pairwise *proximities* on time of day and geographical area of check-ins are crucial for global spatial-temporal correlation learning, but have not been comprehensively considered by previous methods. To tackle these issues, we propose a novel next POI recommendation framework named STA-TCN. Specifically, instead of RNNs, STA-TCN augments the Temporal Convolutional Network with gated input injection to learn sequential transition correlation. Furthermore, STA-TCN fuses two novel *grid-difference* and *time-sensitivity* learning mechanisms with attention network to learn the pairwise spatial-temporal proximities among a user's check-ins. Extensive experiments are conducted on two large-scale real-world LBSN datasets, and the results show that STA-TCN outperforms the best state-of-the-art baseline with an average improvement of 9.71% and 7.88% on hit rate and normalized discounted cumulative gain, respectively.

CCS Concepts: • **Information systems** → **Data mining**; **Spatial-temporal systems**; **Recommender systems;**

Additional Key Words and Phrases: Next POI recommendation, TCN, self-attention, LBSN

ACM Reference format:
Junjie Ou, Haiming Jin, Xiaocheng Wang, Hao Jiang, Xinbing Wang, and Chenghu Zhou. 2023. STA-TCN: Spatial-temporal Attention over Temporal Convolutional Network for Next Point-of-interest Recommendation. *ACM Trans. Knowl. Discov. Data.* 17, 9, Article 124 (June 2023), 19 pages.
https://doi.org/10.1145/3596497

This work was supported in part by NSF China under Grant Nos. 42050105, U20A20181, U21A20519.
Authors' addresses: J. Ou, H. Jin (corresponding author), X. Wang, H. Jiang, and X. Wang (corresponding author), Shanghai Jiao Tong University, No. 800 Dongchuan Road, Minhang District, Shanghai, Shanghai 200240, China; emails: {j_michael, jinhaiming, curryjam_cg, jianghao091, xwang8}@sjtu.edu.cn; C. Zhou, 52 Sanlihe Rd., Xicheng District, Beijing, China Postcode: 100864; email: zhouch@lreis.ac.cn.

1556-4681/2023/06-ART124 $15.00
https://doi.org/10.1145/3596497

1 INTRODUCTION

Nowadays, with the rapid development of ***location-based social networks (LBSNs)*** such as Foursquare and Yelp, a vastly increasing number of users prefer to share with friends their ***Point-of-Interest (POI)*** check-in records at restaurants, museums, and so on, via LBSNs. Such huge amount of user check-in data facilitates researches on learning the users' preferences for POI recommendation. Clearly, accurately recommending POIs to users is of high value both for POI owners to attract potential users, and for users to explore their surroundings and discover potential places of interest to visit. Therefore, in this article, we study the problem of *next POI recommendation* with the check-in data collected from LBSNs.

In practice, a user's POI visiting behavior shows strong *sequential transition correlation*. That is, a user's next visited POI highly correlates with the previous few POIs visited successively by the user. For example, as shown in Figure 1, after having dinner in a restaurant during weekends, it is highly possible that some user will subsequently visit a movie theater, a bar, or some other recreational facility for entertainment. Naturally, capturing such sequential transition correlation among successive check-ins is essential for accurate POI recommendation.

Recently, RNN-based models [13, 21, 38, 40, 41] have been proposed to learn the sequential transition correlation of a user's POI check-ins. To train such RNN-based models, a user's historical check-in sequence has to be divided into multiple short subsequences and fed into these models one after another, which is inevitably rather time-consuming. To improve the training efficiency, we seek to develop an RNN-independent model, which takes as inputs users' entire historical check-in sequences that typically contain hundreds of check-in records for learning the sequential correlation. We innovatively introduce the ***Temporal Convolutional Network (TCN)*** [1] to handle the long check-in sequence for learning the sequential transition correlation. More concretely, the TCN adopts the dilated 1D causal convolution whose parallel convolution framework combined with its exponentially enlarging receptive fields enables much faster training than RNN. In addition, for learning better representations of a user's preference, we augment the TCN by adding to it a gated input injection mechanism, which selectively fuses the original input sequence with the convolutional output results.

Though promising, TCN could not unleash its full power unless the *global spatial-temporal proximities* of a user's POI visiting behavior are appropriately considered. As shown in Figure 1, in the past several days, the user visited the other three bars nearby in midnight as well. Such phenomenon indicates that a user's decision for visiting the next POI could also correlate with those check-ins that happened a long time ago at similar a time of day and in geographically close areas. Thus, properly dealing with such global spatial-temporal correlation then becomes another critical factor for POI recommendation.

Thus far, learning such global correlation has been taken into account by LSTPM [31]. However, LSTPM treats a user's check-ins within an entire day as a trajectory and only performs global correlation learning at such coarse-grained trajectory level. Thus, we propose to empower TCN with the ability to learn the global spatial-temporal proximity correlation by integrating it with a novel *spatial-temporal attention* module. More specifically, we propose two *grid-difference* and *time-sensitivity* learning mechanisms to learn, respectively, the pairwise spatial and temporal proximity scores of a user's check-in history on a fine-grained check-in level. Then, the learned proximity scores are incorporated into the self-attention network to capture the global spatial-temporal correlations.

Collectively, we aggregate our neural network structures for learning the sequential transition correlation and global spatial-temporal correlation into an integrated framework, named as ***Spatial-Temporal Attention over Temporal Convolutional Network (STA-TCN)***. That is,

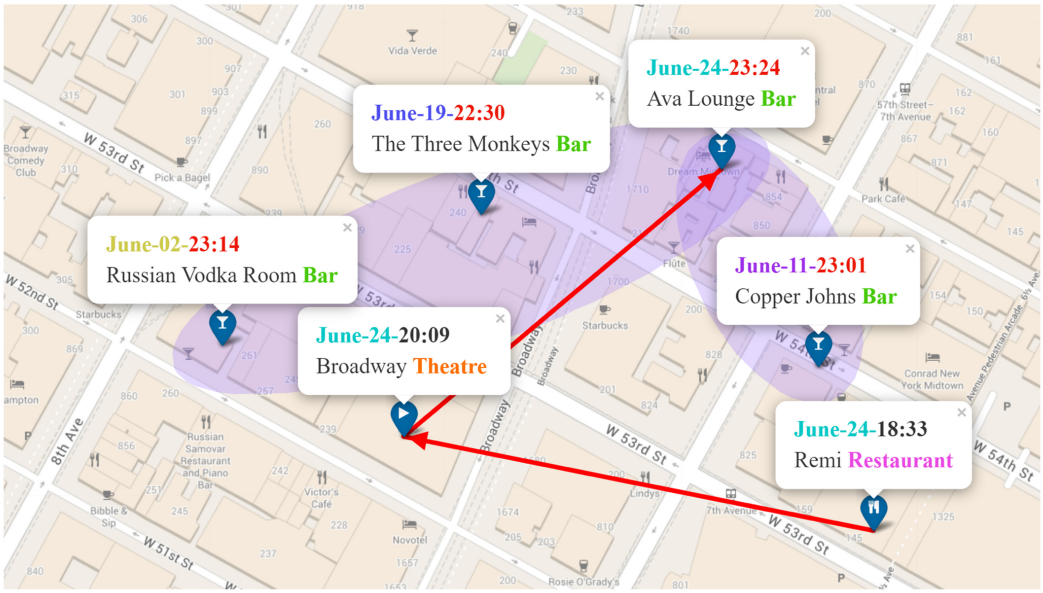


Fig. 1. An example showing the sequential transition correlation and global spatial-temporal correlations of user visiting behaviors, where the red solid lines denote the user's three successive check-ins where the final destination is highly correlated with the previous two, and the blue circle includes the final destination and another several POIs that were visited by user long time apart but performs high proximity.

STA-TCN employs a TCN module with gated injection to capture the sequential transition correlation, which is followed by a spatial-temporal attention module to further capture the global spatial-temporal correlation of a user's check-in history. To summarize, this article makes the following contributions:

- We propose STA-TCN, a novel next POI recommendation framework, to jointly capture the sequential transition and global spatial-temporal correlations of a user's check-in history.
- Instead of RNNs, we innovatively augment temporal convolutional architecture with a gated input injection mechanism to improve the model efficiency of sequential learning for a user's preference.
- We carefully design two novel grid-difference and time-sensitivity learning mechanisms, so the pairwise spatial and temporal proximities among user's check-ins can be comprehensively learned.
- Extensive experiments results on two real-world LBSN datasets show that our proposed STA-TCN outperforms the existing state-of-the-art baseline methods with an average improvement of 9.71% and 7.88% on hit rate and normalized discounted cumulative gain, respectively.

2 RELATED WORK

Recommender system aims to predict users' interests and recommend items that quite likely are interesting for them [10]. Next, POI recommendation aims at recommending users places to visit, which has become an area of increasing research and developing interest within recommender system in recent years [30]. In this section, we will review a series of representative works in next POI recommendation.

Researchers start the next POI recommendation study from learning the user's POI visiting behavior with traditional methods. Matrix factorization-based approaches [2, 14, 26] take the historical user-POI visiting matrix as input and exploit the factorization to get the user-POI preference score. Reference [35] proposes a collaborative filtering method based on non-negative tensor factorization to further model the users' social relation feature. STSCR [6] models the the impact of time and social influence into a unified tensor factorization framework for location recommendation. However, these methods usually perform less desirable for POI recommendation, as they assume the static user preferences and overlook the sequential transition correlation among successive check-ins. Although such correlation could be learned by a series of Markov chain-based methods [3, 5, 29], merely learning the linear transition correlation as in these methods is still far from accurate POI recommendation. Moreover, the next POI recommendation often faces the data sparsity issue that check-in data has a low sampling rate in both space and time compared with user's real trajectory. To alleviate such problem, contextual embedding based POI recommendation methods have gained popularity, since the contextual information plays an important role in user preferences [28]. Reference [16] integrates the personalized Markov chain with personalized latent behavior patterns learned from contextual features. Reference [36] proposes a time-aware metric embedding approach with asymmetric projection for POI check-in transition probabilities at category level. In addition, inspired by the knowledge graph embedding method, Reference [27] proposes a translation-based framework to embed POIs and location-time factors into transition space for large-scale POI recommendation. Though having taken abundant contextual information into consideration, the embedding-based methods can not well capture the high-order sequential and global spatial-temporal correlations in user's check-in sequence impacting the visiting behavior.

Nowadays, RNN-based models [13, 21, 31, 33, 34, 41] has become pervasive for POI recommendation, as RNN performs promisingly for mining the sequential correlation. As one of the pioneering works of this area, ST-RNN [21] proposes to extend RNN to model local temporal and spatial contexts by introducing a time-specific and distance-specific transition matrix. Apart from ST-RNN, a line of other works also incorporate the spatial-temporal factors into RNNs, including ST-LSTM [13], TMCA [15], STGN [41], and STOD-PPA [18]. CARA [23] captures the user's dynamic preferences by exploiting GRU's gate mechanism. DeepMove [4] designs a multi-modal RNN to capture the sequential transition features. Flashback [33] proposes using all the historical hidden states of RNN with the spatial-temporal context to improve the model's predictive power. There are also a few works studying the session-based next POI recommendation. Session in recommendation means a sequence of items grouped by time slots visited by a user [8]. For example, HST-LSTM [13] proposed a hierarchical spatial-temporal LSTM to explore the area of interest of user in different sessions. Such session information will not be considered in this article, because the user's visit sessions need to be predefined. The drawback of RNN-based methods is the training of RNNs is rather inefficient with high memory requirement and time consumption due to its recurrent architecture. Instead of exploring RNNs, we effectively exploit the strengths of the TCN in this article for handling the long-range input sequence and capturing the sequential transition correlation in user check-in sequence.

Recently, the self-attention network has also been introduced for addressing the next POI recommendation problem. SASRec [12] and AttRec [39] are two powerful item recommender systems that employ attention to learn historical user items' correlation. However, these two works do not consider the time and geographical influence, which are important factors in next POI recommendation scenario. Based on SASRec, SANST [7] adds the GPS location and relative time into the user check-ins embedding vector. Nonetheless, without extra designed learning mechanism for spatial-temporal correlation, SANST is suboptimal for accurate POI recommendation.

GeoSAN [17] exploits an attention-based encoder to consider the geographical information between check-ins, whereas the temporal information is overlooked. STP-UDGAT [19] proposes using the graph attention on POI-POI graphs to fuse the spatial-temporal and user preference factors. The state-of-the-art POI recommender STAN [22] updates the attention by modeling time interval and geographical distance matrices between all candidate POIs and check-ins along the trajectory. However, training these huge matrices is time-consuming and rather inefficient. Moreover, these attention-based methods mostly place little focus on explicit learning of the sequential transition correlation. In this article, we notably improve the POI recommendation performance by considering the sequential correlation with a gated temporal convolutional architecture and further integrating two spatial and temporal learning mechanisms with attention that effectively learn the global pairwise proximity between user check-ins.

3 PROBLEM DESCRIPTION

We consider an LBSN with a set of users denoted as $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$ and a set of POIs denoted as $\mathcal{P} = \{p_1, p_2, \ldots, p_m\}$, where each POI $p \in \mathcal{P}$ is geo-coded by a tuple $g_p = (lon, lat)$, with lon and lat denoting its longitude and latitude coordinates. Next, we introduce in Definitions 1 and 2 the check-in and check-in history, respectively.

Definition 1 (Check-in). We denote a check-in of a user as a 3-tuple (p, g_p, t), which indicates that the user visited POI p in location g_p at a past timestamp t.

Definition 2 (Check-in History). Given an LBSN dataset, a user u's check-in history $\mathcal{H}^u$ is defined as the set of all the check-ins visited by the user in time order, where each element $h_i^u \in \mathcal{H}^u$ denotes user u's ith check-in in the dataset.

Next, we define the next POI recommendation problem.

Definition 3 (Next POI Recommendation). For a given target user $u \in \mathcal{U}$, the next POI recommendation problem aims to recommend a list of top-M POIs that the target user u will prefer to go next.

4 PROPOSED FRAMEWORK

In this section, we present our proposed model in detail. To address the next POI recommendation problem, we propose a novel neural network framework, named as ***Spatial-Temporal Attention over Temporal Convolutional Network (STA-TCN)***, shown in Figure 2. STA-TCN consists of an Input Embedding Layer, a **Gated Temporal Convolutional Network (Gated TCN)**, a **Spatial-Temporal Attention (STAtt)** module, and an Output module. The Input Embedding Layer takes the user's historical check-in sequence as input, which consists of POI, GPS location, and timestamp and outputs their embedding sequences, respectively. Then, the Gated TCN takes the POI embedding sequence as input and outputs the results capturing the sequential transition correlation. Next, the STAtt module takes the outputs of the TCN, jointly with the check-in timestamp and GPS location embedding sequences, as inputs and outputs a final representation vector that captures the global spatial and temporal correlations. Finally, the Output module employs a Selector to yield the POI recommendation result based on the final representation vector.

4.1 Input Embedding Layer

Due to the huge amount of possible POI index p, GPS coordinates g_p, and timestamp t in an LBSN, directly using one-hot encoding to represent each of them produces sparse high-dimensional vector, which will degrade the performance of learning the user preference. The mainstreaming method is first randomly initialize an embedding look-up table for all inputs and then update each

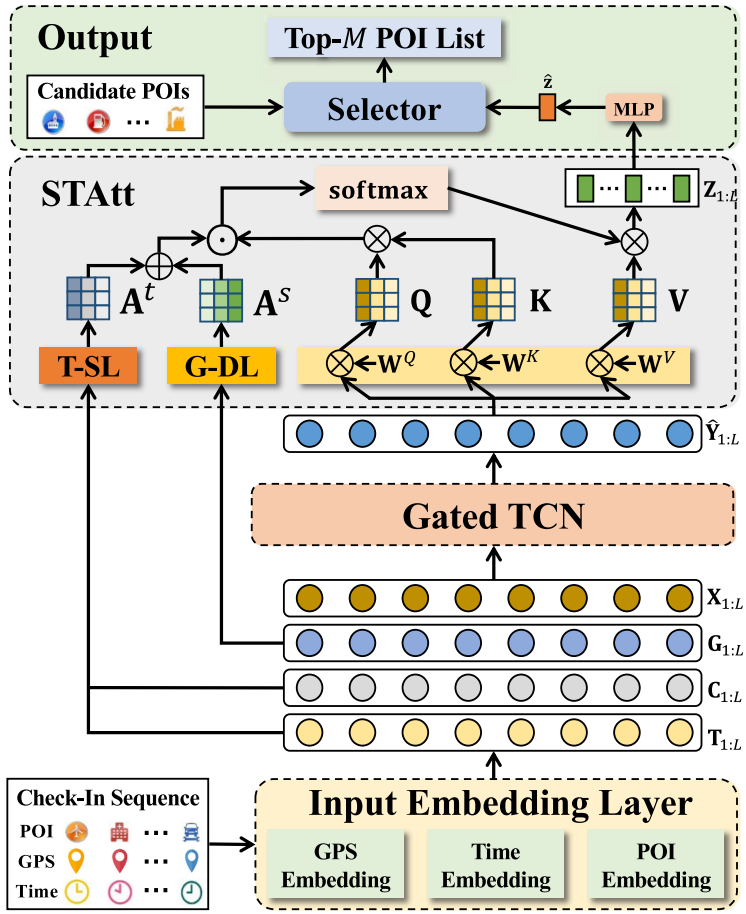
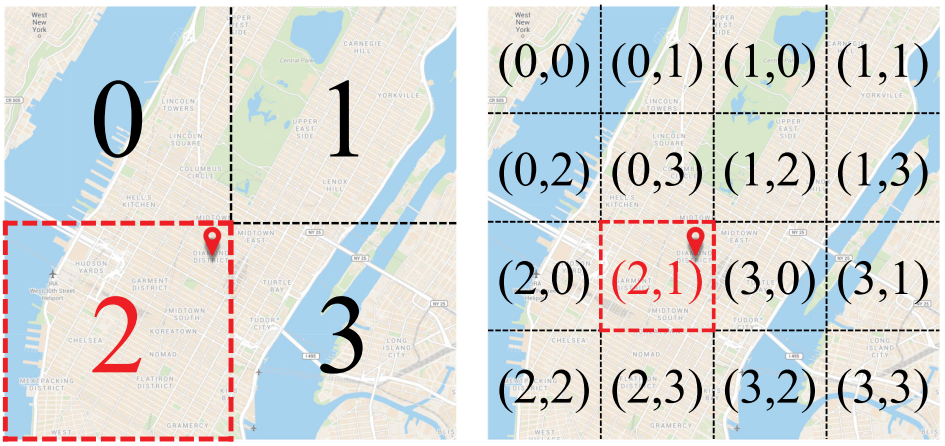


Fig. 2. The proposed STA-TCN model, where $\odot$ denotes element-wise multiplication, $\bigoplus$ denotes weighted sum, and $\otimes$ denotes matrix multiplication.

embedding in table during model training procedure. However, this will lose some auxiliary information of check-in data. To fully leverage the auxiliary information in our data, we create an input embedding layer that embeds each POI, GPS coordinate, and timestamp into a low-dimensional representation vector by performing the following operations:

(1) **GPS coordinates embedding:** We follow the method described in Reference [17] to divide the entire region of interest into grids hierarchically and represent each grid using a base-4 hashed key whose length equals the level of division. For example, as shown in Figure 3, given the division level 2, we have four grids denoted as (0,0), (0,1), (1,0), and (1,1), respectively. Within such Tile Map system, we then represent all the GPS coordinates that fall into the same grid as the r-dimensional vector $\mathbf{g} = (a_1, a_2, \ldots, a_r) \in \mathbb{R}^r$, where a_i denotes the base-4 number of the grid in corresponding level i.

(2) **Timestamp embedding:** Considering that a user usually has different POI visiting behavior in different times of day and different days, we first divide a day into N equal-length time periods $\{\tau_1, \tau_2, \ldots, \tau_N\}$ and further categorize each time period τ_i into τ'^i or $\tau^{\star i}$ according to whether it belongs to weekdays or weekends. Thus, we have $2N$ time periods denoted as $\{\tau'_1, \tau'_2, \ldots, \tau'_N, \tau^\star_1, \tau^\star_2, \ldots, \tau^\star_N\}$. We embed a timestamp t to a vector



(a) Level 1. (b) Level 2.

Fig. 3. An example of the first two levels of a hierarchically gridded map based on the Tile Map system, where the marked location is encoded by 2 in level 1 and by $(2, 1)$ in level 2.

$\mathbf{t} = (\mathbf{0}, \ldots, \mathbf{0}, \mathbf{1}, \mathbf{0}, \ldots, \mathbf{0}) \in \mathbb{R}^{2N}$ with the position of value 1 corresponding to the time period that t belongs to.

(3) **POI embedding:** Clearly, the probabilities at which a POI is visited by users vary in different times of day [20, 37]. We utilize such *time-sensitivity* attribute to categorize a POI. That is, for a POI in a check-in sequence, we count in the training dataset its user check-in frequency in each of the $2N$ time periods defined above. These check-in frequencies are further combined as a time-sensitivity embedding vector $\mathbf{c} = (\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_{2N}}) \in \mathbb{R}^{2N}$, where b_i denotes the user check-in frequency in the ith time period at the considered POI. Since a POI holds a unique GPS coordinate and time-sensitivity attribute simultaneously, we concatenate $\mathbf{c}$ with the GPS embedding vector $\mathbf{g}$ to form the final representation vector of the POI as $\mathbf{x} = \mathbf{c}||\mathbf{g}$.

By jointly carrying out the above three types of embeddings, the Input Embedding Layer will transform a length-L user check-in sequence into four embedding vector sequences, including the POI embedding sequence, denoted as $\mathbf{X}_{1:L}$, the time-sensitivity embedding sequence, denoted as $\mathbf{C}_{1:L}$, the GPS embedding sequence, denoted as $\mathbf{G}_{1:L}$, and the time embedding sequence, denoted as $\mathbf{T}_{1:L}$.

4.2 Gated Temporal Convolutional Network

We propose to employ **temporal convolutional network (TCN)** [1] to learn the sequential transition correlation between user check-ins. Unlike RNNs where the transition information is serially propagated among sequential cells and lots of cells and hidden states are used to store the partial results, TCN conducts convolution operation with a parallel framework, which enables lower memory requirement and faster training speed.

As illustrated in Figure 4, TCN employs a unique convolutional structure called dilated causal convolution, which significantly enhances the learning of sequential transition correlation. The causal convolution operation convolves the inputs with POI embeddings only from the past timestamps, thus preserving the chronological order of the user check-in sequence. Moreover, the dilated nature of TCN's convolution kernel enables exponentially enlarging receptive fields for handling long input check-in sequences.

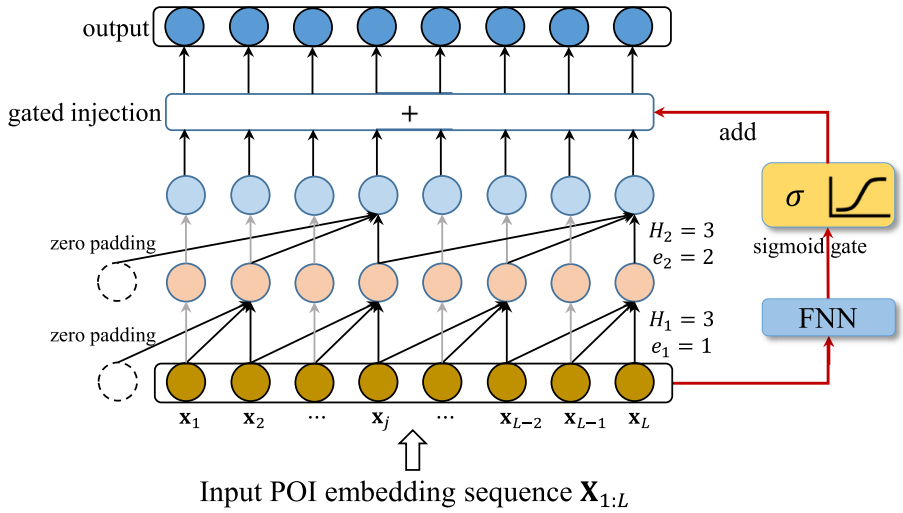


Fig. 4. A gated TCN module consisting of two temporal convolutional layers with kernel size $H_1 = H_2 = 3$, and dilation factors $e_1 = 1, e_2 = 2$, where + denotes element-wise addition and FNN represents feedforward neural network. For better presentation, some convolutional operations over the input elements inside TCN layers are omitted, as marked by the gray arrows.

More specifically, we feed the length-L POI embedding sequence $\mathbf{X}_{1:L} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_L)$ into our TCN, where $\mathbf{x}_i$ denotes the embedding vector of the ith POI in the check-in sequence. Mathematically, the dilated causal convolution on the jth element $\mathbf{x}_j$ in $\mathbf{X}_{1:L}$ conducts the operation,

$$\mathbf{x}_j * \mathbf{f} = \sum_{h=0}^{H-1} \mathbf{f}(h) \cdot \mathbf{x}_{j-eh}, \tag{1}$$

where $*$ denotes the dilated casual convolution operator, $\mathbf{f}$ denotes the convolution filter with kernel size H, e is the dilation factor that controls the dilation window size of the convolution kernel, and $\mathbf{x}_{j-eh}$ denotes the past $(e \times h)$th vector before the current position j.

We apply the above operation across all input POIs using the same kernel weight matrix. Then, the entire computation of a temporal convolutional layer on the input user check-in sequence $\mathbf{X}_{1:L}$ is

$$\mathrm{Y}_{1:L} = \mathrm{ReLU}(\mathbf{W} * \mathrm{X}_{1:L}), \tag{2}$$

where $\mathbf{W}$ denotes the sharing kernel weights matrix in TCN, ReLU is an activation function endowing TCN with non-linearity, and $\mathrm{Y}_{1:L}$ denotes the convolutional output. As shown in Figure 4, to yield an equal-length output sequence, zero padding is added to the beginning of the input of each TCN layer. Then, the transformation of whole TCN with multiple temporal convolutional layers is accessed by series connection of their inputs and outputs. For symbolic brevity, we will use $\mathrm{Y}_{1:L}$ to represent the output of whole TCN in what follows.

For capturing the sequential transition correlation, different input check-in possesses different importance. In light of that, to adaptively select relevant check-ins in input sequence, as shown in Figure 4, we design a *gated injection mechanism* that injects the original input sequence with the output of TCN as

$$\widehat{\mathrm{Y}}_{1:L} = \mathrm{Y}_{1:L} + \mathrm{X}_{1:L} \odot \sigma(\mathbf{W}^g \cdot \mathrm{X}_{1:L} + \mathbf{b}^g), \tag{3}$$

where the final output $\widehat{\mathbf{Y}}_{1:L}$ fuses the convolutional features $\mathbf{Y}_{1:L}$ output by TCN, and the input injection $\mathbf{X}_{1:L}$ controlled by a sigmoid gate $\odot\sigma(\mathbf{W}^g * \mathbf{X}_{1:L} + \mathbf{b}^g)$ with $\mathbf{W}^g$ and $\mathbf{b}^g$ denoting the parameters, $\sigma(\cdot)$ denoting the sigmoid function, and $\odot$ denoting the element-wise multiplication. Here, the sigmoid gate outputs values ranging from 0 to 1, which allows the model to filter the historical check-ins in input sequence $\mathbf{X}_{1:L}$, hence taking into account influence of different weights of check-ins for better representing the user preference of visiting next POI.

4.3 Spatial-temporal Attention Module

Although TCN could help capture the sequential transition correlation in a user's check-in sequence, considering such sequential transition correlation alone is not enough to accurately capture the user preference for POI check-ins. As introduced in Section 1, a user typically performs similar POI visiting behavior at similar time of day and geographically close areas. To capture such global spatial-temporal correlation, on top of the TCN, we propose a ***spatial-temporal attention (STAtt)*** module that augments the canonical self-attention [32] with two novel learning mechanisms, including the ***grid-difference learning (G-DL)*** and the ***time-sensitivity learning (T-SL)*** mechanisms.

To capture the global spatial correlation, we propose G-DL, which takes as input the GPS embedding sequence $\mathbf{G}_{1:L} = (\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_L)$ with $\mathbf{g}_i$ denoting the embedding vector of the ith GPS coordinates in the check-in sequence.

For each pair of embedding vectors $\mathbf{g_i}$ and $\mathbf{g}_j$ in the sequence $\mathbf{G}_{1:L}$, our G-DL mechanism obtains a vector $\mathbf{a}^s_{ij}$ by performing the difference operation given by

$$\mathbf{a}^s_{ij} = \Big(\text{Abs}(\mathbf{g}_i(q) - \mathbf{g}_j(q)), \forall q \in \{1, 2, \ldots, r\} \Big), \tag{4}$$

where $\mathbf{g}_i(q)$ denotes the qth element in vector $\mathbf{g}_i$, and $\text{Abs}(\cdot)$ denotes the absolute value calculation operator. Clearly, the vector $\mathbf{a^s}_{ij}$ represents the grid-level proximity between the pair of GPS coordinates embedded by $\mathbf{g}_i$ and $\mathbf{g}_j$.

Then, the mechanism combines the $\mathbf{a}^s_{ij}$ that corresponds to each pair of the embedding vectors in $\mathbf{G}_{1:L}$ as a vector matrix $\mathbf{M}^s = \{\mathbf{a}^s_{ij}\} \in \mathbb{R}^{L \times L \times r}$ and applies a feed-forward neural network to compute the output spatial correlation score matrix $\mathbf{A}^s \in \mathbb{R}^{L \times L}$ as

$$\mathbf{A}^s = \mathbf{M}^s \cdot \mathbf{W}^s + \mathbf{b}^s, \tag{5}$$

where $\mathbf{W}^s \in \mathbb{R}^r$ and $\mathbf{b}^s \in \mathbb{R}^{L \times L}$ denote the parameters in G-DL mechanism.

As introduced in Section 4.1, users' POI visiting preferences show a time-sensitivity property. That is, the user prefers to visit those POIs that possess high time-sensitivity at interested time period. Enlightened by this, we propose T-SL to capture the global temporal correlation. Such mechanism takes as inputs the time-sensitivity embedding vector sequence $\mathbf{C}_{1:L} = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_L) \in \mathbb{R}^{L \times 2N}$, where $\mathbf{c}_i$ denotes the time-sensitivity embedding vector of the ith POI in the check-in sequence, and the time embedding sequence $\mathbf{T}_{1:L} = (\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_L) \in \mathbb{R}^{L \times 2N}$, where $\mathbf{t}_i$ denotes the embedding vector of the ith timestamp in the check-in sequence. T-SL outputs a temporal correlation score matrix $\mathbf{A^t} \in \mathbb{R}^{L \times L}$ by computing,

$$\mathbf{A}^t = \mathbf{W}^t \cdot \mathbf{T}_{1:L} \cdot \mathbf{C}^\top_{1:L}, \tag{6}$$

where $\mathbf{W}^t \in \mathbb{R}^{L \times L}$ denotes the parameters in our T-SL mechanism.

As aforementioned, to better capture the global spatial-temporal correlation among user check-ins, STAtt integrates the above spatial and temporal correlation score matrices with the scaled dot-product attention and outputs a sequence $\mathbf{Z}_{1:L} \in \mathbb{R}^{L \times (2N+r)}$ according to the following

equation:

$$\mathbf{Z}_{1:L} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top \odot (w_1\mathbf{A}^s + w_2\mathbf{A}^t)}{\sqrt{d_k}} + \mathbf{M}\right)\mathbf{V},$$

- $\mathbf{Q} = \widehat{\mathbf{Y}}_{1:L}\mathbf{W}^Q \in \mathbb{R}^{L\times(2N+r)}$, $\mathbf{K} = \widehat{\mathbf{Y}}_{1:L}\mathbf{W}^K \in \mathbb{R}^{L\times(2N+r)}$, and $\mathbf{V} = \widehat{\mathbf{Y}}_{1:L}\mathbf{W}^V \in \mathbb{R}^{L\times(2N+r)}$ with $\mathbf{W}^Q, \mathbf{W}^K$, and $\mathbf{W}^V \in \mathbb{R}^{(2N+r)\times(2N+r)}$ denoting, respectively, the query matrix, key matrix, and value matrix similar to those in the scaled dot-product attention;
- d_k is the dimension of the key vector;
- w_1 and w_2 are learning parameters that control the weighted sum fusion of spatial and temporal score matrix, and $\odot$ denotes the element-wise multiplication;
- $\mathbf{M} \in \mathbb{R}^{L\times L}$ is a mask matrix, which is filled with $-\infty$ in all upper triangular elements and 0 in the other entries to meet the attention causality constraint.

Finally, the output vector sequence $\mathbf{Z}_{1:L}$, is fed into an MLP aggregator parameterized by $\mathbf{W}_a$ to get the final representation vector $\hat{\mathbf{z}} \in \mathbb{R}^{2N+r}$ for the user preference on the next POI. Hence, we have $\hat{\mathbf{z}} = \mathbf{W}_a \cdot \mathbf{Z}_{1:L}$.

4.4 Output

As in Figure 2, the Output module feeds $\hat{\mathbf{z}}$ into a *selector* to yield the recommendation results. More specifically, Output generates the *candidate POI* sets $\mathcal{V}$ by retrieving R POIs nearest to the user's current check-in location. Then, the selector takes $\mathcal{V}$ as input and calculates the *preference score* on each candidate POI $j \in \mathcal{V}$ based on the inner-product between its POI embedding vector $\mathbf{x}_j$ and the user preference representation vector $\hat{\mathbf{z}}$ as $\text{score}_j = \mathbf{x}_j \cdot \hat{\mathbf{z}}$. Finally, the selector outputs the recommended POI list by picking out the candidate POIs with top-M highest preference scores.

4.5 Training

In the training process, we aim to minimize the loss represented by the cross-entropy function. Given a user i's training target POI set $\mathcal{M}_i$, the target POI $m \in \mathcal{M}_i$, and the corresponding candidate POI sets $\mathcal{V}_m$, the loss is calculated as

$$-\sum_{i\in\mathcal{U}}\sum_{m\in\mathcal{M}_i}\left(\log\sigma(\text{score}_m) - \sum_{j=1,j\neq m}^{|\mathcal{V}_m|}\log(1-\sigma(\text{score}_j)\right),$$

where score_j represents the user preference score on POI j, σ denotes the sigmoid function, and $\mathcal{U}$ is the users set.

5 EXPERIMENTS

5.1 Experimental Setups

5.1.1 Datasets and Preprocessing. Our experiments are conducted on two widely adopted real-world LBSN datasets, Gowalla[1] and Foursquare.[2] The Gowalla dataset contains world-wide check-ins from February 2009 to October 2010, and the Foursquare dataset is collected from February 2010 to January 2011 in New York city. All recorded timestamps of check-ins are UTC time in both datasets. We preprocess both datasets by removing inactive users with less than 10 check-in records, and unpopular POIs with less than 10 visited users. To ensure the diversity of user interests, we filter out the users with less than 5 different visited POIs. Detailed information of the two datasets is given in Table 1.

[1] http://snap.stanford.edu/data/loc-gowalla.html.
[2] https://github.com/yaodi833/serm.

Table 1. Detailed Information of the Evaluated Datasets

Properties	Gowalla		Foursquare	
	Raw	Preprocessed	Raw	Preprocessed
# of Users	22,325	18,929	15,639	4,212
# of POIs	50,835	31,291	43,380	6,553
# of Check-ins	1,502,536	823,136	293,559	168,715

To generate personalized recommendations for a user, we consider their check-in history and use the check-in records from the past fixed d days as the input check-in sequence. The check-ins on the $(d+1)$th day are then treated as the target POIs to be recommended. In this manner, we slide along each user's check-in sequence and partition the data into training and test sets. Specifically, we consider all check-ins from the $(d+1)$th day to the second-last day as the training target POI set, and check-ins on the last day as the test target POI set. d is set as 6 in our experiment. We will give analysis about the choice of value d in Section 5.5.2.

5.1.2 Settings. The hyperparameters are set as follows: The number of time interval divisions in a day $N = 12$ and the level of hierarchically map gridding $r = 17$. The number of convolutional layers and kernel size of TCN are set as 3 and 4, respectively. Moreover, the dilation factors in three convolutional layers of TCN are 1, 1, and 2, respectively. As for the candidate POI sets, we retrieve the $R = 100$ POIs nearest to the current POI. The number of training epoch is set as 30. We train our model using the Adam optimizer with a learning rate of 0.001. To optimize each baseline method's performance, we fine-tune key hyperparameters on the test set, as outlined in their respective papers. All experiments are done on a GeForce GTX 2080Ti GPU, Intel(R) Xeon(R) CPU E5-2650 v3 @ 2.30 GHz, and 256 GB RAM memory.

The evaluation of POI recommendation systems is typically classified into two categories: offline and online. The offline setting measures how well the model ranks the target POIs in a split test set, which is the evaluation approach we adopted in our experiment. The online setting requires the development of a live system to gather feedback from users, which has been adopted in some recent studies, such as References [24, 25, 30]. We believe that the user feedback-based evaluation approach could provide more comprehensive and accurate measurement of the recommendation quality, and we will consider adopting this approach in our future work. We adopt two commonly applied evaluation metrics in our experiments, namely, **Hit Rate (HR)** and **Normalized Discounted Cumulative Gain (NDCG)**. HR@K [17] counts the proportion of the ground truth POI appearing among the recommended top-K POI list, given as

$$HR@K = \frac{|\mathcal{P}^K \cap \mathcal{M}|}{|\mathcal{M}|},$$

where $|\cdot|$ represents the cardinality of a set, $\mathcal{P}^K$ denotes the top-K recommended POI list, and $\mathcal{M}$ is the ground truth POI set.

NDCG@K [11] further evaluates the ranking quality of the ground truth POI in the top-K recommendation list, given as

$$NDCG@K = \frac{DCG@K}{IDCG@K}$$

$$DCG@K = \sum_{i=1}^{K} \frac{G_i}{\log_2(i+1)},$$

where G_i represents the gain score at rank position i, and G_i is 1 if the recommended POI at position i is the ground truth and 0 otherwise. IDCG@K stands for ideal DCG@K representing the maximum possible DCG for a given set of recommendations to normalize the NDCG score to range $[0, 1]$.

5.2 Baselines Methods

We compare our STA-TCN with the following methods:

- **TOP:** A model recommending the most frequently visited POI in the historical check-in sequence for each user.
- **Matrix Factorization (MF)** [14]: We adapt this traditional product recommendation method for our task by using the user's historical visited frequency as their rating score in user-POI matrix. Through factorization of the user-POI matrix, we can generate personalized recommendations for each user.
- **Markov Chain (MC):** A classical sequential model that estimates the user's transition probability matrix among POIs.
- **FPMC** [29]: A model that estimates a personalized transition matrix via matrix factorization techniques.
- **ST-RNN** [21]: A variant of RNN that incorporates spatial-temporal transition matrices to control the hidden states propagation in RNN.
- **HST-LSTM** [13]: A hierarchical LSTM-based model introducing the spatial-temporal factors into the gate mechanism. Notice that session information is not accessible in this application scenario, so its ST-LSTM version is used here.
- **ATST-LSTM** [9]: An updated version of HST-LSTM that leverages the attention mechanism to focus on critical parts of a user check-in sequence.
- **SASRec** [12]: A self-attention-based sequential item recommendation method. We regard the POI as the item to recommend.
- **STGN** [41]: A model that enhances LSTM by adding gates controlled by the spatial-temporal distances between successive check-ins.
- **Flashback** [33]: An RNN-based architecture that performs the external searching over the historical hidden states without modifying the RNN's internal structure.
- **LSTPM** [31]: A model consisting of a non-local network for long-term preference modeling and a geo-dilated LSTM for short-term correlation learning.
- **GeoSAN** [17]: A sequential next POI recommender exploiting a self-attention based geography-encoder to incorporate the GPS information in check-in sequence.
- **STAN** [22]: A state-of-the-art POI recommender that uses a bi-attention network for aggregating the spatial-temporal interval information in the check-in sequence.
- **STA-TCN-past:** A variant of STA-TCN that solely picks from a user's previously visited POIs instead of the universe POI set $\mathcal{P}$ for recommendation.

5.3 Experimental Results

5.3.1 Recommendation Performance. Table 2 depicts the performance of our proposed method as compared to all the state-of-the-art competitors in Gowalla and Foursquare datasets, respectively, where the highest performances have been highlighted in boldface. To mitigate results randomness, each method is executed five times with different random seeds, and the mean value is selected as the final result. It is worth noting that there is a discrepancy between the experimental results of some baselines and their original papers, which is primarily attributable to differences in the test data split strategies. It is easily observable that our proposed STA-TCN consistently

Table 2. Performance Comparison with Baseline Methods on Both Datasets

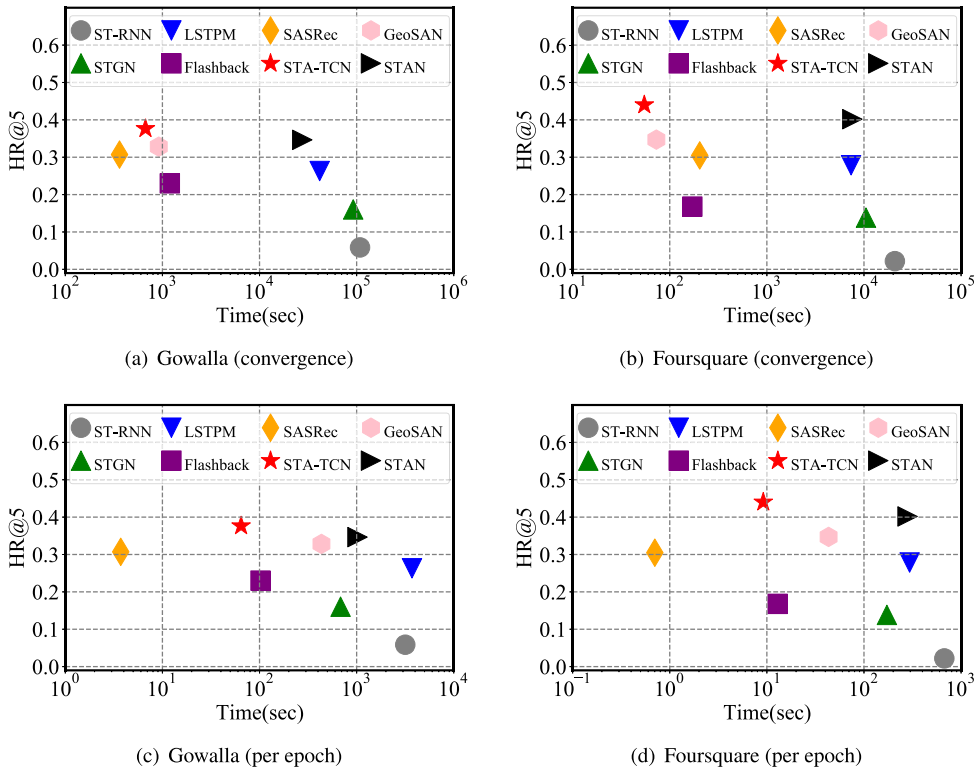
	Gowalla				Foursquare			
	HR@5	HR@10	NDCG@5	NDCG@10	HR@5	HR@10	NDCG@5	NDCG@10
TOP	0.0044	0.0134	0.0033	0.0122	0.0027	0.0101	0.0013	0.0094
MC	0.0131	0.0332	0.0102	0.0292	0.0121	0.0345	0.0116	0.0255
MF	0.0216	0.0407	0.0180	0.0296	0.0207	0.0372	0.0125	0.0299
FPMC	0.0410	0.0554	0.0267	0.0495	0.0478	0.0613	0.0253	0.0490
ST-RNN	0.0586	0.0746	0.0542	0.0613	0.0219	0.0388	0.0157	0.0231
HST-LSTM	0.0844	0.0886	0.0649	0.0699	0.0378	0.0502	0.0203	0.0277
ATST-LSTM	0.1195	0.1261	0.0913	0.1039	0.0440	0.0653	0.0381	0.0341
STGN	0.1603	0.1953	0.1186	0.1299	0.1384	0.1699	0.1112	0.1235
Flashback	0.2301	0.2812	0.1777	0.1924	0.1679	0.2203	0.1291	0.1436
SASRec	0.3071	0.4145	0.2242	0.2589	0.3048	0.4168	0.2230	0.2589
LSTPM	0.2630	0.3377	0.1962	0.2199	0.2785	0.3403	0.2174	0.2358
GeoSAN	0.3283	0.4432	0.2450	0.2820	0.3471	0.4881	0.2322	0.2777
STAN	0.3465	0.4679	0.2594	0.3017	0.4018	0.5186	0.2822	0.3532
STA-TCN-past	0.3232	0.4230	0.2011	0.2559	0.3435	0.4787	0.2340	0.2822
STA-TCN	**0.3765**	**0.5011**	**0.2796**	**0.3200**	**0.4401**	**0.5758**	**0.3253**	**0.3659**

achieves the best performance in terms of all metrics on both datasets. In particular, compared to the second best method STAN, STA-TCN shows an improvement of 8.65%/7.09% and 12.02%/11.03% of HR@5/10 on Gowalla and Foursquare, respectively. Additionally, from the experiment results, we have the following several observations:

First, TOP has undesirable recommendation performance, which proves the necessity of proposing effective next POI recommendation method. MF and MC both obtain inferior performance, because they make two limiting assumptions about user behavior. MF assumes that user preferences for POIs are static. MC assumes that the next POI that a user visits only depends on the current POI that the user is visiting. FPMC slightly improves the results compared with MF and MC, due to its integration of factorization technique and Markov chain. ST-RNN, HST-LSTM, ATST-LSTM, and STGN consistently perform worse than STA-TCN on both datasets, mainly because they only focus on the local transition correlation leaning in recent successive check-ins. Although the attention mechanism empowers the ATST-LSTM with the global correlation learning ability, it is still unable to surpass STA-TCN, due to its less efficient nature in modeling the spatial and temporal factors for user preference.

Flashback performs much better than the above methods by incorporating richer spatial-temporal contexts from RNN hidden states. However, not considering the global correlation among user check-ins degrades its performance. Though not designed for POI recommendation, SASRec performs decently here on two datasets, thanks to its powerful self-attention structure on modeling the global correlation. Compared to the other RNN-based methods, LSTPM achieves the higher performance, verifying its effectiveness of modeling the long-term spatial-temporal correlation with a non-local network. However, LSTPM treats the user check-ins within an entire day as a trajectory and only considers the overall correlation between trajectories. In contrast, our STA-TCN focuses on the pairwise spatial-temporal correlation learning on a finer-grained check-in level, which indeed promotes the performance, as reflected by our results.

The attention-based models GeoSAN and STAN get the third- and second-best performances, respectively, verifying the contribution of the attention on capturing the global correlation. GeoSAN employs a self-attention geographical encoder to consider the global spatial correlation between POIs, leading to a significant performance improvement. However, overlooking the temporal correlation between POIs hinders the optimal performance of GeoSAN. STAN obtains the second-best



(a) Gowalla (convergence) (b) Foursquare (convergence)

(c) Gowalla (per epoch) (d) Foursquare (per epoch)

Fig. 5. Tradeoff between training consumption time and recommendation performance of STA-TCN and baselines.

performance in all methods, mainly because it explicitly aggregates the spatial-temporal interval information between POIs with a bi-attention layer. Compared to STAN, STA-TCN further carefully devises the attention with both a spatial and a temporal proximity learning mechanism to enhance the ability of capturing the spatial-temporal correlation. In addition, our experiments show that STA-TCN outperforms STA-TCN-past significantly. This result demonstrates the effectiveness of our method in handling dynamic changes in user preferences and exploring new POIs for recommendations.

5.3.2 Comparison of Model Efficiency. We discuss the training efficiency of STA-TCN, compared to the baseline methods. We measure the time consumption of all methods from the beginning of training until the convergence of the training loss, as well as training time per epoch under the same training environment. Figure 5 shows the time consumption results and the corresponding HR@5 of each method. Apparently, those markers that are closer to the upper left corner represent the methods with higher training efficiency.

We can observe that most RNN-based models are very time-consuming to train, due to the serial architecture of RNN. In contrast, our proposed STA-TCN significantly improves the training speed and recommendation performance, which verifies the superiority of employing TCN to learn the sequential transition correlation of user check-in sequence over RNN. Moreover, as STA-TCN adopts a low complexity framework with only 3 convolutional layers and single self-attention layer, it also has a faster training speed and better recommendation performance than the other parallel frameworks GeoSAN and STAN. Note that, since the SASRec exploits merely simple self-attention

Table 3. Ablation Analysis on Both Datasets

	Gowalla				Foursquare			
	HR@5	HR@10	NDCG@5	NDCG@10	HR@5	HR@10	NDCG@5	NDCG@10
STA-TCN$_{add}$	0.3665	0.4826	0.2716	0.3091	0.4294	0.5496	0.3092	0.3478
STA-TCN$_{mul}$	0.3404	0.4514	0.2496	0.2854	0.4261	0.5446	0.3039	0.3423
T-SL(-)	0.3731	0.4842	0.2767	0.3125	0.4068	0.5346	0.2920	0.3332
G-DL(-)	0.3412	0.4463	0.2527	0.2865	0.4155	0.5289	0.2963	0.3329
Embedding(-)	0.3564	0.4635	0.2626	0.2971	0.4215	0.5475	0.3032	0.3439
Gate(-)	0.3323	0.4421	0.2444	0.2798	0.3042	0.4039	0.2200	0.2519
STAtt(-)	0.2752	0.3567	0.2033	0.2296	0.2691	0.3642	0.1969	0.2273
STA-TCN	**0.3765**	**0.5011**	**0.2796**	**0.3200**	**0.4401**	**0.5758**	**0.3253**	**0.3659**

layer without extra design for POI recommendation, its training consumption time is rather low here.

5.4 Ablation Study

To verify the contribution of different components of STA-TCN, we vary its structure and evaluate the performances of the following groups of architectural variants:

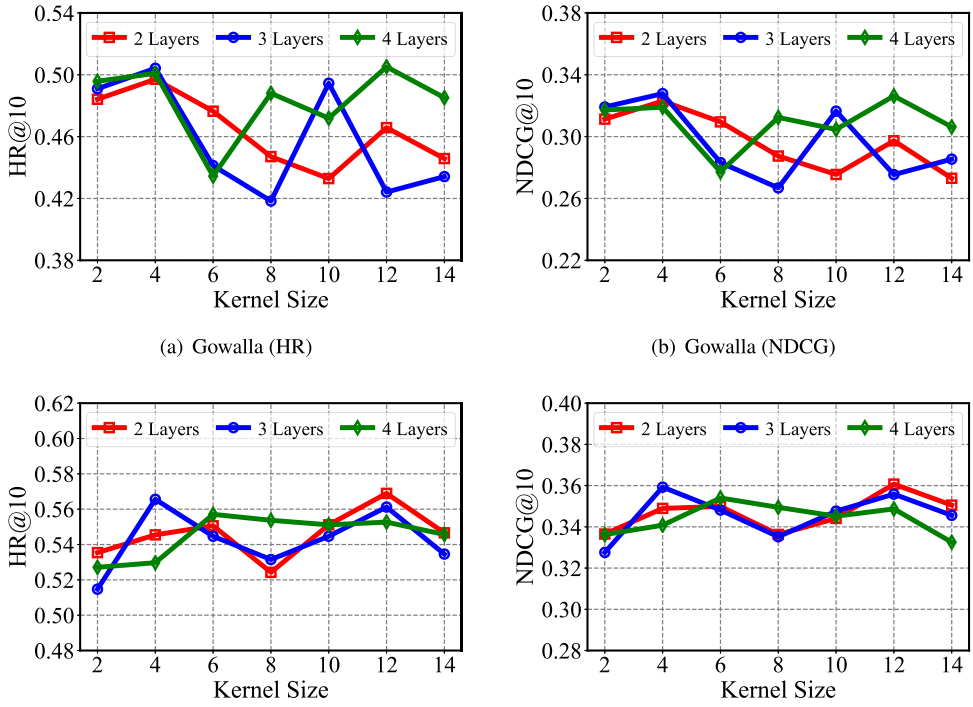
- **STA-TCN$_{add}$** and **STA-TCN$_{mul}$:** Two variants fusing the spatial and temporal correlation score matrices in STAtt module by direct addition and element-wise multiplication, respectively.
- **G-DL(-)** and **T-SL(-):** Two variants removing the spatial and temporal correlation learning components, G-DL and T-SL in STAtt module, respectively.
- **STAtt(-):** A variant removing the whole spatial-temporal attention module (i.e., STAtt), and the gated TCN structure is retained.
- **Gate(-):** A variant removing the gated input injection mechanism of gated TCN module.
- **Embedding(-):** A variant replacing the designed input embedding layer with the one-hot encoding followed by a fully connected embedding layer.

Table 3 shows their performance on both datasets. Through the results, we have the following observations: First, on the fusion method of the spatial and temporal correlation learning component, the weighted sum method outperforms the other two methods, element-wise multiplication and direct addition. Second, removal of G-DL or T-SL component degrades the model performance, verifying the effectiveness of our proposed two proximity learning mechanisms. Furthermore, removing the STAtt module arises the large performance drop, proving the contribution of our designed spatial-temporal attention module. Third, the gated input injection mechanism and input embedding layer both bring the positive impacts on performance, which indicates that the gated input injection mechanism indeed assists our model for better capturing the user preference and our careful design of input embedding is beneficial for the POI recommendation task.

5.5 Analysis of Hyper-parameter

5.5.1 Analysis of TCN's Hyper-parameter. We explore the best hyper-parameter setting of temporal convolutional module by conducting experiments with different combinations of the number of convolutional layers and kernel size in TCN. We change the kernel size from 2 to 14 with a step of 2 and the number of layers from 2 to 4 and evaluate the performance of STA-TCN on both datasets.

As shown in Figure 6, our STA-TCN's performance shows some fluctuations in a limited range against the number of convolutional layers and kernel size. Simultaneously considering the best



(a) Gowalla (HR) (b) Gowalla (NDCG)

(c) Foursquare (HR) (d) Foursquare (NDCG)

Fig. 6. STA-TCN's performance under different numbers of convolutional layers and kernel sizes in its TCN.

setting for both the recommendation performance and model complexity, we adopt 3 convolutional layers and a kernel size of 4 in our TCN.

5.5.2 Influence of Input Days. In this subsection, we investigate the impact of different value d of the input check-in sequence. We vary the input days from 2 to 10 with a step of 2, and 10 to 30 with a step of 5, and evaluate the model recommendation performance in terms of HR@10 and NDCG@10. The results are shown in Figure 7. It is observed that the curve of performance first slightly goes up and then down with the increase of the days on both datasets. More specifically, the best performance is achieved at $d = 6$ over Gowalla and $d = 8$ over Foursquare, respectively. Although longer input days offers more historical information, it will also incur unnecessary noises from those irrelevant check-ins, which causes the performance drop when further increasing the d value. In our experiments, to keep the best performance and as few input days as possible, we consistently set the input check-in days as 6 for both Gowalla and Foursquare datasets.

5.6 Case Study

To further investigate the interpretability of our spatial-temporal learning mechanisms (i.e., G-DL and T-SL), we pick a user's historical check-in sequence from Foursquare and plot their correlation results output by these two mechanisms. As shown in Figure 8, check-ins are marked 1 to 6 by their visited time order in the map. The matrix $\mathbf{S}$ and $\mathbf{T}$ at the right part are the output matrices of G-DL and T-SL, respectively, where the (i, j)th entry denotes the correlation score between check-in i and j. We can observe that:

- Those geographically close check-ins, e.g., 5 and 6, get high correlation score in the matrix S. This is because check-ins that are geographically close are more likely to be correlated,

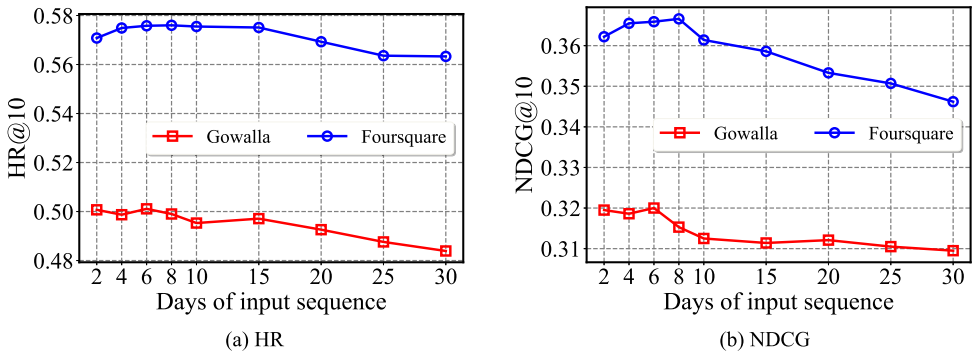


Fig. 7. STA-TCN's performance under different days of input check-in sequence.

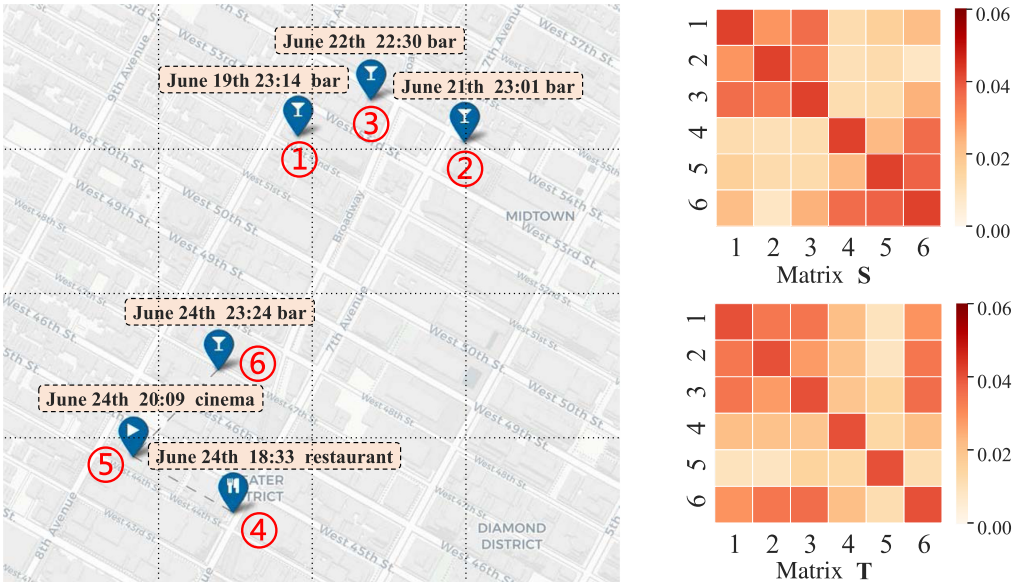


Fig. 8. Case study about the spatial-temporal correlation learning mechanisms.

as they are likely to be visited by the user for similar reasons. As shown in Figure 8, a user who checks in to a cinema is more likely to also check in to a nearby bar.

- Those check-ins visited by the user at the similar time of the day, e.g., 2 and 6, get high correlation score in the matrix **T**. This is because check-ins that are visited at the similar time of the day are more likely to be correlated, as they are likely to be part of the same activity. As shown in Figure 8, a user who checks in to a bar at midnight is more likely to also check in to a bar at similar time of another day.

Hence, with our designed spatial-temporal learning mechanisms, the model better focuses on check-ins with closer spatial-temporal proximity. This is a significant property for next POI recommendation systems, as it allows the model to recommend more relevant POIs to users.

6 CONCLUSION

In this article, we propose a novel neural network framework STA-TCN for the next POI recommendation. Instead of the pervasively adopted RNNs, our STA-TCN framework employs a TCN

augmented by an additional gated input injection mechanism to capture the sequential transition correlation. Furthermore, STA-TCN fuses two novel grid-difference and time-sensitivity learning mechanisms with attention to capture the global spatial-temporal correlations in a user's check-in sequence. Our extensive experiments with two large-scale real-world LBSN datasets show that STA-TCN not only has a better POI recommendation performance, but also a higher model efficiency than existing state-of-the-art methods. Meanwhile, the ablation experiment and case study well verify the rationality and effectiveness of each designed component in STA-TCN. In future work, we will pursue an effective solution to devise our model for alleviating the check-in data sparsity problem, which is a prevailing challenge in the next POI recommendation task.

REFERENCES

[1] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271* (2018).

[2] Preeti Bhargava, Thomas Phan, Jiayu Zhou, and Juhan Lee. 2015. Who, what, when, and where: Multi-dimensional collaborative recommendations using tensor factorization on sparse user-generated data. In *Proceedings of the International World Wide Web Conference (WWW)*.

[3] Jun Chen, Chaokun Wang, and Jianmin Wang. 2015. A personalized interest-forgetting Markov model for recommendations. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

[4] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. 2018. DeepMove: Predicting human mobility with attentional recurrent networks. In *Proceedings of the International World Wide Web Conference (WWW)*.

[5] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, and Yeow Meng Chee. 2015. Personalized ranking metric embedding for next new POI recommendation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

[6] Rong Gao, Jing Li, Xuefei Li, Chenfang Song, Jun Chang, Donghua Liu, and Chunzhi Wang. 2018. STSCR: Exploring spatial-temporal sequential influence and social information for location recommendation. *Neurocomputing* 319 (2018).

[7] Qianyu Guo and Jianzhong Qi. 2020. SANST: A self-attentive network for next point-of-interest recommendation. *arXiv preprint arXiv:2001.10379* (2020).

[8] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[9] Liwei Huang, Yutao Ma, Shibo Wang, and Yanbo Liu. 2019. An attention-based spatiotemporal LSTM network for next POI recommendation. *IEEE Transactions on Services Computing* 14, 6 (2019), 1585–1597.

[10] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. 2010. *Recommender Systems: An Introduction*. Cambridge University Press.

[11] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.* 20, 4 (2002), 422–446.

[12] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*.

[13] Dejiang Kong and Fei Wu. 2018. HST-LSTM: A hierarchical spatial-temporal long-short term memory network for location prediction. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

[14] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[15] Ranzhen Li, Yanyan Shen, and Yanmin Zhu. 2018. Next point-of-interest recommendation with temporal and multi-level context attention. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*.

[16] Xin Li, Dongcheng Han, Jing He, Lejian Liao, and Mingzhong Wang. 2019. Next and next new POI recommendation via latent behavior pattern inference. *ACM Trans. Inf. Syst.* 37, 4 (2019), 1–28.

[17] Defu Lian, Yongji Wu, Yong Ge, Xing Xie, and Enhong Chen. 2020. Geography-aware sequential location recommendation. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

[18] Nicholas Lim, Bryan Hooi, See-Kiong Ng, Xueou Wang, Yong Liang Goh, Renrong Weng, and Rui Tan. 2021. Origin-aware next destination recommendation with personalized preference attention. In *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM)*.

[19] Nicholas Lim, Bryan Hooi, See-Kiong Ng, Xueou Wang, Yong Liang Goh, Renrong Weng, and Jagannadan Varadarajan. 2020. STP-UDGAT: Spatial-temporal-preference user dimensional graph attention network for next POI recommendation. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*.

[20] Yan Lin, Huaiyu Wan, Shengnan Guo, and Youfang Lin. 2021. Pre-training context and time aware location embeddings from spatial-temporal trajectories for user next location prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

[21] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

[22] Yingtao Luo, Qiang Liu, and Zhaocheng Liu. 2021. STAN: Spatio-temporal attention network for next location recommendation. In *Proceedings of the International World Wide Web Conference (WWW)*.

[23] Jarana Manotumruksa, Craig Macdonald, and Iadh Ounis. 2018. A contextual attention recurrent architecture for context-aware venue recommendation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.

[24] David Massimo and Francesco Ricci. 2021. Popularity, novelty and relevance in point of interest recommendation: An experimental analysis. *Inf. Technol. Tour.* 23 (2021), 473–508.

[25] David Massimo and Francesco Ricci. 2022. Building effective recommender systems for tourists. *AI Mag.* 43, 2 (2022), 209–224.

[26] Andriy Mnih and Russ R. Salakhutdinov. 2008. Probabilistic matrix factorization. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*.

[27] Tieyun Qian, Bei Liu, Quoc Viet Hung Nguyen, and Hongzhi Yin. 2019. Spatiotemporal representation learning for translation-based POI recommendation. *ACM Trans. Inf. Syst.* 37, 2 (2019), 1–24.

[28] Hossein A. Rahmani, Mohammad Aliannejadi, Mitra Baratchi, and Fabio Crestani. 2022. A systematic analysis on the impact of contextual information on point-of-interest recommendation. *ACM Trans. Inf. Syst.* 40, 4 (2022), 1–35.

[29] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the International World Wide Web Conference (WWW)*.

[30] Pablo Sánchez and Alejandro Bellogín. 2022. Point-of-interest recommender systems based on location-based social networks: A survey from an experimental perspective. *ACM Comput. Surv.* 54, 11s (2022), 1–37.

[31] Ke Sun, Tieyun Qian, Tong Chen, Yile Liang, Quoc Viet Hung Nguyen, and Hongzhi Yin. 2020. Where to go next: Modeling long-and short-term user preferences for point-of-interest recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*.

[33] Dingqi Yang, Benjamin Fankhauser, Paolo Rosso, and Philippe Cudre-Mauroux. 2020. Location prediction over sparse user mobility traces using RNNs: Flashback in hidden states. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

[34] Di Yao, Chao Zhang, Jianhui Huang, and Jingping Bi. 2017. SERM: A recurrent model for next location prediction in semantic trajectories. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*.

[35] Lina Yao, Quan Z. Sheng, Yongrui Qin, Xianzhi Wang, Ali Shemshadi, and Qi He. 2015. Context-aware point-of-interest recommendation using tensor factorization with social regularization. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1007–1010.

[36] Haochao Ying, Jian Wu, Guandong Xu, Yanchi Liu, Tingting Liang, Xiao Zhang, and Hui Xiong. 2019. Time-aware metric embedding with asymmetric projection for successive POI recommendation. In *Proceedings of the International World Wide Web Conference*.

[37] Fuqiang Yu, Lizhen Cui, Wei Guo, Xudong Lu, Qingzhong Li, and Hua Lu. 2020. A category-aware deep model for successive POI recommendation on sparse check-in data. In *Proceedings of the International World Wide Web Conference (WWW)*.

[38] Lu Zhang, Zhu Sun, Jie Zhang, Yu Lei, Chen Li, Ziqing Wu, Horst Kloeden, and Felix Klanner. 2020. An interactive multi-task learning framework for next POI recommendation with uncertain check-ins. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

[39] Shuai Zhang, Yi Tay, Lina Yao, Aixin Sun, and Jake An. 2019. Next item recommendation with self-attentive metric learning. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI)*.

[40] Kangzhi Zhao, Yong Zhang, Hongzhi Yin, Jin Wang, Kai Zheng, Xiaofang Zhou, and Chunxiao Xing. 2020. Discovering subsequence patterns for next POI recommendation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

[41] Pengpeng Zhao, Haifeng Zhu, Yanchi Liu, Jiajie Xu, Zhixu Li, Fuzhen Zhuang, Victor S. Sheng, and Xiaofang Zhou. 2019. Where to go next: A spatio-temporal gated network for next POI recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Received 1 August 2022; revised 28 March 2023; accepted 26 April 2023