Learning Social Relations and Spatio-Temporal Trajectories for Next Check-in Inference

Wenwei Liang, Wei Zhang, *Member, IEEE*

***Abstract*—The proliferation of location-aware social networks (LSNs) has facilitated the researches of user mobility modeling and check-in prediction, thereby benefiting various downstream applications like precision marketing and urban management. Most of existing studies only focus on predicting the spatial aspect of check-ins, whereas the joint inference of the spatial and temporal aspects more fits the real application scenarios. Moreover, although social relations have been extensively studied in recommender system, only a few efforts have been observed in the next check-in location prediction, leaving room for further improvement. In this paper, we study the next check-in inference problem which demands the joint inference of the next check-in location (Where) and time (When) for a target user (Who). We devise a model named ARNPP-GAT, which combines an attention-based recurrent neural point process (ARNPP) with a graph attention networks (GAT). The core technical insight of ARNPP-GAT is to integrate user long-term representation learning, short-term behavior modeling, and temporal point process into a unified architecture. Specifically, ARNPP-GAT first leverages graph attention networks to learn the long-term representation of users by encoding their social relations. More importantly, the attention-based recurrent neural point process endows the model with the capability of characterizing the effects of past check-in events and performing multi-task learning to yield the next check-in time and location prediction. Empirical results on two real-world datasets demonstrate ARNPP-GAT is superior compared with several competitors, validating the contributions of multi-task learning and social relation modeling.**

***Index Terms*—multi-task learning, check-in prediction, deep recurrent modeling, temporal point process, graph attention networks**

I. Introduction

The rapid development of mobile communication technology facilitates users to share their real-time location in location-aware social networks (LSNs) with their friends. This trend significantly promotes the growth of LSNs and generates a huge amount of user spatio-temporal trajectories. Under this background, user mobility modeling and inference has incurred a wide spectrum of industrial applications, including urban planning, location-aware recommendation, and targeted marketing, to name a few. Consequently, a lot of research attention has bee paid to this domain. In the literature, one direction aims to recommend all the locations a target user will visit in the future [46], [47], and another direction focused on predicting a user's next check-in location [4], [49]. In this paper, we address the latter perspective, which is more coincident with the sequential nature of user mobility.

Most of existing studies in this regard optimize their models to solely estimate check-in locations [3], [4], [11], [12] or just regard check-in time information as an additional type of model input [10], [20], [23], [49]. However, these studies overlook the utilization of the temporal signal contained in user spatio-temporal trajectories as another supervisory signal to benefit model optimization. As such, how to simultaneously utilize these two aspects of information to guide the model learning is critical for effective user mobility modeling, especially considering the sparsity nature of check-in data in LSNs [27] (the sparsity of the adopted datasets is shown in Table I). In addition to the above consideration, another manner to alleviate the sparsity issue is to leverage social relations among users. This is because friends tend to affect users' decisions and behaviors to some extent [22], thus preference propagation among users is feasible to enhance the understanding of the users with sparse behaviors.

Inspired by the perspective of multi-task learning [9], [50], we explore the problem of predicting where a target user will visit next and at what time, based on his spatio-temporal trajectory up to now. The most relevant study is RMTPP [7], which leverages point process [8] to model event temporal dynamics and deep recurrent neural networks (RNNs) [14] to predict which event will happen. However, two main issues of RMTPP still remain unresolved. 1) RMTPP employs deep sequential model to learn from users' recent check-ins to represent their recent dynamic preference, yet users' static and long-term preference is overlooked, which might affect the accurate user behavior modeling. 2) The conditional intensity function proposed in RMTPP only uses the most recent hidden representation of users' historical check-in events, simply inheriting the idea of the standard point process. Actually, more powerful Hawkes process [8] exists, which characterizes the intensity function by explicitly capturing the effect of past events on the future event, not limited to the most recent one.

To address the above issues in the multi-task learning for next check-in time and location inference, we develop a model named ARNPP-GAT, which combines an attention-based recurrent neural point process with graph attention networks, a natural extension of RMTPP. Specifically, ARNPP-GAT divides representations of users into two categories: long-term preference and short-term preference. Long-term preference

Manuscript received xx; revised xxx; accepted xxx. Date of publication xxx; date of current version xxx. This work was supported in part by the National Natural Science Foundation of China (Grant 61702190 and Grant U1609220) and in part by the foundation of Key Laboratory of Artificial Intelligence, Ministry of Education, P.R. China. *(Corresponding author: Wei Zhang.)*

W. Liang is with the School of Computer Science and Techonology, East China Normal University, Shanghai 200062, China (e-mail: 51174500033@stu.ecnu.edu.cn).

W. Zhang is with the School of Computer Science and Technology, East China Normal University, Shanghai 200062, China, and also with the MOE Key Lab of Artificial Intelligence, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: zhangwei.thu2011@gmail.com).

is learned by encoding social relations with graph attention networks [33], while short-term preference is revealed in the sequential modeling of users' recent check-in behaviors. The blending of long-term and short-term representations benefits the next check-in inference (issue 1). Moreover, temporal point process (TPP) is utilized to capture both the continuity of time information associated with spatio-temporal trajectories. Benefiting from this, multi-task learning w.r.t. time and location inference could be performed. To explicitly quantify the effects of past check-in events, attention mechanism is used to directly associate them with the current conditional intensity function of TPP (issue 2).

In a nutshell, we summarize the main contributions of this paper as follows:

- We address the user mobility modeling from the multi-task perspective and propose a novel model named ARNPP-GAT. To the best of our knowledge, ARNPP-GAT is the first to seamlessly combine user representation learning, deep recurrent modeling, and point process into a unified model.
- ARNPP-GAT introduces an attention based method to quantify the effect of past check-in events in the conditional intensity function for deep recurrent modeling, and combines graph attention networks to help learn user long-term representations.
- Extensive experiments on two real datasets show that the proposed model has consistently better performance than several strong competitors for both tasks.

A preliminary short version of this paper is published in [19]. Compared with the preliminary version, we conduct a substantial extension in this paper from three aspects. First of all, we reformulate the problem setting by additionally consider the impact of social relations between users on predicting next check-in time and location. We therefore develop an enhanced version of the original model by adding graph attention networks to learn user social-aware interest from social relations. The user social-aware interest, along with user dynamic interest, jointly affects the predicted probabilities of next check-in time and location, enabling to better grasp user interest. Secondly, we provide a more detailed introduction of the model design, including graphical illustration and mathematical computational equations, thereby making a better understanding of the key parts of this paper. Finally, we enhance the experiment part by adding some more baselines, which model social relations as well for the location prediction task and a Hawkes process based baseline for the time prediction task. We also design a variant of our final model for both of the two tasks. The extensive results show more insights into the effectiveness of the proposed model and the benefits of incorporating social relations.

II. Related Work

A. Next Check-in Inference

In the research domain of predicting next check-ins, most pioneering studies concentrate on **location prediction**. Traditional approaches [4], [49], [11] are usually based on latent factor models to cope with this problem, with the consideration of their big success on recommender system. These studies assume that the determination of the next check-in location only depends on a user's most recently visited location, which does not conform to the real situation.

More recently, since deep recurrent models have surged as the paradigm for sequential modeling tasks, such as language modeling [26] and speech recognition [13], some studies attempt to leverage RNNs for the next check-in location prediction problem [3], [10], [20], [23]. The major strength is that RNNs regard each mobile trajectory as a whole for modeling. To be specific, DeepMove [10] takes multi-modal embeddings as input to an RNN. CARA [23] incorporates spatial and temporal information into the gating unit of RNNs. Although these models behave better than traditional approaches, they neglect the nature of the next check-in prediction is indeed a multi-task learning problem, involving both the location and time inference.

On the other hand, only a few studies [44], [45] investigate the next check-in **time prediction** task. Yang et al. [45] first extracted user mobility and check-in features and proposed a survival based model to generate check-in time prediction. Afterwards, they revised the original model and further introduced a deep point process model [44] to achieve good performance. Nevertheless, they assume next check-in locations are pre-specified and thus cannot provide location prediction. As aforementioned, RMTPP [7] is the most relevant study for our multi-task problem. However, due to its two intrinsic limitations, there is still room for improving the check-in location and time inference performance, which is the focus of this study.

B. Point Process and Attention Mechanism

Point process is a stochastic process that exhibits a wide spectrum of applications in temporal event sequence modeling. For example, Zhao et al. [51] introduced a coupled point process to predict the dynamic popularity of social media short messages such as tweets. Wang et al. [36] proposed a coevolutionary point process to capture the interactions between users and items. Liu et al. [21] designed a feature-driven point process to forecast paper citation count. By further combining the advantages of deep learning with point process, a recurrent point process model [7] and a neural Hawkes process model [25] are presented, enhancing the form of their intensity functions to be more flexible. On this basis, we take a further step by incorporating the idea of Hawkes process into deep recurrent modeling through attention mechanism.

Attention mechanism [2] is widely employed to select important parts from image [1] or text [35], leading to significant improvements. In this paper, we propose a simple attention based method to automatically quantify the effect of users' past check-in events on the happening of the next check-in event.

C. Graph Neural Networks

Graph Neural Networks (GNNs) [53] have proven to be the state-of-the-art approaches to learn diverse graph data. They are good at propagating node representations along with edges and updating node representations with the combination of the representations of the current node and its neighbors. Graph convolutional networks (GCNs) [18] and Graph Attention Networks (GATs) [33] are representative types of GNNs. The former one equally aggregates the neighbors' embeddings in each convolution by using the existing edge weights to quantify the different contributions of the neighbors. By contrast, the latter leverages attention mechanism to learn the edge weights and characterize the contributions. Existing studies have applied GATs to social influence analysis [40], [29], conversation generation [52], and relevance matching [48], etc. Since the original social graph only contains binarized edges, in this paper we leverage GATs to characterize the relation strength of social links, which provides more informative signals to learn user long-term representations. Note that the recent progress in GNNs shows that they have been adapted to the sequential recommendation scenario [39], [34]. However, these studies emphasize the importance of constructing item-item graphs in a general session-based scenario. This in contrast to our study that aims at learning from social relations for the joint inference of next check-in location and time.

III. Preliminaries

In this section, we first give the main notations used throughout this paper and formally define the studied problem. After that, we review the basic concepts of temporal point process theory and the mathematical expressions of the conditional intensity function.

A. Problem Formulation

Definition 1. *(Check-in). Assume that the whole check-in dataset is C, U represents a user set, L corresponds to a set of unique locations, and T means the domain of time. Each check-in record c is represented by a triplet $(u, l, t) \in U \times L \times T$, which indicates user u has visited location l at timestamp t.*

Definition 2. *(User Social Graph). We organize the user relations in the form of an undirected graph $G_U = (V_U, E_U)$, where V_U ($|V_U| = |U|$) and E_U denote a node set and an edge set, respectively. Each node is assumed to be connected to itself, i.e., $(v; v) \in E_U$ for any v. We further represent the graph with an adjacency matrix A. If there exists an edge between node i and node j, which means $A_{ij} = 1$ and $A_{ji} = 1$, otherwise $A_{ij} = 0$ and $A_{ji} = 0$.*

We arrange all the check-ins belonging to the same user in a chronological order and regard them as a whole check-in sequence. For user u, the sequence is composed of check-in records $C_u = \{c_{u,1}, \ldots, c_{u,k}, \ldots\}$. Meanwhile, we represent the time interval of two consecutive check-ins as τ, *i.e.*, $\tau_1 = 0$ and $\tau_k = t_k - t_{k-1}$.

Definition 3. *(Spatio-Temporal Trajectory). Given a specified maximum time interval ζ, a spatio-temporal trajectory of user u is defined as a sequence consisting of multiple consecutive check-ins, i.e., $S_u = \{c_{u,1}, \ldots, c_{u,k}, \ldots, c_{u,j}\}$, s.t., $t_k - t_{k-1} < \zeta, \forall k \in \{2, \ldots, j\}$, where j is the length of the trajectory.*

In other words, a whole user check-in sequence C_u can be divided into several non-overlapping spatio-temporal trajectories, i.e., $C_u = \{S_u^1, \ldots, S_u^{n_u}\}$, where n_u is the trajectory number of user u. For simplicity, we omit the superscript of S_u if not specified.

Problem 1. *(Next Check-in Location and Time Inference Problem). Assuming social graph G_U, user u, and his current trajectory S_u are given, the aim of the problem is to infer the next check-in c_u to be visited by the user, including the corresponding location l_u and time t_u derived from time interval τ_u.*

B. Basics of Temporal Point Process

Temporal point process is an effective mathematical tool to model temporal sequential data by using the conditional intensity function $\lambda^*(t)$: for a short time window $[t, t + dt)$, $\lambda^*(t)dt = P\{c_u \text{ in } [t, t + dt)|H_t\}$ is the probability of the occurrence of user u's new check-in conditioned on the historical records H_t. Assuming $f^*(t)$ denotes the density function and $F^*(t)$ is the cumulative distribution function, $\lambda^*(t)$ is given as:

$$\lambda^*(t)dt = \frac{f^*(t)dt}{\big(1 - F^*(t)\big)}. \quad (1)$$

Based on the above equation, the density function can be represented as:

$$f^*(t) = \lambda^*(t) \exp(-\int_{t_j}^{t} \lambda(\epsilon)d\epsilon), \quad (2)$$

where t_j is the timestamp of the last event.

The conditional intensity function $\lambda^*(t)$ could have different forms. The homogeneous Poisson process [17] has the simplest intensity function, which is non-negative and independent of the history, *i.e.*, $\lambda^*(t) = \lambda(0) \geq 0$, where $\lambda(0)$ is the base intensity. Inhomogeneous Poisson process defines its intensity function $g(t)$ dependent on t but still irrelevant to the history, *i.e.*, $\lambda^*(t) = g(t) \geq 0$. By contrast, Hawkes process [8] is designed to capture the explicit effect of the past history events with the intensity function defined as $\lambda^*(t) = \lambda(0) + \alpha \sum_{t_j<t} g(t, t_j)$, where $g(t, t_j)$ is a triggering kernel function relying on the history up to time t. However, all the above models assume specific parametric forms of intensity functions, which limit their expressive ability. To alleviate this issue, the studies [7], [41] define the intensity functions dependent on the hidden states learned by RNNs, providing more functional capacity and achieving state-of-the-art performance.

IV. The Computational Approach

In this section, we elaborate on the proposed approach ARNPP-GAT. We first show how to encode user dynamic interests by gated recurrent unit (GRU), followed by the introduction of social graph learning with GATs that update the node representations for long-term user preference. Afterwards, we look into the details of the multi-task prediction module. Finally, we explain how the framework can be learned in an end-to-end fashion. The graphical architecture is illustrated in Figure 1.

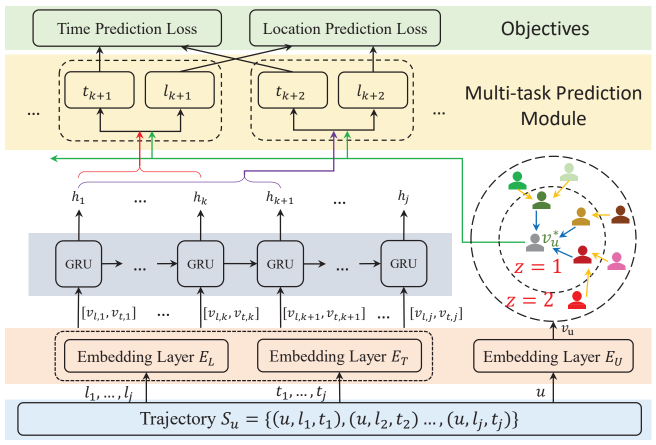


Fig. 1: Architecture of the proposed ARNPP-GAT model.

A. *User Short-term Behavior Modeling*

Our model ARNPP-GAT is an end-to-end learning framework which takes user ID, user social graph, location ID, time interval (hours) and timestamp (hours in a week) as inputs, and outputs the inferred next check-in location and time information.

In practice, we first build three dictionaries, each of which contains all valid timestamps, location IDs, and user IDs in a target dataset, respectively. Upon this, each timestamp or ID could be converted into a one-hot vector which is later fed into the corresponding embedding layer. As shown in Figure 1, v_l and v_t represent the dense vectors of location representation and time representation, respectively. We concatenate and input them to the subsequent layers.

Regarding to the recurrent modeling part of the model, we consider two widely used variants, long short-term memory (LSTM) [14], and GRU [6]. LSTM is composed of input, output, and forget gates to transfer information and update states. In comparison, GRU is a light-weighted variant of LSTM, with one update gate to replace the forget gate and the input gate in LSTM. For the simplicity of GRU, we adopt it in ARNPP-GAT.

Given input $V_k = [v_{l,k}; v_{t,k}]$ at time step k, hidden state h_{k-1} at the last time step, and candidate state $\tilde{h}_k$ of GRU, we obtain the current hidden state h_k through the following equations,

$$\begin{bmatrix} \omega_k \\ r_k \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \end{bmatrix} (W_{\omega r} \cdot [V_k; h_{k-1}] + b_{\omega r}), \tag{3}$$

$$\tilde{h}_k = \tanh(W_c \cdot [r_k \odot h_{k-1}; V_k] + b_c), \tag{4}$$

$$h_k = (1 - \omega_k) \odot \tilde{h}_k + \omega_k \odot h_{k-1}, \tag{5}$$

where ω_k and r_k are the activation states of the update gate and reset gate, respectively. σ denotes the sigmoid function and $\odot$ represents the Hardmard product operation. $\tilde{h}_k$ is activated by element-wise $tanh(x)$. $W_{\omega r}, W_c, b_{\omega r}$, and b_c are the parameters of GRU to be learned. After looping over each time step, the hidden state sequence $\{h_1, \ldots, h_j\}$ is collected.

B. *User Long-term Representation Learning*

ARNPP-GAT utilizes GATs to learn deep representations for user long-term and static preference from a given social graph. As aforementioned, we map each user ID to a low-dimensional vector (e.g., v_u) through a user embedding layer. One-layered GATs could only capture information from immediate neighbors. When stacking multiple GATs layers, information from distant neighborhoods is grasped. Specifically, given the social relation matrix A, GATs update user representations at the z-th layer based on neighbor user representations at the previous layer. The computational formula is formally defined as follows:

$$v_u^z = \rho(\sum_{u'=1}^{|U|} A_{uu'} W^z v_{u'}^{z-1} + b^z), \tag{6}$$

where W^z and b^z are the trainable parameters at layer z, and ρ is an activation function (e.g. ReLU). v_u^0 is set as the initial user embedding v_u.

To differentiate the relative importance of social edges, GATs dynamically determine the weights of social edges (equal to one in the original graph) in the adjacency matrix. Multi-head self-attention [32] mechanism is adopted to measure the relevance between different nodes, allowing each node to have different representation subspaces. It first linearly projects v_u^{z-1} into M subspaces, equal to the number of heads. Correspondingly, M attention functions are used in parallel to summarize the representations of user neighborhoods. They are concatenated together and regarded as the updated representations:

$$\begin{aligned} \text{MultiHead}(v_U^{z-1}) &= [\text{head}_1; \text{head}_2; ..., \text{head}_M] W^O, \\ \text{head}_m &= \text{Self-Attention}(v_U^{z-1} W_m^Q, v_U^{z-1} W_m^K, v_U^{z-1} W_m^V), \end{aligned} \tag{7}$$

where the projections matrices for each head W_m^Q, W_m^K, W_m^V and W^O are trainable parameters. For simplicity, we omit the layer subscript $z-1$. In actuality, the projection parameters are not shared across different GATs layers. Given this, the Self-Attention function is defined as follows:

$$\begin{aligned} &\text{Self-Attention}(Q, K, V) \\ &= \text{Softmax}(\frac{v_U^{z-1} W_m^Q \times (v_U^{z-1} W_m^K)^T}{\sqrt{d/M}}) v_U^{z-1} W_m^V, \end{aligned} \tag{8}$$

where *query* Q, *key* K, and *value* V are the same projection matrices as in Equation 7, d is the dimension of user embedding. Moreover, in order to have a stable update of user static presentation, we update it by linear interpolation:

$$v_u^* = \mu * v_u^z + (1 - \mu) * v_u, \tag{9}$$

where $\mu < 1$ is a hyper-parameter tuned on validation datasets. We execute the whole GATs procedure for every several epochs. For future work, we could extend the social relation learning approach if the time information of social edges is given [37].

C. *Multi-task Prediction Module*

Figure 2 shows the details of the two consecutive multi-task prediction modules. We pass the hidden state sequence with the user dense vector as input to the module.

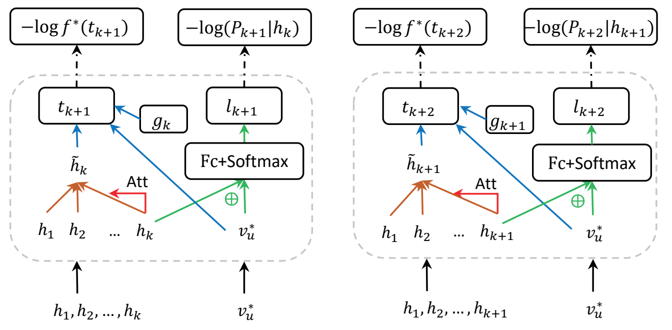


Fig. 2: Detailed illustration of two consecutive multi-task prediction modules at time step k and $k+1$. $\oplus$ represents the concatenate operator. $Fc + Softmax$ is a fully connected layer followed by a Softmax layer.

1) Check-in Time Prediction: We first focus on the left part of the multi-task prediction module. As aforementioned, previous studies [7], [28], [38] regard the most recent hidden state h_k gotten from RNNs to denote user short-term preference. This is suboptimal since past hidden states are not fully employed in determining how to compute the conditional intensities. Intuitively, check-ins occur in different time steps have different degrees of importance for representing the whole user trajectory and affecting the happening of future check-ins. In this work, a simple but effective attention computation is proposed to automatically quantify the effect of past check-in events. To achieve this, we treat current hidden state h_k as the query vector and all the hidden states $\{h_{k'}\}_{k'=1}^{k}$ as candidate vectors. The attention computation is formulated as follows:

$$\text{Attention}(h_k, h_{k'}) = \text{Softmax}(\frac{h_k \cdot h_{k'}}{\sqrt{d_h}}), \tag{10}$$

where $\sqrt{d_h}$ is the scaling factor. We set d_h to be same as the dimension of GRU based hidden vectors according to [32]. Based on the attention weights, we get the integrated representation $\tilde{h}_k$ defined to be:

$$\tilde{h}_k = \sum_{k'=1}^{k} \text{Attention}(h_k, h_{k'}) \cdot h_{k'}. \tag{11}$$

On this basis, we can formulate the conditional intensity function for the next check-in time prediction by:

$$\lambda^*(t) = \exp(v_T^{\mathrm{T}} \cdot \tilde{h}_k + w_U \cdot v_u^* + w_T \cdot g_k + b_T), \tag{12}$$

where v_T, w_U, w_T, and b_T are trainable parameters. $g_k = t - t_k$ is the time interval since the last check-in. $\tilde{h}_k$ is the latent representation w.r.t the current step, accounting for the influence from all previous check-ins in the trajectory. v_u^* is the user long-term representation gotten from the last layer of GAT. The third term measures the current influence by the k-th check-in, and the last term b_t stands for the base intensity. The outside exponential function plays the role of non-linear transformation and guarantees that the intensity value is always positive [7].

Finally, by replacing the conditional density function in Equation 2 with Equation 12, we can derive the full expression of the proposed deep point process by the following equation:

$$\begin{aligned} f^*(t) = \exp\Big\{ & v_T^{\mathrm{T}} \cdot \tilde{h}_k + w_U \cdot v_u^* + w_T \cdot g_k + b_T \\ & + \frac{1}{w_T} \exp\left(v_T^{\mathrm{T}} \cdot \tilde{h}_k + w_U \cdot v_u^* + b_T\right) \\ & - \frac{1}{w_T} \exp\left(v_T^{\mathrm{T}} \cdot \tilde{h}_k + w_U \cdot v_u^* + w_T \cdot g_k + b_T\right) \Big\}. \end{aligned} \tag{13}$$

The expected return time of next check-in $\hat{t}_{k+1}$ can be computed as follows:

$$\hat{t}_{k+1} = \int_{t_k}^{\infty} t \cdot f^*(t) dt. \tag{14}$$

However, the above integration does not have an analytic solution due to the complex form shown in Equation 13. As an alternative strategy, numerical integration techniques [30] are commonly used for one-dimensional functions to approximate the expectation computation.

2) Check-in location prediction: In the right part of the multi-task prediction module, we try to predict the next check-in location. We first combine the user short-term preference with the user long-term preference through the following way:

$$O_i = W_i[h_k; v_u^*] + b_i, \tag{15}$$

where $[h_k; v_u^*]$ is the concatenation of the two types of user representations. W_i and b_i are the trainable parameters associated with the i-th location. Then we apply the Softmax operation to those representations $O = \{O_1, \dots, O_i, \dots, O_{|L|}\}$ and the i-th dimension can be seen as the probability of visiting the corresponding location. The maximum element is selected as the most possible location l_{k+1}.

D. Multi-task Training

We integrate both the tasks of location prediction and time prediction into a unified multi-task learning framework. The loss for the check-in time prediction can be specified as:

$$\mathcal{L}_T(t_{k+1}) = -\log f^*(t_{k+1}). \tag{16}$$

The loss for the location prediction task is denoted as:

$$\mathcal{L}_{Loc}(l_{k+1}) = -y_{k+1} \log(P_{k+1}), \tag{17}$$

where y_{k+1} is the one-hot encoding of the target location and $P_{k+1} = \text{Softmax}(O)$ is the probability distribution w.r.t. each location.

The final objective function is the sum of the time prediction loss and the location prediction loss, which is formulated as follows:

$$\mathcal{L} = \sum_{u \in U} \sum_{S_u \in C_u} \sum_{k=1}^{j-1} \mathcal{L}_{Loc}(l_{k+1}) + \beta \mathcal{L}_T(t_{k+1}) + \gamma \|\Theta\|_2^2, \tag{18}$$

where β and γ are the hyperparameters used for tuning relative influence of the time prediction loss and the regularization loss of model parameters (denoted by Θ). The whole framework is trained using back-propagation in an end-to-end fashion.

V. Experiments

A. Datasets

We evaluate different methods based on two public real datasets, which contain user check-in records and social relations from two different LSNs.

Gowalla[1]: This most widely used dataset was collected from Gowalla [5]. The raw dataset contains 6,442,890 check-ins reported by 196,591 users distributed around the world. Each check-in is composed of user ID, a location ID and its corresponding GPS coordinate, and a timestamp. Since the dataset is large, sparse, and globally distributed, we first restrict the geographical area to the city of Los Angles (LA) according to its coordinates. After that, we filter out the locations with less than 5 visits and segment the whole trajectory sequence into several trajectories based on the time interval between two neighbor check-ins. To avoid the length of a trajectory to be too long, we also constrain the maximal length of a trajectory. Furthermore, we filter out these trajectories with less than 3 check-ins and users with less than 3 trajectories. We build the user social graph on the remaining users. The check-in time interval threshold ζ is set to 72 hours, due to the assumption that two successive check-ins with a time interval larger than 72 hours are usually irrelevant to each other.

Foursquare[2]: It was collected from Foursquare [43], the records of which ranges from Apr. 2012 to Jan. 2014 (about 22 months). We extract user check-ins within New York City (NYC) and apply the same preprocessing procedures with different settings. We filter out users with less than 10 check-ins. Table I summarizes the statistics of the filtered datasets.

To perform an evaluation, we divide each dataset into its respective training set, validation set, and test set based on users. That is to say, the earliest 70% trajectories of each user are selected as the training data, the following 20% as the validation data, and the remaining 10% as the test data.

TABLE I: **Statistics of the experiment datasets**

	Gowalla-LA	Foursquare-NYC
# Users	716	1344
# Locations	5871	5771
# Check-ins	42273	73960
# User relations	682	1606
# Trajectories	7919	18806
# Sparsity	98.99%	99.04%
Avg. trajectory/user	11.06	13.99
Avg. friend/user	0.9525	1.1949
Avg. trajectory length	5.34	3.93

B. Evaluation Metrics

To evaluate model performance on the location prediction task, the following two metrics are adopted: Recall and Mean Reciprocal Rank (MRR). Given the top-n returned prediction list for user u and its testing trajectory S_u, these two metrics reflect different aspects of the results: recall measures the number of correct predictions in the result, while MRR considers the rank of the prediction. Recall is computed as:

$$\text{Recall@}n = \frac{1}{|U|}\sum_{u\in U}\frac{1}{|S_u|}\sum_{c_{u,k}\in S_u}\mathbb{I}(rank_{lg(u,k)} \leq n), \quad (19)$$

where $lg(u,k)$ denotes the k-th ground truth location for trajectory S_u, $rank_{lg(u,k)}$ refers to the rank position generated by the model, and $\mathbb{I}(\cdot)$ is an indicator function. Since Recall@n does not differentiate the positions within the top-n list, we adopt another metric MRR as a complement:

$$\text{MRR} = \frac{1}{|U|}\sum_{u\in U}\frac{1}{|S_u|}\sum_{c_{u,k}\in S_u}\frac{1}{rank_{lg(u,k)}}. \quad (20)$$

Since each prediction in a trajectory of the test set has only one ground-truth location, Recall@n is equivalent to Hit Rate@n and proportional to Precision@n. In this work, we report Recall@n ($n \in \{1,3,5,10\}$) and MRR@10 ($rank_{lg(u,k)} \leq 10$). The larger values of these metrics indicate better performance.

For the time interval prediction task, we adopt Mean Square Error (MSE) which is suitable for measuring of the difference between the two continuous values, i.e., ground-truth t_k and predicted $\hat{t}_k$,

$$\text{MSE} = \frac{1}{|U|}\sum_{u\in U}\frac{1}{|S_u|}\sum_{c_{u,k}\in S_u}\left(\hat{t}_k - t_k\right)^2. \quad (21)$$

C. Baselines

We compare our approach with several recently proposed competitive methods for next check-in location prediction and time prediction, respectively.

1) Next check-in location prediction task:

- Markov model [24]: It is a simple baseline widely used to predict future locations. It calculates the first-order transition probabilities in a transition matrix consisting of all the visited locations.
- RMTPP [7]: It uses a recurrent point process to model the event sequence data. The main issues of this model have been illustrated in Section I.
- CARA [23]: This is a GRU based sequential modeling architecture in the location prediction task, which utilizes the temporal and spatial data to create the contextual attention gate and the spatio-temporal gate. We utilize the released code of CARA cell[3].
- DeepMove [10]: This is a particularly designed attention-based recurrent model for human mobility prediction. It relies on a multi-modal embedding module to transform the available data into dense representations. We redevelop our training trajectories to cater to the paper settings that distinguish current trajectory and historical trajectories, while the testing trajectories remain unchanged. We implement the whole model along with the sequential encoding module by utilizing the released code[4].
- LOCALBAL [31]: This model integrates ratings with local and global social relations to capture user preference

[1]https://snap.stanford.edu/data/loc-gowalla.html

[2]https://sites.google.com/site/yangdingqi/home/foursquare-dataset

[3]https://github.com/feay1234/CARA

[4]https://github.com/vonfeng/DeepMove

TABLE II: **Method comparison on the location prediction task.**

Method	Gowalla-LA					Foursquare-NYC				
	Rec@1	Rec@3	Rec@5	Rec@10	MRR@10	Rec@1	Rec@3	Rec@5	Rec@10	MRR@10
Markov	0.08077	0.13520	0.16066	0.19051	0.11385	0.12721	0.23710	0.26345	0.29469	0.18441
DeepMove	0.12655	0.22577	0.26252	0.30552	0.18355	0.18154	0.29357	0.32015	0.35628	0.24255
CARA	0.12467	0.22827	0.26076	0.29236	0.18092	0.17915	0.29131	0.32405	0.34889	0.24178
RMTPP	0.09131	0.14925	0.17471	0.20369	0.12589	0.15770	0.24313	0.26760	0.29206	0.20417
LOCALBAL	0.12467	0.22037	0.26514	0.30817	0.18238	0.18218	0.29281	0.32556	0.37185	0.24685
eSMF	0.12467	0.22476	0.26163	0.30641	0.18292	0.18555	0.29770	0.32593	0.36696	0.24821
JNTM	0.11819	0.21188	0.25476	0.29884	0.17388	0.17087	0.27136	0.31351	0.34964	0.23005
PRPPA	0.13257	0.22994	0.26778	0.31343	0.19010	0.18329	0.29755	0.33195	0.37147	0.24665
ARNPP-GAT-NN	0.12905	0.22577	0.26452	0.30328	0.18512	0.18034	0.29281	0.32446	0.36527	0.24489
ARNPP-GAT	**0.13433**	**0.23739**	**0.27129**	**0.32748**	**0.19558**	**0.18743**	**0.30773**	**0.34738**	**0.39067**	**0.25564**

TABLE III: **Method comparison on the time prediction task. Unit: 4h.**

Method	Gowalla-LA	Foursquare-NYC
Avg	47.39	58.93
RMTPP	33.69	38.45
RSTPP	32.33	33.54
THP	31.82	32.67
PRPPA	31.40	31.88
ARNPP-GAT-NN	32.52	32.45
ARNPP-GAT	**29.69**	**30.49**

correlation. We incorporate the contribution from social relations by adding this term to our objective function.

- eSMF [15]: By considering how much user behavior would be affected by its direct friends, eSMF is further improved over LOCALBAL by exploiting the graph structure of neighbors. In other words, it calculates a trust value for each friend to determine its importance.
- JNTM [42]: It is a neural network based approach to jointly model social networks and mobile trajectories. It models four key factors: user visit preference, social relation influence, long and short-term sequential contexts.

2) Next check-in time prediction task:

- Average time (Avg): This method returns the average time interval of historical check-ins as the predicted next check-in time interval.
- RMTPP [7]: This model is capable of predicting the next check-in time. Its intensity function considers the current hidden state h_k from a recurrent neural network.
- RSTPP [44]: It proposes a recurrent spatio-temporal point process to predict check-in time. The spatial distance impact is also grasped in the proposed intensity function.
- THP [54]: It is the latest point process model that couples the ideas of transformer networks (self-attention mechanism) with Hawkes process to fit event sequence data. Thus it enjoys the ability to model long sequences.

Finally, we summarize the proposed models as follows:

- PRPPA: This is our approach proposed in the short version [19] of this paper, which does not consider modeling social relations.
- ARNPP-GAT-NN: This method shares the same multi-task setting with a normal regression loss for the time prediction task. The corresponding time prediction results are shown in Table III.
- ARNPP-GAT: It is the full model proposed in this paper, which fuses temporal and spatial aspects of check-ins, as well as learning to encode social relations.

D. Hyperparameter Setting

We implement all the baselines and our proposed model using Tensorflow. We select the hyper-parameters for both datasets as follows. (1) We set the dimension of the latent factors and hidden state size to 10 for all methods, and the user embedding size is set to 64. (2) We use Gaussian distribution (with the mean to be 0 and the standard deviation to be 0.01) to randomly initialize all the embeddings and parameters. (3) Adam [16] optimizer is exploited to learn model parameters in a mini-batch fashion. (4) We execute the whole GATs updating procedure for every 5 epochs. (5) We choose the initialized learning rate from $\{0.001, 0.005, 0.01\}$ and set the batch size to 64. (6) Hyperparameters β and τ are tuned by performance reference on the validation sets. (7) L2 regularization (l2 norm with the coefficient to be 0.0001) and Dropout strategy (with the dropout rate to be 0.5) are employed to alleviate the overfitting issue. Detailed analysis of hyper-parameters on the final performances is examined below in Section V-G.

E. Method Comparison

Location Prediction Performance: Table II shows the experimental results in Gowalla and Foursquare on the location prediction task. From a whole perspective, the Markov model performs the worst, which is consistent with our expectation since it only considers the first-order transition probabilities. The RMTPP model shows some improvements over the Markov model in terms of Recall, since it captures more sequential information by recurrent units in the model. However, without the help of user representation modeling, these two models are not competitive with the other models. Both DeepMove and CARA introduce user representations into their models. Specifically, DeepMove uses a multi-modal embedding module to transform the temporal data into dense vectors, while CARA incorporates time interval and geographical distance value into a spatio-temporal gate in recurrent modeling. As we can see, DeepMove outperforms CARA for both datasets

TABLE IV: **Component analysis on both tasks.**

Architecture	Gowalla-LA						Foursquare-NYC					
	Rec@1	Rec@3	Rec@5	Rec@10	MRR@10	MSE	Rec@1	Rec@3	Rec@5	Rec@10	MRR@10	MSE
ARNPP-GAT (T-LOC)	0.12379	0.21686	0.26327	0.30992	0.18543	N/A	0.18216	0.30147	0.33513	0.37938	0.25052	N/A
ARNPP-GAT (T-TIME)	N/A	N/A	N/A	N/A	N/A	33.25	N/A	N/A	N/A	N/A	N/A	32.69
PRPPA	0.13257	0.22994	0.26778	0.31343	0.19010	31.40	0.18329	0.29755	0.33195	0.37147	0.24665	31.88
ARNPP-GAT w/o user	0.12467	0.22218	0.26339	0.32133	0.18813	31.15	0.18536	0.30222	0.33722	0.38653	0.25326	30.65
ARNPP-GAT w/o att	0.12818	0.22354	0.26563	0.32485	0.19012	33.74	0.18329	0.30034	0.34136	0.38553	0.25166	32.10
ARNPP-GAT	**0.13433**	**0.23739**	**0.27129**	**0.32748**	**0.19558**	**29.69**	**0.18743**	**0.30773**	**0.34738**	**0.39067**	**0.25564**	**30.49**

in most cases. The reason might be that the trajectory history module provides some valuable information.

Compared with the above models, PRPPA considers the spatial and temporal factors from the multi-task learning perspective and achieves a higher performance. This phenomenon also makes sense as more supervised signals from the next check-in time prediction task are injected into the model optimization process through back-propagation. The final model ARNPP-GAT is an extension of PRPPA by learning from social relations to improve user long-term representations. The better performance of ARNPP-GAT over PRPPA shows that incorporating social relation learning is indeed beneficial. We also consider the variant, i.e., ARNPP-GAT-NN, to show that using neural point process is better than conventional fully connected networks for joint prediction. In addition, the middle region in Table II covers several location prediction models with social relation learning. By comparing ARNPP-GAT with these models, we could observe the performance advantage of ARNPP-GAT.

Time Prediction Performance: Table III shows the performance of all the methods on the time prediction task. We can observe Avg performs the worst since no check-in related information has been considered. RSTPP has relatively better performance than RMTPP since it considers user long-time preference and spatial distance impact. In our study, however, the performance gained by simply incorporating the geographical distance through a linear transformation in the intensity function is relatively low. THP is a strong baseline that considers the impact of past events on current predictions through transformer networks, which further validates the idea of explicitly modeling the impact of past events in the intensity function. The ARNPP-GAT-NN approach does not perform so well for time prediction, revealing the fact that temporal point process is a more elegant mechanism to model temporal sequences. Generally speaking, PRPPA has considerable improvements on both datasets over all the baselines, due to the multi-task learning and the incorporation of attention mechanism in conditional intensity function. Thanks to introducing social relations, the final model ARNPP-GAT learns better user long-term representations than PRPPA and gets the best performance.

F. Ablation Study

To verify the rationality of the main components in ARNPP-GAT, we consider its several variants to test their contributions to both tasks. We first use “ARNPP-GAT (T-LOC)” to denote only considering the location prediction task and “ARNPP-GAT (T-TIME)” to represent only focusing on the time prediction task, respectively. From Table IV we can easily observe that their performance is worse than the full model ARNPP-GAT, indicating the significance of multi-task learning for better prediction performance. By comparing PRPPA with “ARNPP-GAT (T-LOC)”, the performance is better on Gowalla but worse on Foursquare. This reveals that multi-task learning and social relation learning contribute differently to the two datasets.

Furthermore, we consider removing user representations from the conditional intensity function to check how it affects the performance and name this variant as “ARNPP-GAT w/o user”. The results show it is crucial for both tasks to achieve better results. Last but not least, we validate the benefit of explicitly capturing the impact of all past check-in events in the same trajectory to the generation of future check-in events. “ARNPP-GAT w/o att” erases the attention based computation from the intensity function. The corresponding results reveal that the above idea contributes to both tasks, especially for the time prediction part.

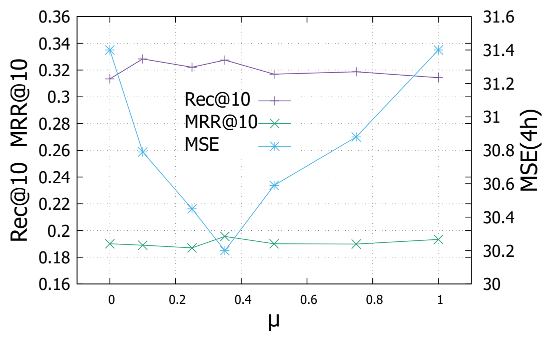
G. Result Analysis of Hyperparameters

TABLE V: **Results of layer number (z) for propagation.**

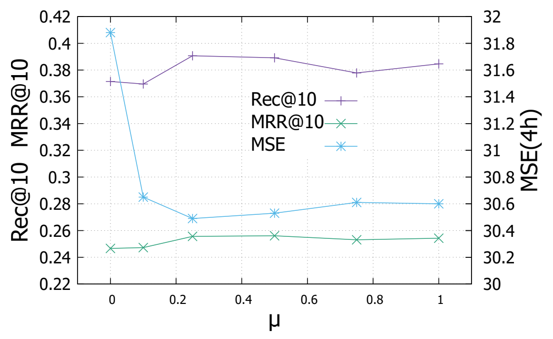
ARNPP-GAT	Gowalla-LA		Foursquare-NYC	
	Rec@10	MRR@10	Rec@10	MRR@10
z=0	0.31343	0.19010	0.37147	0.24665
z=1	0.32094	0.19357	0.37900	0.25490
z=2	**0.32748**	**0.19558**	**0.39067**	**0.25564**
z=3	0.32046	0.18567	0.38328	0.25402

1) Effect of layer number in GATs: Table V shows the location prediction performance with different numbers of GATs layers. We observe that stacking GATs layers can boost performances especially when $z = 2$. This demonstrates learning high-order user relations via representation learning is indeed beneficial. The results decline as the model goes deeper, which might be caused by the over-smoothing issue of GNNs.

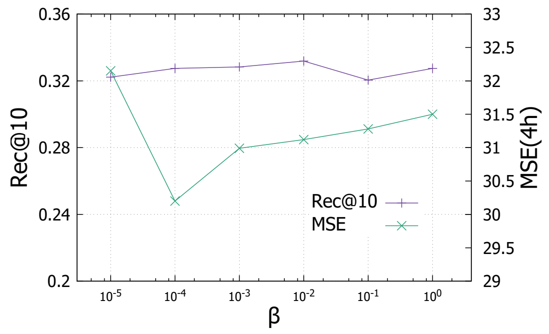
2) Effect of user embedding dimension and head number: We now study how the combination of user embedding dimension and head number affects the prediction performance. Table VI shows the results varying with user embedding dimension and head number while keeping other optimal hyper-parameters unchanged. When the user dimension and the number are set to 64 and 8, respectively, the performance is good for gaining the best results in half of the cases.



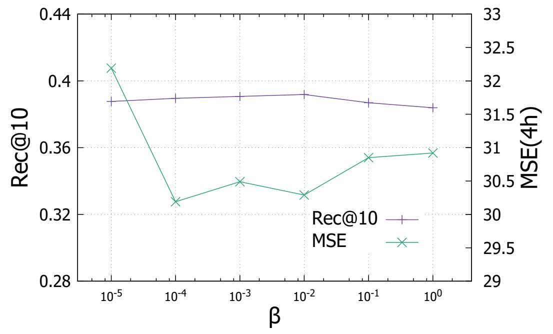
(a) Gowalla-LA



(b) Foursquare-NYC



(c) Gowalla-LA



(d) Foursquare-NYC

Fig. 3: Effect of hyperparameters μ and β.

TABLE VI: **Results of different user embedding dimensions and number of heads.**

ARNPP-GAT		Gowalla-LA		Foursquare-NYC	
U_dim	N_head	Rec@10	MRR@10	Rec@10	MRR@10
16	4	0.29939	0.17337	0.35943	0.24503
32	4	0.31519	0.18838	0.38163	0.25489
32	8	**0.33099**	0.18803	0.37862	0.25500
64	4	0.32221	0.19379	0.38239	0.25316
64	**8**	0.32748	**0.19558**	**0.39067**	0.25564
64	16	0.32924	0.19410	0.38766	**0.25680**

3) Effect of hyperparameter μ: We first study the impact of hyperparameter μ by changing its value from 0 to 1. Note that when μ equals to 0, our model degenerates to PRPPA. As shown in Figure 3(a) and 3(b), the overall performance increases first and then gets a little drop as value gets larger in both Rec@10 and MRR@10. As for the time prediction task, the best μ is around 0.35 for Gowalla and 0.25 for Foursquare. The shapes of the curves corresponding to MSE again demonstrate the advantage of learning social relations for time prediction.

4) Effect of hyperparameter β.: To investigate how L_t influences the prediction performance, we set it to different values ranging from 10^{-5} to 10^{0}. As the curves shown in Figure 3(c) and 3(d), ARNPP-GAT obtains a little better results in both Rec@10 and MRR@10 when β becomes larger in a small numerical range. After β surpasses some thresholds, the performance becomes worse to some extent. Specifically, Gowalla finds its descent performance around 10^{-4}, while Foursquare reaches to the peak at 10^{-2}.

VI. Conclusion

In this paper, we have addressed the next check-in location and time prediction from a multi-task learning perspective. Inspired by the power of deep recurrent modeling and temporal point process, we have proposed a novel model named ARNPP-GAT, a natural extension of RMTPP by: 1) introducing user representation learned from GATs to denote user long-term preference and combining it with user short-term preference gained by sequential modeling; 2) proposing an attention based method to explicitly capture the effect of past check-in events, along with user representations, for the generation of next check-in event. Comprehensive experiments on two real datasets have shown ARNPP-GAT is superior among all the adopted methods for both tasks and demonstrated the significance of the main components in the model architecture.

References

[1] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh. Vqa: Visual question answering. In *ICCV*, pages 2425–2433, 2015.
[2] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.
[3] B. Chang, Y. Park, D. Park, S. Kim, and J. Kang. Content-aware hierarchical point-of-interest embedding model for successive poi recommendation. In *IJCAI*, pages 3301–3307, 2018.
[4] C. Cheng, H. Yang, M. R. Lyu, and I. King. Where you like to go next: Successive point-of-interest recommendation. In *IJCAI*, pages 2605–2611, 2013.
[5] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *SIGKDD*, pages 1082–1090, 2011.
[6] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[7] N. Du, H. Dai, R. Trivedi, U. Upadhyay, M. Gomez-Rodriguez, and L. Song. Recurrent marked temporal point processes: Embedding event history to vector. In *SIGKDD*, pages 1555–1564, 2016.

[8] J. Durbin and G. S. Watson. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

[9] T. Evgeniou and M. Pontil. Regularized multi–task learning. In *SIGKDD*, pages 109–117, 2004.

[10] J. Feng, Y. Li, C. Zhang, F. Sun, F. Meng, A. Guo, and D. Jin. Deepmove: Predicting human mobility with attentional recurrent networks. In *WWW*, pages 1459–1468, 2018.

[11] S. Feng, X. Li, Y. Zeng, G. Cong, Y. M. Chee, and Q. Yuan. Personalized ranking metric embedding for next new POI recommendation. In *IJCAI*, pages 2069–2075, 2015.

[12] F. Figueiredo, B. Ribeiro, J. M. Almeida, and C. Faloutsos. Tribeflow: Mining & predicting user trajectories. In *WWW*, pages 695–706, 2016.

[13] A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *ICASSP*, pages 6645–6649, 2013.

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

[15] G.-N. Hu, X.-Y. Dai, Y. Song, S.-J. Huang, and J.-J. Chen. A synthetic approach for recommendation: combining ratings, social relations, and reviews. In *IJCAI*, pages 1756–1762, 2015.

[16] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[17] J. F. C. Kingman. *Poisson processes*, volume 3. Clarendon Press, 1992.

[18] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017.

[19] W. Liang, W. Zhang, and X. Wang. Deep sequential multi-task modeling for next check-in time and location prediction. In *DASFAA*, pages 353–357, 2019.

[20] Q. Liu, S. Wu, L. Wang, and T. Tan. Predicting the next location: A recurrent model with spatial and temporal contexts. In *AAAI*, pages 194–200, 2016.

[21] X. Liu, J. Yan, S. Xiao, X. Wang, H. Zha, and S. M. Chu. On predictive patent valuation: Forecasting patent citations and their types. In *AAAI*, pages 1438–1444, 2017.

[22] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King. Recommender systems with social regularization. In *WSDM*, pages 287–296, 2011.

[23] J. Manotumruksa, C. Macdonald, and I. Ounis. A contextual attention recurrent architecture for context-aware venue recommendation. In *SIGIR*, pages 555–564, 2018.

[24] W. Mathew, R. Raposo, and B. Martins. Predicting future locations with hidden markov models. In *Ubicomp*, pages 911–918, 2012.

[25] H. Mei and J. M. Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *NIPS*, pages 6754–6764, 2017.

[26] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur. Recurrent neural network based language model. In *INTERSPEECH*, 2010.

[27] A. Noulas, S. Scellato, C. Mascolo, and M. Pontil. An empirical study of geographic user activity patterns in foursquare. In *ICWSM*, 2011.

[28] Z. Qiao, S. Zhao, C. Xiao, X. Li, Y. Qin, and F. Wang. Pairwise-ranking based collaborative recurrent neural networks for clinical event prediction. In *IJCAI*, pages 3520–3526, 2018.

[29] J. Qiu, J. Tang, H. Ma, Y. Dong, K. Wang, and J. Tang. Deepinf: Social influence prediction with deep learning. In *SIGKDD*, pages 2110–2119, 2018.

[30] M. C. Seiler and F. A. Seiler. Numerical recipes in c: the art of scientific computing. *Risk Analysis*, 9(3):415–416, 1989.

[31] J. Tang, X. Hu, H. Gao, and H. Liu. Exploiting local and global social context for recommendation. In *IJCAI*, pages 2712–2718, 2013.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[33] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *ICLR*, 2018.

[34] W. Wang, W. Zhang, S. Liu, Q. Liu, B. Zhang, L. Lin, and H. Zha. Beyond clicks: Modeling multi-relational item graph for session-based target behavior prediction. In *WWW*, pages 3056–3062, 2020.

[35] W. Wang, W. Zhang, J. Wang, J. Yan, and H. Zha. Learning sequential correlation for user generated textual content popularity prediction. In *IJCAI*, pages 1625–1631, 2018.

[36] Y. Wang, N. Du, R. Trivedi, and L. Song. Coevolutionary latent feature processes for continuous-time user-item interactions. In *NIPS*, pages 4547–4555, 2016.

[37] Y. Wang, Y. Yuan, Y. Ma, and G. Wang. Time-dependent graphs: Definitions, applications, and algorithms. *DSE*, 4(4):352–366, 2019.

[38] Q. Wu, C. Yang, H. Zhang, X. Gao, P. Weng, and G. Chen. Adversarial training model unifying feature driven and point process perspectives for event popularity prediction. In *CIKM*, pages 517–526, 2018.

[39] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan. Session-based recommendation with graph neural networks. In *AAAI*, pages 346–353, 2019.

[40] Y. Wu, D. Lian, S. Jin, and E. Chen. Graph convolutional networks on user mobility heterogeneous graphs for social relationship inference. In *IJCAI*, pages 3898–3904, 2019.

[41] S. Xiao, J. Yan, X. Yang, H. Zha, and S. M. Chu. Modeling the intensity function of point process via recurrent neural networks. In *AAAI*, pages 1597–1603, 2017.

[42] C. Yang, M. Sun, W. X. Zhao, Z. Liu, and E. Y. Chang. A neural network approach to jointly modeling social networks and mobile trajectories. *TOIS*, 35(4):36, 2017.

[43] D. Yang, B. Qu, J. Yang, and P. Cudre-Mauroux. Revisiting user mobility and social relationships in lbsns: a hypergraph embedding approach. In *WWW*, pages 2147–2157, 2019.

[44] G. Yang, Y. Cai, and C. K. Reddy. Recurrent spatio-temporal point process for check-in time prediction. In *CIKM*, pages 2203–2211, 2018.

[45] G. Yang, Y. Cai, and C. K. Reddy. Spatio-temporal check-in time prediction with recurrent neural network based survival analysis. In *IJCAI*, pages 2976–2983, 2018.

[46] M. Ye, P. Yin, W. Lee, and D. L. Lee. Exploiting geographical influence for collaborative point-of-interest recommendation. In *SIGIR*, pages 325–334, 2011.

[47] H. Yin, Y. Sun, B. Cui, Z. Hu, and L. Chen. LCARS: a location-content-aware recommender system. In *SIGKDD*, pages 221–229, 2013.

[48] T. Zhang, B. Liu, D. Niu, K. Lai, and Y. Xu. Multiresolution graph attention networks for relevance matching. In *CIKM*, pages 933–942, 2018.

[49] W. Zhang and J. Wang. Location and time aware social collaborative retrieval for new successive point-of-interest recommendation. In *CIKM*, pages 1221–1230, 2015.

[50] Z. Zhang, P. Luo, C. C. Loy, and X. Tang. Facial landmark detection by deep multi-task learning. In *ECCV*, pages 94–108, 2014.

[51] Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, and J. Leskovec. Seismic: A self-exciting point process model for predicting tweet popularity. In *SIGKDD*, pages 1513–1522, 2015.

[52] H. Zhou, T. Young, M. Huang, H. Zhao, J. Xu, and X. Zhu. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*, pages 4623–4629, 2018.

[53] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*, 2018.

[54] S. Zuo, H. Jiang, Z. Li, T. Zhao, and H. Zha. Transformer hawkes process. *ICML*, 2020.

Wenwei Liang received the B.Sc. degree in computer science and technology from East China Normal University, Shanghai, China, in 2017. He is currently pursuing his master degree with the Department of Computer Science and Technology, East China Normal University, Shanghai. His main research area is sequential user behavior modeling.

Wei Zhang received his Ph.D. degree in computer science and technology from Tsinghua University, Beijing, China, in 2016. He is currently an associate researcher in the School of Computer Science and Technology, East China Normal University, and also with the MOE Key Lab of Artificial Intelligence, Shanghai Jiao Tong University. His research interests mainly include data mining and machine learning applications, especially for user generated data modeling.