Calendar Graph Neural Networks for Modeling Time Structures in Spatiotemporal User Behaviors

Daheng Wang[1], Meng Jiang[1], Munira Syed[1], Oliver Conway[2], Vishal Juneja[2]
Sriram Subramanian[2], Nitesh V. Chawla[1,3]
[1]University of Notre Dame, Notre Dame, IN 46556, USA
[2]Condé Nast, New York, NY 10007, USA
[3]Department of Computational Intelligence, Wrocław University of Science and Technology, Wrocław, Poland
{dwang8,mjiang2,msyed2,nchawla}@nd.edu
{oliver_conway,vishal_juneja,sriram_subramanian}@condenast.com

ABSTRACT

User behavior modeling is important for industrial applications such as demographic attribute prediction, content recommendation, and target advertising. Existing methods represent behavior log as a sequence of adopted items and find sequential patterns; however, concrete location and time information in the behavior log, reflecting dynamic and periodic patterns, joint with the spatial dimension, can be useful for modeling users and predicting their characteristics. In this work, we propose a novel model based on graph neural networks for learning user representations from spatiotemporal behavior data. Our model's architecture incorporates two networked structures. One is a tripartite network of items, sessions, and locations. The other is a hierarchical calendar network of hour, week, and weekday nodes. It first aggregates embeddings of location and items into session embeddings via the tripartite network, and then generates user embeddings from the session embeddings via the calendar structure. The user embeddings preserve spatial patterns and temporal patterns of a variety of periodicity (e.g., hourly, weekly, and weekday patterns). It adopts the attention mechanism to model complex interactions among the multiple patterns in user behaviors. Experiments on real datasets (i.e., clicks on news articles in a mobile app) show our approach outperforms strong baselines for predicting missing demographic attributes.

KEYWORDS

Behavior modeling, Graph neural network, Spatiotemporal pattern

ACM Reference Format:
Daheng Wang, Meng Jiang, Munira Syed, Oliver Conway, Vishal Juneja, Sriram Subramanian, Nitesh V. Chawla. 2020. Calendar Graph Neural Networks for Modeling Time Structures in Spatiotemporal User Behaviors. *In The 26th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (KDD '20), August 23–27, 2020, Virtual Event, CA, USA.* ACM, NY, NY, USA, 9 pages. https://doi.org/10.1145/3394486.3403308

KDD '20, August 23–27, 2020, Virtual Event, CA, USA

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403308

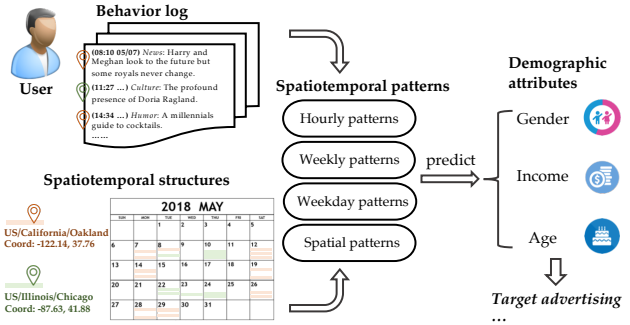


Figure 1: Our framework incorporates calendar structure to model spatiotemporal patterns (including multi-level periodicity) for predicting missing demographic attributes.

1 INTRODUCTION

Online web platforms have large databases to record user behaviors such as reading news articles, posting social media messages, and clicking ads. Behavior modeling is important for a variety of applications such as user categorization [2], content recommendation [18, 33], and targeted advertising [1]. Typical approaches learn users' vector presentations from their behavior log for predicting missing demographic attributes and/or preferred content.

Spatiotemporal patterns in behavior log are reflecting user characteristics and thus expected to be preserved in the vector representations. Earlier work modeled a user's temporal behaviors as a sequence of his/her adopted items and used recurrent neural networks (RNNs) to learn user embeddings [10]. For example, Hidasi *et al.* proposed parallel RNN models to extract features from (sequential) session structures [11]; Tan *et al.* proposed to model temporal shifts in RNNs; and Jannach *et al.* combined RNNs with neighborhood-based methods to capture sequential patterns in user-item co-occurrence [13]. Recently, Graph Neural Networks (GNNs) have attracted increasing interests for learning representations from graph structured data [5, 7, 16, 29]. The core idea is to use convolution or aggregation operators to enhance representation learning through the graph structures [3, 8, 37]. For modeling temporal information in network, Manessi *et al.* [22] stacked RNN modules [12] on top of graph convolution networks [16]; Seo *et al.* [26] replaced fully connected layers in RNNs with graph convolution [5]. However, existing GNNs can only model sequential

patterns or incremental changes in graph series. The spatiotemporal patterns are much more complex in real-world behavior log.

In existing GNN-based user models, the missing yet significant type of patterns is *periodicity* at different levels such as hourly, weekly, and weekday patterns (see Figure 1). For example, some users may have the habit of browsing news articles early in the morning during workdays; some may browse news late at midnight right before sleep. To discover these patterns one needs to process concrete time information beyond simple sequential ordering. So the time levels (or say, the hierarchical structure of calendar) must be incorporated into the process of user embedding learning.

User behaviors exhibit temporal patterns across different periodicities on the time dimension. Our idea is to leverage the explicit scheme of the *Calendar* system for modeling the hierarchical time structures of user behaviors. A standard annual calendar system, e.g., the Gregorian calendar, imposes natural temporal units for timekeeping such as day, week, and month. A daily calendar system imposes more refined temporal units such as hour and minute. These temporal units can naturally be applied to frame temporal patterns. Patterns of various periodicity can be complementary with each other when jointly learned to extract user representations.

In this work, we propose a novel GNN-based model, called CalendarGNN, for modeling spatiotemporal patterns in user behaviors by incorporating time structures of the calendar systems as neural network architecture. It has three aspects of novel designs.

First, a user's behavior log forms a tripartite graph of items, sessions, and locations. In CalendarGNN, session embeddings are aggregated from embeddings of the corresponding items and locations; embeddings of time units (e.g., node "3PM", node "Tuesday", or node "the 15th week of 2018") are aggregated from the session embeddings. The embedding of each time unit captures a certain aspect of the user's temporal patterns. Then the model aggregates these time unit embeddings into temporal patterns of different periodicity such as hourly, weekly, and weekday patterns. The temporal patterns are distilled from all his/her previous sessions happened during the time periods specified by the time unit.

Second, in addition to the temporal dimension, CalendarGNN discovers spatial patterns from spatial signals in user sessions. It aggregates session embeddings into location unit embeddings which can be later aggregated into the user's spatial pattern. The latent user representations are generated by concatenating all temporal patterns and spatial pattern. The user embeddings are used (by classifiers or predictive models) for various downstream tasks.

Third, temporal patterns and spatial patterns should not be separately learned because they interact with each other in user behavior. For example, people may read news at Starbucks in the morning, in restaurants at noon, and at home in the evening; people may prefer different types of topics at different places when they travel to different cities or countries for business. Our model considers the interactions between spatial pattern and the multi-level temporal patterns. We develop a model variant CalendarGNN-Attn that utilizes interactive attentions between location units and different time units for capturing user's complex spatiotemporal patterns.

We conduct experiments on two real-world spatiotemporal behavior datasets (in industry) for predicting user demographic labels (such as gender, age, and income). Results demonstrate the effectiveness of our proposed model compared to existing work.

2 RELATED WORK

We discuss three lines of research related to our work.

Temporal GNNs. The success of GNN on tasks in static setting such as link prediction [37, 41] and node classification [8, 29] motives many work to look at the problem of dynamic graph representation learning. Some deep graph neural methods explored the idea of combining GNN with recurrent neural network (RNN) for leaning node embeddings in dynamic attributed network [22, 26]. These methods aim at modeling the structural evolution among a series of graphs and they cannot be directly applied on users' spatiotemporal graphs for generating behavior patterns. Another set of approaches for spatiotemporal traffic forecasting aim at capturing the evolutionary pattern of node attribute given a fixed graph structure. Li *et al.* [19] modeled the traffic flow as a diffusion process on a directed graph and adopted an encoder-decoder architecture for capturing the temporal attribute dependencies. Yu *et al.* [39] modeled the traffic network as a general graph and employed a fully convolutional structure [5] on time axis. These methods assume the graph structure remains static and model the change of node attributes. They are not designed for capturing the complex time structures among a large number of user spatiotemporal graphs.

Graph-level GNNs. Different from learning node representations, there are some work focus on the problem of learning graph-level representation leveraging node embeddings. A basic approach is applying a global sum or average pooling on all extracted node embeddings as the last layer [6, 27]. Some methods rely on specifying or learning the order over node embeddings so that CNN-based architectures can be applied [24]. Zhang *et al.* [42] proposed a SortPooling layer to take unordered vertex features as input and outputs sorted graph representation of a fixed size in analogous to sorting continuous WL colors [32]. Another way of aggregating node embeddings into graph embedding is learning hierarchical representation through differentiable pooling [38]. Simonovsky *et al.* [27] proposed to perform edge-conditioned convolutions over local graph neighborhoods exploiting edge labels and generate the final graph embedding using a graph coarsening algorithm followed by a global sum pooling layer. These methods are not designed to model user's spatiotemporal behaviors data and cannot explicitly capture the complex time structures of different periodicity.

Session-based user behavior modeling. Hidasi *et al.* [11] proposed a recurrent neural network based approach for modeling users by employing a ranking loss function for session-based recommendations. Tan *et al.* [28] considered temporal shifts of user behavior [40] and incorporated data augmentation techniques to improve the performance of RNN-based model. Jannach *et al.* [13] combined the RNN model with the neighborhood-based method to capture the sequential patterns and co-occurrence signals [14, 15]. Different from these user behavior modeling methods mostly basing on RNN architectures, our framework models each user's behaviors as a tripartite graph of items, sessions and locations, then learns user latent representations via a calendar neural architecture. One recent work by Wu *et al.* [34] models user's session of items as graph structure and use GNN to generate node or item embeddings. However, it is not capable of learning user embeddings. Our work aims at learning effective user representations capturing both the spatial pattern and temporal patterns for different predictive tasks.

Table 1: Symbols and their description.

Symbol	Description
u, s, v, l	a user, a session, an item, and a location
$\mathcal{U}, \mathcal{S}, \mathcal{V}, \mathcal{L}$	set of users, sessions, items and locations
S (S_u)	subset of sessions $\mathcal{S}$ of user u
V (V_u)	subset of items $\mathcal{V}$ of user u
L (L_u)	subset of locations $\mathcal{L}$ of user u
G_u	user u's spatiotemporal behavior graph
E	edge set of G_u
$E^{(L)}$	subset of E containing location-session edges
$E^{(V)}$	subset of E containing item-session edges
$\mathcal{G}$	set of user spatiotemporal behavior graphs
$a_u, \mathcal{A}$	user label, and set of user labels
$\mathcal{B}$	spatiotemporal behavior graph data
$\mathbf{u}, \mathbf{s}, \mathbf{v}, \mathbf{l}$	emb. of user, session, item, and location nodes
$K_{\mathcal{U}}, K_{\mathcal{S}}, K_{\mathcal{V}}, K_{\mathcal{E}}$	dimensions of $\mathbf{u}$, $\mathbf{s}$, $\mathbf{v}$, $\mathbf{l}$ vectors
h_i, w_i, y_i, l_i	hour, week, weekday and location unit of s_i
$\mathcal{T}_h, \mathcal{T}_w, \mathcal{T}_y$	set of temporal units: hour, week, and weekday
$\mathbf{e}_h, \mathbf{e}_w, \mathbf{e}_y, \mathbf{e}_l$	hour, week, weekday, and location unit emb.
$\mathbf{p}_{\mathcal{T}_h}, \mathbf{p}_{\mathcal{T}_w}, \mathbf{p}_{\mathcal{T}_y}, \mathbf{p}_{\mathcal{L}}$	hourly, weekly, weekday, and spatial pattern
$\mathbf{p}^{\mathcal{L}}_{\mathcal{T}_h}, \mathbf{p}^{\mathcal{L}}_{\mathcal{T}_w}, \mathbf{p}^{\mathcal{L}}_{\mathcal{T}_y}$	hourly, weekly, weekday pattern under impacts from spatial pattern
$\mathbf{p}^{\mathcal{T}_h}_{\mathcal{L}}, \mathbf{p}^{\mathcal{T}_w}_{\mathcal{L}}, \mathbf{p}^{\mathcal{T}_y}_{\mathcal{L}}$	spatial patterns under impacts from hourly, weekly, weekday pattern
$\mathbf{p}_{\mathcal{L},\mathcal{T}_h}, \mathbf{p}_{\mathcal{L},\mathcal{T}_y}, \mathbf{p}_{\mathcal{L},\mathcal{T}_w}$	interactive spatial-hourly, spatial-weekly and spatial-weekday patterns

3 PROBLEM DEFINITION

In this section, we first introduce concept of the user spatiotemporal behavior graph then formally define our research problem. The notations used throughout this paper are summarized in Table 1.

A traditional online browsing behavior log contains the transaction records between users and the server. Typically, a user can start multiple sessions and each session is associated with one or more items such as news articles or update feeds. For a spatiotemporal behavior log, in addition to the sessions and items information, there are also corresponding spatial information, e.g., the city or the neighborhood, for each session of the user; and, explicit temporal information, e.g., server timestamp, for each item of the session.

Definition 3.1 (Spatiotemporal Behavior Log). A spatiotemporal behavior log is defined on a set of users $\mathcal{U}$, a set of sessions $\mathcal{S}$, a set of items $\mathcal{V}$, and a set of locations $\mathcal{L}$. For each user $u \in \mathcal{U}$, her behavior log can be represented by a set of session-location tuples $\{(s_{u,1}, l_{u,1}), \ldots, (s_{u,m_u}, l_{u,m_u})\}$ where m_u denotes user u's number of sessions. Each session $s_{u,i}$ comprises a set of item-timestamp tuples $\{(v_{i,1}, t_{i,1}), \ldots, (v_{i,n_i}, t_{i,n_i})\}$ where n_i denotes the number of items in the i-th session of user u.

In a large-scale spatiotemporal behavior log, each user $u \in \mathcal{U}$ is associated with a subset of sessions $S_u \subseteq \mathcal{S}$, a subset of items $V_u \subseteq \mathcal{V}$ have been interacted with, and a subset of locations $L_u \subseteq \mathcal{L}$. Each session $s_{u,i} \in S_u$ is paired with a geographical location signal $l_{u,i} \in L_u$ and each item $v_{i,j} \in V_u$ is paired with an explicit timestamp $t_{i,j}$ forming a behavior entry. To capture the complex temporal and spatial patterns in the spatiotemporal behavior log, we represent a user's behaviors as a tripartite graph structure G_u

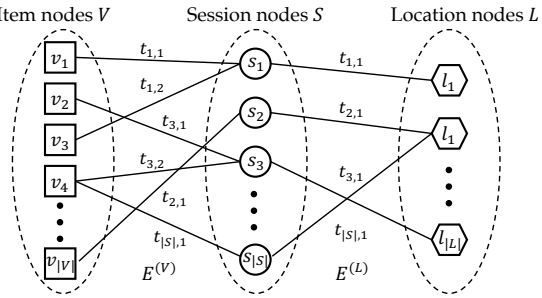


Figure 2: Schematic view of user spatiotemporal behavior graph G_u. This tripartite graph consists of user's sessions S, locations L, and items V as nodes; and, $E^{(V)}$ of session-item edges and $E^{(L)}$ of session-location edges.

as shown in Figure 2. The graph G_u is defined on S_u, L_u and V_u, along with the their corresponding relationships. (Without causing ambiguity, we reduce the subscript u on S_u, L_u and V_u for brevity.)

Definition 3.2 (User Spatiotemporal Behavior Graph). A user u's spatiotemporal behavior graph $G_u = (S, L, V, E)$ includes the user's sessions S, locations L and items V as nodes. There exists an edge $(s_i, l_i) \in E^{(L)} \subseteq E$ between a session node $s_i \in S$ and a location node $l_i \in L$ if the user started the session at this location. And, there exists an edge $(s_i, v_{i,j}) \in E^{(V)} \subseteq E$ between a session node $s_i \in S$ and an item node $v_{i,j} \in V$ if the user interacted with this item within the session. Each edge of E possesses a time attribute indicating the temporal signal of the interaction between two nodes.

The pairing timestamp $t_{i,j}$ ($i < m_u, j < n_i$) for each item in the behavior log can be directly used as the time attribute value for any edge of $E^{(V)}$. For an edge between a session node and a location node of $E^{(L)}$, we use the timestamp of the first item in the session, i.e., the leading timestamp $t_{i,1}$ of the session, as the time attribute value. Note that the subset of edges $E^{(V)}$ describe the many-to-many relationships between the session nodes S and item nodes V, whereas the subset of edges $E^{(L)}$ describe the one-to-many relationships between location nodes L and session nodes S. By modeling each user's behaviors as a spatiotemporal behavior graph G, we are able to format the spatiotemporal behavior log as:

Definition 3.3 (Spatiotemporal Behavior Graph Data). A spatiotemporal behavior graph data $\mathcal{B} = (\mathcal{G}, \mathcal{A})$ represent each user u as a user spatiotemporal behavior graph $G_u = (S, L, V, E) \in \mathcal{G}$, and is related to a specific label $a_u \in \mathcal{A}$ where $\mathcal{A}$ can be categorical or numerical. All user spatiotemporal behavior graphs $\forall G_u \in \mathcal{G}$ share the same sets of sessions $\mathcal{S}$, items $\mathcal{V}$ and locations $\mathcal{L}$.

After we have formatted the spatiotemporal behavior graph data, we can now formally define our research problem as:

Problem: Given a spatiotemporal behavior graph data $\mathcal{B} = (\mathcal{G}, \mathcal{A})$ on a set of users $\mathcal{U}$, **learn** an embedding function f that can map each user $u \in \mathcal{U}$, denoted by her spatiotemporal behavior graph $G_u \in \mathcal{G}$, in to a low-dimensional hidden representation $\mathbf{u}$, i.e., $f : \mathcal{G} \mapsto \mathbb{R}^{K_{\mathcal{U}}}$, where $K_{\mathcal{U}}$ is the dimensionality of vector $\mathbf{u}$ ($K_{\mathcal{U}} << |\mathcal{U}|, |\mathcal{S}|, |\mathcal{V}|, |\mathcal{L}|$). The user embedding vector $\mathbf{u}$ should (1) capture the spatial pattern and temporal patterns of different periodicity in the user's behaviors, and (2) be highly indicative about the corresponding label $a_u \in \mathcal{A}$.

4 THE CALENDARGNN FRAMEWORK

In this section, we present a novel deep architecture CalendarGNN for predicting user attributes by learning user's spatiotemporal behavior patterns. The overall design is shown in Figure 4. We first introduce the item and location embedding layers for embedding the heterogeneous features of item and location nodes in the input user spatiotemporal behavior graph into initial embeddings; then, we present the spatiotemporal aggregation layers as core functions for generating spatial and temporal unit embeddings; next, we describe the aggregation and fusion of different spatial and temporal patterns as user representation, and the subsequent predictive model. At last, to capture the interactions between the spatial pattern and various temporal patterns, we present an enhanced model variant CalendarGNN-Attn that employs an interactive attention mechanism to dynamically adapt importances of different patterns.

4.1 Item and Location Embedding Layers

The inputs into *CalendarGNN* are a user spatiotemporal behavior graphs $G_u = (S, L, V, E)$ and all users $\forall u \in \mathcal{U}$ share the same space of items $\bigcup V = \mathcal{V}$ and locations $\bigcup L = \mathcal{L}$. The first step of *CalendarGNN* is to embed all items $\mathcal{V}$ and locations $\mathcal{L}$ of heterogenous features into their initial embeddings. Figure 3 illustrates the design of the item embedding layer and the location embedding layer.

4.1.1 Item embedding layer. An item $v \in \mathcal{V}$ such as a news article can be described by a group of heterogeneous features: (i) the identification, e.g., the ID of article; (ii) the topic, e.g., the category of article; and, (iii) the content, e.g., the title of the article. For each item, we feed its raw features into the item embedding layer (shown in Figure 3(a)) to generate the initial embedding. Particularly, for categorical features such as the item ID and category, we use *Multilayer Perceptron* (MLP) to embed them into dense hidden representations; and, for textual feature, i.e., the item title, we use *Bidirectional Long Short-Term Memory* (BiLSTM) [25] encoder to generate its hidden representation. Then, the embeddings of different features are concatenated together as the item embedding $\mathbf{v} \in \mathbb{R}^{K_\mathcal{V}}$ where $K_\mathcal{V}$ is the dimensions of the item embedding vector.

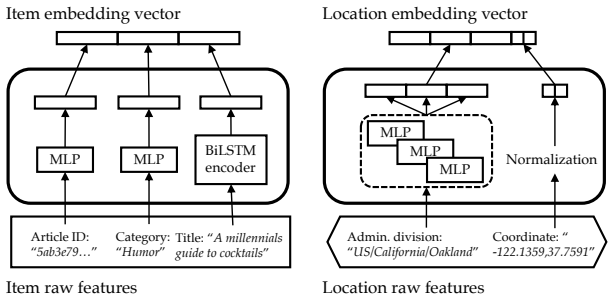
4.1.2 Location embedding layer. Each location $l \in \mathcal{L}$ is denoted by a multi-level administrative division name in the format of "*county/region/city*", and a coordinate point of longitude and latitude. One example location is "*US/California/Oakland*" and its coordinate "*-122.1359, 37.7591*". We use three distinct MLPs to encode the administrative division at different levels which could be partially empty. The outputs are concatenated with normalized coordinates (shown in Figure 3(b)) as the location embedding vector $\mathbf{l} \in \mathbb{R}^{K_\mathcal{E}}$.

4.2 Spatiotemporal Aggregation Layer

After item and location nodes are embedded into initial embeddings, CalendarGNN generates the embeddings of session nodes by aggregating from item embeddings. For a session node $s_i \in S$ in $G_u = (S, L, V, E)$, its embedding vector $\mathbf{s}_i$ is generated by applying an aggregation function Agg_{sess} on all item nodes linked to it:

$$\mathbf{s}_i = \sigma\left(\mathbf{W}_\mathcal{S} \cdot \text{Agg}_{\text{sess}}(\{\mathbf{v}_{i,j} \mid \forall(s_i, v_{i,j}) \in E\}) + \mathbf{b}_\mathcal{S}\right), \quad (1)$$

where σ is a function for non-linearity, such as ReLU [23]; and, $\mathbf{W}_\mathcal{S}$ and $\mathbf{b}_\mathcal{S}$ are parameters to be learned. The weight matrix $\mathbf{W}_\mathcal{S} \in \mathbb{R}^{K_\mathcal{S} \times K_\mathcal{V}}$ transforms the $K_\mathcal{V}$-dim item embedding space to the $K_\mathcal{S}$-dim session embedding space (assuming Agg_{sess} has the same number of input and output dimensions). The aggregation function Agg_{sess} can be arbitrary injective function for mapping a set of vectors into an output vector. Since the session node's neighbor of item nodes $\{v_{i,j} \mid \forall(s_i, v_{i,j}) \in E\}$ can naturally be ordered by their timestamps $t_{i,j}$, we arrange items as sequence and choose to use *Gated Recurrent Unit* (GRU) [4] as the Agg_{sess} function.



(a) Item embedding layer (b) Location embedding layer

Figure 3: The item embedding layer (left) takes raw features of an item, i.e., the ID, category and title, as input and generates its embedding vector; and, the location embedding layer (right) takes the administrative division and coordinate of a location as input and generates its embedding vector.

Now, we have generated session node embeddings $\{\mathbf{s} \mid s \in S\}$ for G_u, CalendarGNN is ready to generate spatial and temporal patterns. The core intuition is to inject external knowledge about the calendar system's structure into the architecture of CalendarGNN so that we can aggregate a user's session node embeddings into spatial pattern and temporal patterns of various periodicity based on their spatial and temporal signals. Specifically, we pass session node embeddings to: (1) the temporal aggregation layer for generating temporal patterns of various periodicity; and, (2) the spatial aggregation layer for generating spatial pattern.

4.2.1 Temporal aggregation layer. Given session node embeddings $\{\mathbf{s} \mid s \in S\}$ of G_u, the idea of temporal aggregations in this layer is to: (1) map sessions S's continuous timestamps into a set of discrete time units, and (2) aggregate sessions of the same time unit into the corresponding time unit embeddings, and, (3) aggregate time unit embeddings into the embedding of temporal pattern.

Mapping sessions S' timestamps $\{t_i \mid s_i \in S\}$ into set of discrete time units is analogous to bucket session embeddings by discrete time units. We regard the leading timestamp of corresponding item nodes as the session's timestamp, i.e., $t_i = \min(\{t_{i,j} \mid \forall(s_i, v_{i,j}) \in E\})$. Particularly, taken inspiration from the daily calendar system, we convert t_i into three types of time units:

- $h_i = hour(t_i) \in \mathcal{T}_h$, where $\mathcal{T}_h$ has 24 distinct values: 0AM, 1AM, ..., 11PM;
- $w_i = week(t_i) \in \mathcal{T}_w$, where $\mathcal{T}_w$ is the set of weeks of the year, e.g., Week 18;
- $y_i = weekday(t_i) \in \mathcal{T}_y$, where $\mathcal{T}_y$ has 7 values: Sunday, Monday, ..., Saturday.

The time unit mapping functions *hour*, *week* and *weekday* takes a timestamp as input and outputs a specific time unit. The cardinality of the output time units set can vary, e.g., $|\mathcal{T}_h| = 24$ or $|\mathcal{T}_y| = 7$.

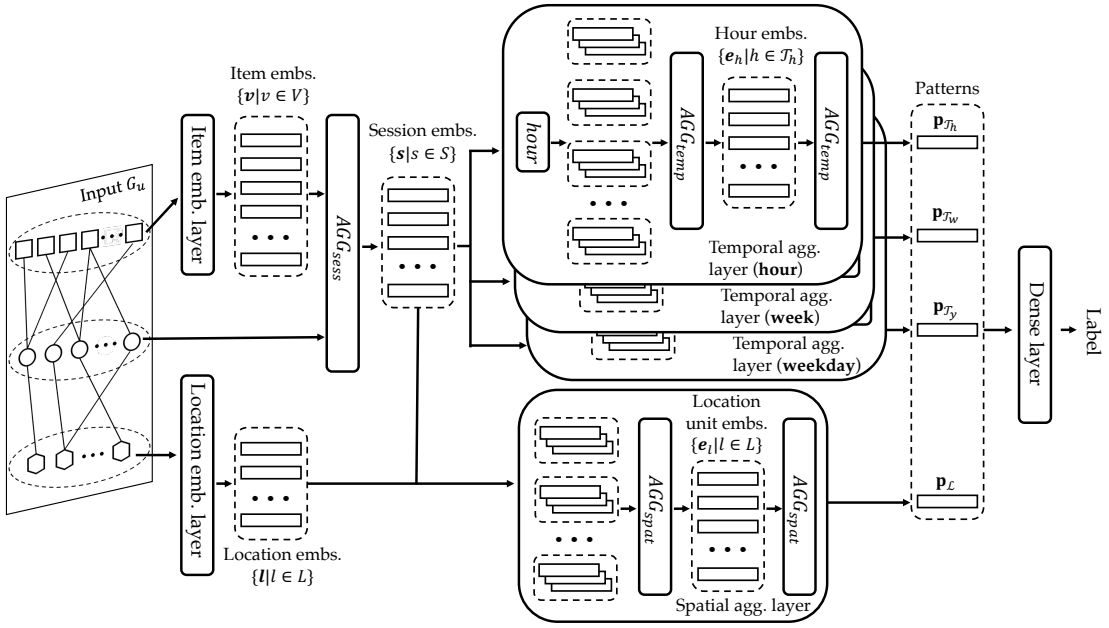


Figure 4: CalendarGNN architecture: Session embeddings are generated by aggregating its item embeddings. The embeddings of sessions are aggregated into hour, week, weekday unit embeddings, and location unit embeddings. Next, embeddings of temporal/spatial units are aggregated into pattern embeddings, and further fused into the user embedding for prediction.

In this work, we leverage 3 time units of common sense, i.e., hour, week, and weekday, for capturing the complex time structures in user behaviors. CalendarGNN maintains the flexibility to model temporal pattern of arbitrary periodicity, such as daytime/night or minute, providing the new time unit mapping function(s).

Once the session nodes are mapped into specified time units, CalendarGNN aggregates the session node embeddings into various time unit embeddings by applying a temporal aggregation function AGG_{temp} on sessions of the same time unit:

$$\mathbf{e}_h = \sigma\left(\mathbf{W}_h \cdot \text{AGG}_{\text{temp}}(\{\mathbf{s}_i \mid h_i = h \in \mathcal{T}_h\}) + \mathbf{b}_h\right), \tag{2}$$

$$\mathbf{e}_w = \sigma\left(\mathbf{W}_w \cdot \text{AGG}_{\text{temp}}(\{\mathbf{s}_i \mid w_i = w \in \mathcal{T}_w\}) + \mathbf{b}_w\right), \tag{3}$$

$$\mathbf{e}_y = \sigma\left(\mathbf{W}_y \cdot \text{AGG}_{\text{temp}}(\{\mathbf{s}_i \mid y_i = y \in \mathcal{T}_y\}) + \mathbf{b}_y\right), \tag{4}$$

where the weight matrices $\mathbf{W}_h \in \mathbb{R}^{K_h \times K_S}$, $\mathbf{W}_w \in \mathbb{R}^{K_w \times K_S}$ and $\mathbf{W}_y \in \mathbb{R}^{K_y \times K_S}$ transform the K_S-dim session embedding space into K_h-dim hour embedding space, K_w-dim week embedding space, and K_y-dim weekday embedding space, respectively. The choice of AGG_{temp} is also set to GRU since all items of the same time unit can naturally be ordered by their raw timestamp.

Next, these time unit embeddings in the three dimensions (i.e., hour, week, and weekday) are further aggregated into embeddings of respective temporal patterns:

$$\mathbf{p}_{\mathcal{T}_h} = \sigma\left(\mathbf{W}_{\mathcal{T}_h} \cdot \text{AGG}_{\text{temp}}(\{\mathbf{e}_h \mid \forall h \in \mathcal{T}_h\}) + \mathbf{b}_{\mathcal{T}_h}\right), \tag{5}$$

$$\mathbf{p}_{\mathcal{T}_w} = \sigma\left(\mathbf{W}_{\mathcal{T}_w} \cdot \text{AGG}_{\text{temp}}(\{\mathbf{e}_w \mid \forall w \in \mathcal{T}_w\}) + \mathbf{b}_{\mathcal{T}_w}\right), \tag{6}$$

$$\mathbf{p}_{\mathcal{T}_y} = \sigma\left(\mathbf{W}_{\mathcal{T}_y} \cdot \text{AGG}_{\text{temp}}(\{\mathbf{e}_y \mid \forall y \in \mathcal{T}_y\}) + \mathbf{b}_{\mathcal{T}_y}\right), \tag{7}$$

where the weight matrices $\mathbf{W}_{\mathcal{T}_h} \in \mathbb{R}^{K_{\mathcal{T}_h} \times K_h}$, $\mathbf{W}_{\mathcal{T}_w} \in \mathbb{R}^{K_{\mathcal{T}_w} \times K_w}$, $\mathbf{W}_{\mathcal{T}_y} \in \mathbb{R}^{K_{\mathcal{T}_y} \times K_y}$ transform the aggregated hour, week, and weekday embeddings into the corresponding ($K_{\mathcal{T}_h}$-dim) hourly, ($K_{\mathcal{T}_w}$-dim) weekly, and ($K_{\mathcal{T}_h y}$-dim) weekday patterns, respectively. Each one of these temporal pattern captures the user's temporal behavior pattern of a specific periodicity.

In addition to temporal patterns, another indispensable aspect of user's behavior pattern relates to the spatial signals of sessions. CalendarGNN is capable of discovering user's spatial pattern by aggregating session embeddings via the spatial aggregation layer.

4.2.2 Spatial aggregation layer. Similar to the treatment of temporal aggregation layer previous introduced, for generating spatial pattern, CalendarGNN first aggregates the session node embeddings into location unit embeddings based on their spatial signals:

$$\mathbf{e}_l = \sigma\left(\mathbf{W}_{\mathcal{S}\times\mathcal{L}} \cdot \text{AGG}_{\text{spat}}(\{\mathbf{s}_i \oplus \mathbf{l}_i \mid l_i = l \in L\} + \mathbf{b}_{\mathcal{S}\times\mathcal{L}}\right), \tag{8}$$

where $\oplus$ is concatenation operator, and $\mathbf{W}_{\mathcal{S}\times\mathcal{L}} \in \mathbb{R}^{K_l \times (K_S + K_\mathcal{E})}$ transforms the concatenated space of session embedding initial location embedding into the location unit embedding space, and AGG_{spat} is the spatial aggregation function. We also arrange sessions of the same location unit by their timestamps and choose to use GRU as AGG_{spat}.

Then, CalendarGNN aggregates various location unit embeddings into the embedding vector of spatial pattern:

$$\mathbf{p}_{\mathcal{L}} = \sigma\left(\mathbf{W}_{\mathcal{L}} \cdot \text{AGG}_{\text{spat}}(\{\mathbf{e}_l \mid \forall l \in L\}) + \mathbf{b}_{\mathcal{L}}\right), \tag{9}$$

where $\mathbf{W}_{\mathcal{L}} \in \mathbb{R}^{K_\mathcal{L} \times K_l}$ transforms the location unit embedding space into the spatial pattern space.

By feeding the session node embeddings into temporal aggregation layers and spatial aggregation layer, CalendarGNN has generated temporal patterns. i.e., $\mathbf{p}_{\mathcal{T}_h}$, $\mathbf{p}_{\mathcal{T}_w}$ and $\mathbf{p}_{\mathcal{T}_y}$, and the spatial pattern, i.e., $\mathbf{p}_{\mathcal{L}}$. At last, CalendarGNN fuses all temporal patterns and spatial pattern into a holistic user latent representation $\mathbf{u}$, and pass it to the subsequent predictive model for prediction and output.

4.3 Fusion of Patterns and Prediction

To get the latent representation of user, we concatenate all temporal patterns and the spatial pattern together:

$$\mathbf{u} = \mathbf{p}_{\mathcal{T}_h} \oplus \mathbf{p}_{\mathcal{T}_w} \oplus \mathbf{p}_{\mathcal{T}_y} \oplus \mathbf{p}_{\mathcal{L}} \in \mathbb{R}^{K_{\mathcal{U}}}, \tag{10}$$

where $K_{\mathcal{U}} = K_{\mathcal{T}_h} + K_{\mathcal{T}_w} + K_{\mathcal{T}_y} + K_{\mathcal{L}}$.

We use a single dense layer as the final predictive model for generating user attribute predictions. The discrepancy between the output of the last dense layer and the target attribute value is measured by the objective function for optimization. Specifically, if the user label $a_u \in \mathcal{A}$ is a categorical value, i.e., the task is multi-class classification (with binary classification as a special case), we employ the following cross-entropy objective function:

$$\mathcal{J} = -\sum_{u \in \mathcal{U}} \sum_{a \in \mathcal{A}} \mathbb{I}_{a_u = a} \cdot \frac{\exp(\mathbf{W}_a \cdot \mathbf{u})}{\sum_{a' \in \mathcal{A}} \exp(\mathbf{W}_a \cdot \mathbf{u})}, \tag{11}$$

where $\mathbf{W}_a \in \mathbb{R}^{K_{\mathcal{U}}}$ is the weight vector for label $a \in \mathcal{A}$ and $\mathbb{I}$ is an indicator function. If the label is a numerical value ($a_u \in \mathbb{R}$), we employ the following objective function for the regression task:

$$\mathcal{J} = -\sum_{u \in \mathcal{U}} (\mathbf{W} \cdot \mathbf{u} - a_u)^2. \tag{12}$$

4.4 Interactive Spatiotemporal Patterns

By utilizing the temporal and spatial aggregation layers, CalendarGNN is able to generate spatial pattern and temporal patterns of different periodicity (Eqn. (5) to (9)). However, there are a few limitations. First, different temporal/spatial unit embeddings are of different importance levels to its pattern and this should be reflected during the pattern generation process. Secondly, there could be rich interactions between the spatial pattern and different temporal patterns. These interactions should be carefully captured by the model and be reflected in the true spatiotemporal patterns [9, 35].

To address these limitations, we propose a model variant that employs an interactive attention mechanism [20] and denote it as CalendarGNN-Attn. It enables interactions between spatial and temporal patterns by summarizing location unit embeddings and a certain type of time unit embeddings into an interactive spatiotemporal pattern. For location unit embeddings $\{\mathbf{e}_l \mid \forall l \in L\}$ and time unit embeddings such as hour embeddings $\{\mathbf{e}_h \mid \forall h \in \mathcal{T}_h\}$, a location query and a temporal query are first generated:

$$\bar{\mathbf{e}}_l = \sum_{l \in L} \mathbf{e}_l / |\mathcal{L}|, \bar{\mathbf{e}}_h = \sum_{h \in \mathcal{T}_h} \mathbf{e}_h / |\mathcal{T}_h|, \tag{13}$$

where $|\cdot|$ denotes the cardinality of the set. On one hand, to consider the impacts from spatial signals on temporal signals, a attention weight vector $\alpha_h^{(\mathcal{L},\mathcal{T}_h)}$ is generated using the location query vector $\bar{\mathbf{e}}_l$ and the temporal unit embeddings $\{\mathbf{e}_h \mid \forall h \in \mathcal{T}_h\}$:

$$\alpha_h^{(\mathcal{L},\mathcal{T}_h)} = \frac{\exp(f(\mathbf{e}_h, \bar{\mathbf{e}}_l))}{\sum_{h \in \mathcal{T}_h} \exp(f(\mathbf{e}_h, \bar{\mathbf{e}}_l))}, \tag{14}$$

where f is a function for scoring the importance of $\mathbf{e}_h$ w.r.t. the location query $\mathbf{e}_l$ and is defined as:

$$f(\mathbf{e}_h, \bar{\mathbf{e}}_l) = \tanh\left(\mathbf{e}_h \cdot \mathbf{W}_{(\mathcal{L},\mathcal{T}_h)} \cdot \bar{\mathbf{e}}_l^T + \mathbf{b}_{(\mathcal{L},\mathcal{T}_h)}\right), \tag{15}$$

Table 2: Summary statistics on two real-world spatiotemporal datasets $\mathcal{B}^{(w1)}$ and $\mathcal{B}^{(w2)}$.

Dataset	$\lvert\mathcal{U}\rvert$	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{L}\rvert$	$\lvert\mathcal{S}\rvert$	Avg. $\lvert G\rvert$
$\mathcal{B}^{(w1)}$	10,545	7,984	7,393	651,356	242.8
$\mathcal{B}^{(w2)}$	8,017	6,389	4,445	135,805	61.3

where $\mathbf{W}_{(\mathcal{L},\mathcal{T}_h)}$ is the weight matrix of a bilinear transformation. Thus, we are able to generate the temporal pattern under impacts from the location units as:

$$\mathbf{p}_{\mathcal{T}_h}^{\mathcal{L}} = \sum_{h \in \mathcal{T}_h} \alpha_h^{(\mathcal{L},\mathcal{T}_h)} \mathbf{e}_h. \tag{16}$$

On the other hand, we also consider the impacts from temporal signals on locations signals. So the attention weight vector for location unit embeddings can be calculated as:

$$\alpha_l^{(\mathcal{T}_h,\mathcal{L})} = \frac{\exp(f(\mathbf{e}_l, \bar{\mathbf{e}}_h))}{\sum_{l \in L} \exp(f(\mathbf{e}_l, \bar{\mathbf{e}}_h))}, \tag{17}$$

and the spatial pattern under impacts from the time units is:

$$\mathbf{p}_{\mathcal{L}}^{\mathcal{T}_h} = \sum_{l \in L} \alpha_l^{(\mathcal{T}_h,\mathcal{L})} \mathbf{e}_l. \tag{18}$$

Then, these two one-way impacted spatiotemporal patterns are concatenated to get the interactive spatiotemporal pattern:

$$\mathbf{p}_{\mathcal{L},\mathcal{T}_h} = \mathbf{p}_{\mathcal{T}_h}^{\mathcal{L}} \oplus \mathbf{p}_{\mathcal{L}}^{\mathcal{T}_h}. \tag{19}$$

Similarly, we can generate the interactive spatiotemporal patterns for the other two type of time units of week $\mathbf{p}_{\mathcal{L},\mathcal{T}_w}$ and weekday $\mathbf{p}_{\mathcal{L},\mathcal{T}_y}$. Then, the final user representation is:

$$\mathbf{u} = \mathbf{p}_{\mathcal{L},\mathcal{T}_h} \oplus \mathbf{p}_{\mathcal{L},\mathcal{T}_w} \oplus \mathbf{p}_{\mathcal{L},\mathcal{T}_y}. \tag{20}$$

Thus, by substituting Eqn. (20) into Eqn. (10), CalendarGNN-Attn considers all interactions between the spatial pattern and temporal patterns when making predictions of user attributes.

5 EXPERIMENTS

In this section, we evaluate the proposed model on 2 real-world spatiotemporal behavior datasets. The empirical analysis covers: (1) effectiveness, (2) explainability, and (3) robustness and efficiency.

5.1 Datasets

We collected large-scale user behavior logs from 2 real portal websites providing news updates and articles on various topics, and created 2 spatiotemporal datasets $\mathcal{B}^{(w1)}$ and $\mathcal{B}^{(w2)}$. They contain users' spatiotemporal behavior log of browsing these 2 websites and both datasets range from Jan. 1 2018 to Jun. 30 2018. After all users have been anonymized, we filtered each dataset to keep around 10,000 users with most clicks. More statistics are provided in Table 2. The 3 user attributes used for prediction tasks are:

- $\mathcal{A}^{(gen)}$: the binary gender of user $\forall a^{(gen)} \in \{$"f", "m"$\}$ where "f" denotes female and "m" denotes male,
- $\mathcal{A}^{(inc)}$: the categorical income level of user such that $\forall a^{(inc)} \in \{0, 1, \ldots, 9\}$ where larger value indicate higher annual household income level and 0 indicates unknown,
- $\mathcal{A}^{(age)}$: the calculated age of user based on registered birthday. This label is treated as real value in all experiments.

Table 3: For dataset $\mathcal{B}^{(w1)}$, the performance of CalendarGNN, CalendarGNN-Attn (CalGNN-Attn), and baseline methods on predicting user attributes. For all metrics except error-based MAE and RMSE, higher values indicate better performance.

Method	Gender $\mathcal{A}^{(gen)}$				Income $\mathcal{A}^{(inc)}$				Age $\mathcal{A}^{(age)}$			
	Acc.	*AUC*	*F*1	*MCC*	*Acc.*	*F*1-macro	*F*1-micro	Cohen's kappa κ	R^2	*MAE*	*RMSE*	Pearson's r
LR	67.08%	.6469	.6628	.3319	19.54%	.0642	.1957	.0121	.0349	12.22	15.53	.2938
LearnSuc	67.41%	.6541	.6680	.3330	14.58%	.0531	.1523	.0078	.0523	12.18	15.49	.2989
SR-GNN	69.82%	.6733	.6854	.3510	20.21%	.0676	.1949	.0182	.0121	15.20	16.87	.2566
ECC	70.29%	.6886	.6832	.3825	23.54%	.0767	.2267	.0222	.2158	11.12	13.88	.4768
DiffPool	72.12%	.7189	.7089	**.4514**	25.87%	.0928	.2763	.0760	.2398	**10.55**	13.81	.4992
DGCNN	71.26%	.7129	.7068	.4189	24.55%	.0879	.2509	.0687	.2351	10.86	13.97	.4809
CapsGNN	70.85%	.6979	.6921	.4031	23.71%	.0750	.2189	.0378	.2270	10.90	13.86	.4645
SAGPool	71.95%	.7156	.7093	.4467	26.13%	.0942	.2554	.0797	.2350	10.77	13.91	.4887
CalendarGNN	**72.98%**	**.7250**	**.7119**	.4503	28.83%	.1059	.2981	.0887	**.2412**	10.57	13.60	.5033
CalGNN-Attn	72.70%	.7236	.7112	.4491	**29.67%**	**.1100**	**.3062**	**.0910**	.2401	10.65	**13.52**	**.5069**

5.2 Experimental Settings

5.2.1 Baseline methods. We compare CalendarGNN against state-of-the-art GNN-based methods:

- ECC [27]: This method performs edge-conditioned convolutions over local graph neighborhoods and generate graph embedding with a graph coarsening algorithm.
- DiffPool [38]: This method generates hierarchical representations of graph by learning a soft cluster assignment for nodes at each layer and iteratively merge nodes into clusters.
- DGCNN [42]: The core component SortPooling layer of this method takes unordered vertex features as input and outputs sorted graph representation vector of a fixed size.
- CapsGNN [36]: This method extracts both node and graph embeddings as capsules and uses routing mechanism to generate high-level graph or class capsules for prediction.
- SAGPool [17]: It uses self-attention mechanism on top of the graph convolution as a pooling layer and take the summation of outputs by each readout layer as embedding of the graph.

Besides above GNN-based approaches, we also consider the following methods for modeling user behaviors in session-based scenario:

- Logistic/Linear Regression (LR): The former one is applied for classification tasks and the later one is used for regression task. The input matrix is a row-wise concatenation of user's item frequency matrix and location frequency matrix.
- LearnSuc [30]: This method considers user's sessions as behaviors denoted by multi-type itemset structure [31]. The embeddings of users, items, and locations are jointly learned by optimizing the collective success rate or the user label.
- SR-GNN [34]: It uses graph structure to model user behavior of sessions and use GNN to generate node embeddings. The user session embedding is generated by concatenating the last item embedding and the aggregated items embedding.

We use open-source implementations provided by the original paper for all baseline methods and follow the recommended setup guidelines when possible. Our code package is available on Github: https://github.com/dmsquare/CalendarGNN.

5.2.2 Evaluation metrics. For classifying binary user label $\mathcal{A}^{(gen)}$, we use metrics of *mean accuracy* (Acc.), *Area Under the precision-recall Curve* (AUC), F1 score and *Matthews Correlation Coefficient* (MCC). For classifying multi-class user label $\mathcal{A}^{(inc)}$, metrics of *mean accuracy* (Acc.), F1 (macro, micro) averaged score and *Cohen's kappa* κ are reported. For numerical user label $\mathcal{A}^{(age)}$, metrics of *R-squared* (R^2), *Mean Absolute Error* (MAE), *Root-Mean-Square Error* (RMSE) and *Pearson correlation coefficient* (r) are reported.

5.3 Quantitative analysis

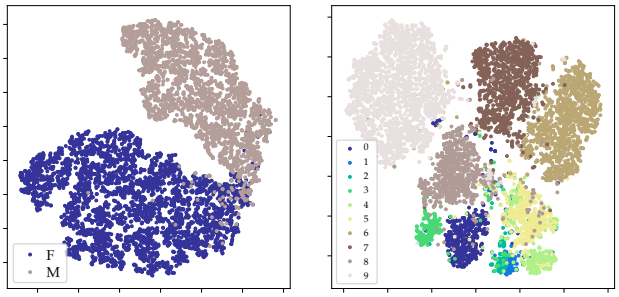
Table 3 and 4 present the experimental results of CalendarGNN and baseline methods on classifying/predicting user labels $\mathcal{A}^{(gen)}$, $\mathcal{A}^{(inc)}$, and $\mathcal{A}^{(age)}$ on datasets $\mathcal{B}^{(w1)}$ and $\mathcal{B}^{(w2)}$, respectively.

5.3.1 Overall performance. On dataset $\mathcal{B}^{(w1)}$, DiffPool achieves the best performance among all baseline methods. It scores an Acc. of 72.12% for predicting $\mathcal{A}^{(gen)}$, an Acc. of 25.87% for predicting $\mathcal{A}^{(inc)}$, and an RMSE of 13.81 for predicting $\mathcal{A}^{(age)}$. While on dataset $\mathcal{B}^{(w2)}$, SAGPool and DiffPool give comparable best performances. SAGPool slightly outperforms DiffPool that it scores a higher Acc. for predicting $\mathcal{A}^{(inc)}$, and a lower RMSE for predicting $\mathcal{A}^{(age)}$. Our proposed CalendarGNN outperforms all baseline methods across almost all metrics. On $\mathcal{B}^{(w1)}$, CalendarGNN scores an Acc. of 72.98% for predicting $\mathcal{A}^{(gen)}$ (+1.19% relatively over DiffPool), an Acc. of 28.83% for predicting $\mathcal{A}^{(inc)}$ (+11.44% relatively over DiffPool), and an RMSE of 13.60 for $\mathcal{A}^{(age)}$ (−1.52% relatively over DiffPool). On $\mathcal{B}^{(w2)}$, it scores an Acc. of 71.63%, an Acc. of 27.10%, and an RMSE of 13.88 for predicting $\mathcal{A}^{(gen)}$, $\mathcal{A}^{(inc)}$, and $\mathcal{A}^{(age)}$ respectively (+0.86%, +10.52%, and −2.32% over SAGPool). CalendarGNN-Attn further improves the Acc. for predicting $\mathcal{A}^{(inc)}$ to 29.67% and 28.17% on both datasets (+2.9% and +3.9% relatively over CalendarGNN); and, decreases RMSE for $\mathcal{A}^{(age)}$ to 13.52 and 13.67 (−0.6% and −1.5% relatively over CalendarGNN).

5.3.2 Compare against behavior modeling methods. SR-GNN gives the best performance of predicting user gender $\mathcal{A}^{(gen)}$ and user income $\mathcal{A}^{(inc)}$ among all behavior modeling methods. LearnSuc gives the best performance of predicting user age $\mathcal{A}^{(age)}$. This is probably because SR-GNN learns embedding for sessions instead of users and inferring user age of real values based on session embeddings are difficult than directly using user embedding. Beside, SR-GNN is designed to model session as a graph of items, but it ignores all spatial and temporal signals. On the contrary, our CalendarGNN models each user's behaviors as a single tripartite graph of sessions, locations, and items attributed by temporal signals.

Table 4: For dataset $\mathcal{B}^{(w2)}$, the performance of CalendarGNN, CalendarGNN-Attn (CalGNN-Attn), and baseline methods on predicting user attributes. For all metrics except error-based MAE and RMSE, higher values indicate better performance.

Method	Gender $\mathcal{A}^{(gen)}$				Income $\mathcal{A}^{(inc)}$				Age $\mathcal{A}^{(age)}$			
	Acc.	*AUC*	*F1*	*MCC*	*Acc.*	*F1*-macro	*F1*-micro	Cohen's kappa κ	R^2	*MAE*	*RMSE*	Pearson's r
LR	66.53%	.6410	.6523	.3100	18.21%	.0655	.1887	.0097	.0320	12.79	16.92	.2763
LearnSuc	67.01%	.6494	.6612	.3199	13.72%	.0522	.1587	.0060	.0489	12.72	16.93	.2789
SR-GNN	67.80%	.6562	.6660	.3289	19.79%	.0686	.1910	.0201	.0209	15.88	17.08	.2370
ECC	68.53%	.6802	.6792	.3580	21.08%	.0723	.2190	.0345	.2030	11.75	14.82	.4320
DiffPool	71.04%	.6998	.6967	.4269	24.09%	.0835	.2753	.0687	.2188	11.23	14.30	.4590
DGCNN	70.20%	.6972	.6855	.3892	22.70%	.0809	.2472	.0600	.2180	11.49	14.69	.4392
CapsGNN	68.29%	.6806	.6800	.3588	21.92%	.0789	.2196	.0438	.2059	11.82	14.69	.4389
SAGPool	71.02%	.7065	.6970	.4287	24.52%	.0856	.2802	.0701	.2223	10.97	14.21	.4652
CalendarGNN	**71.63%**	**.7104**	**.7038**	**.4389**	27.10%	.0909	.2798	.0742	.2223	**10.79**	13.88	.4872
CalGNN-Attn	71.47%	.7098	.7021	.4341	**28.17%**	**.1015**	**.2964**	**.0846**	**.2332**	10.87	**13.67**	**.4963**



(a) Clustering of user embeddings $\mathbf{u}$ is highly indicative about gender $\mathcal{A}^{(gen)}$

(b) Clustering of spatial patterns $\mathbf{p}_{\mathcal{L}}$ is highly indicative about income $\mathcal{A}^{(inc)}$

Figure 5: Clustering of user embeddings and patterns

And, this user spatiotemporal behavior graph is able to capture the complex behavioral spatial and temporal patterns. CalendarGNN outperforms SR-GNN by +4.53% and +42.65% relatively for Accs. of predicting $\mathcal{A}^{(gen)}$ and $\mathcal{A}^{(inc)}$ on dataset $\mathcal{B}^{(w1)}$, and by +5.65% and +36.9% on dataset $\mathcal{B}^{(w2)}$. CalendarGNN outperforms LearnSuc by −12.20% and −18.02% for the RMSEs of predicting $\mathcal{A}^{(age)}$.

5.3.3 Compare against GNN methods. DiffPool performs the best among all GNN-based baseline methods on dataset $\mathcal{B}^{(w1)}$. It scores an Acc. of 72.12% for predicting user gender $\mathcal{A}^{(gen)}$ (+3.29% relatively over SR-GNN), an Acc. of 25.87% for predicting user income $\mathcal{A}^{(inc)}$ (+28.01% relatively over SR-GNN), and an RMSE of 13.81 for predicting user age $\mathcal{A}^{(age)}$ (−10.85% relatively over LearnSuc). SAGPool shows competitive good performance on dataset $\mathcal{B}^{(w2)}$. Both of these two methods learn hierarchical representation of general graphs. They are not designed to capture the specific tripartite graph structure of sessions, items, and locations. And, these methods are not capable of modeling the explicit time structures in user's spatiotemporal behaviors.

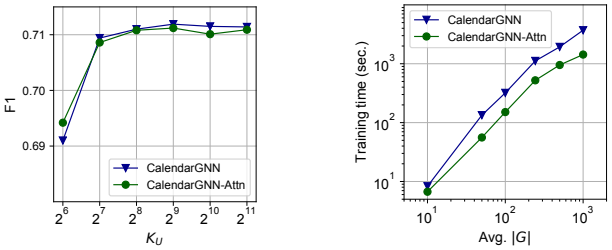
DGCNN underperforms DiffPool and SAGPool across all metrics on both datasets. One reason is that DGCNN's core component SortPooling layer relies on a node sorting algorithm (in analogous to sort continuous WL colors [32]). This strategy produces lower performance for predicting user demographic labels compared with the learned hierarchical representations adopted by DiffPool and SAGPool. ECC and CapsGNN yield slightly better performance than behavior modeling method SR-GNN for predicting user gender $\mathcal{A}^{(gen)}$. But, they can quite outperform SR-GNN for predicting $\mathcal{A}^{(inc)}$, and outperform LearnSuc by a large margin for predicting $\mathcal{A}^{(age)}$. This validates the spatiotemporal behavior graph of sessions, items, and locations (instead of itemset or simple item-session graph) provides more information for the GNN model.

Our CalendarGNN performs the best among all GNN-based methods across almost all metrics. On dataset $\mathcal{B}^{(w1)}$, CalendarGNN scores an Acc. of 72.98% for $\mathcal{A}^{(gen)}$ (+1.19% relatively over DiffPool), an Acc. of 28.83% for $\mathcal{A}^{(inc)}$ (+11.44% relatively over DiffPool), and an RMSE of 13.60 for $\mathcal{A}^{(age)}$ (−1.52% relatively over DiffPool). On dataset $\mathcal{B}^{(w2)}$, it scores an Acc. of 71.63%, an Acc. of 27.10%, and an RMSE of 13.88 for predicting $\mathcal{A}^{(gen)}$, $\mathcal{A}^{(inc)}$, and $\mathcal{A}^{(age)}$ respectively (+0.86%, +10.52%, and −2.32% over SAGPool). This confirms that the proposed calendar-like neural architecture of CalendarGNN is able to distill user embeddings of greater predictive power on demographic labels.

By considering the interactions between spatial and temporal pattern, CalendarGNN-Attn further improves the Acc. for predicting $\mathcal{A}^{(inc)}$ to 29.67% and 28.17% on both datasets (+2.9% and +3.9% relatively over CalendarGNN); and, decreases RMSE for $\mathcal{A}^{(age)}$ to 13.52 and 13.67 (−0.6% and −1.5% relatively over CalendarGNN). We also note that CalendarGNN-Attn underperforms CalendarGNN on both datasets for predicting $\mathcal{A}^{(age)}$. This indicates the interactions between spatial and temporal patterns provide no extra information for predicting user genders. More results for examining the importance of each spatial or temporal pattern in different predictive tasks can be found in the supplemental materials

5.4 Qualitative analysis

In Figure 5, we provide visualizations of user embeddings and patterns learned by CalendarGNN using t-SNE [21]. The clustering results presented in Figure 5(a) clearly demonstrate that the learned user embeddings are highly indicative about the target user attribute $\mathcal{A}^{(gen)}$. Furthermore, we plot the learned spatial patterns $\mathbf{p}_{\mathcal{L}}$ in Figure 5(b) and it can be seen that they are especially useful for determining user's income level $\mathcal{A}^{(gen)}$: users of high income levels (e.g., "7", "8" and "9") forms distinct non-overlapping clusters despite some users of lower income level (e.g., "1") and unknown ("0") scatters at the bottom part.



(a) Change of $F1$ scores for predicting user gender $\mathcal{A}^{(gen)}$ on $\mathcal{B}^{(w1)}$

(b) Per epoch training time w.r.t. average input graph size $|G|$

Figure 6: Sensitivity and efficiency of CalendarGNN.

5.5 Sensitivity and Efficiency

We test through CalendarGNN's hyper-parameters. Figure 6(a) shows the prediction performance is stable for a range of user embedding dimensions $K_{\mathcal{U}}$ from 2^7 to 2^{11}. We also test the model's efficiency. All experiments are conducted on single server with dual 12-core Intel Xeon 2.10GHz CPUs with single NVIDIA GeForce GTX 2080 Ti GPU. Figure 6(b) shows the per epoch training time is linear to the average size of input user spatiotemporal graphs.

6 CONCLUSIONS

In this work, we proposed a novel Graph Neural Network (GNN) model for learning user representations from spatiotemporal behavior data. It aggregates embeddings of items and locations into session embeddings, and generates user embedding on the calendar neural architecture. Experiments on two real datasets demonstrate the effectiveness of our method.

ACKNOWLEDGMENTS

This research was supported in part by Condé Nast, and by NSF Grants IIS-1849816 and IIS-1447795. This research was also supported in part by the National Science Centre, Poland research project no.2016/23/B/ST6/01735.

REFERENCES

[1] Mohamed Aly, Andrew Hatch, Vanja Josifovski, and Vijay K Narayanan. 2012. Web-scale user modeling for targeting. In *WWW*. 3–12.
[2] Ludovico Boratto, Salvatore Carta, Gianni Fenu, and Roberto Saia. 2016. Using neural word embeddings to model user behavior and detect user segments. *Knowledge-based systems* 108 (2016), 5–14.
[3] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv:1312.6203* (2013).
[4] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).
[5] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*. 3844–3852.
[6] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *NeurIPS*. 2224–2232.
[7] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *ICML*. 1263–1272.
[8] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS*. 1024–1034.
[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. 173–182.
[10] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv:1511.06939* (2015).
[11] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *RecSys*. 241–248.
[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[13] Dietmar Jannach and Malte Ludewig. 2017. When recurrent neural networks meet the neighborhood for session-based recommendation. In *RecSys*. 306–310.
[14] Meng Jiang, Peng Cui, Fei Wang, Xinran Xu, Wenwu Zhu, and Shiqiang Yang. 2014. Fema: flexible evolutionary multi-faceted analysis for dynamic behavioral pattern discovery. In *KDD*. 1186–1195.
[15] Meng Jiang, Christos Faloutsos, and Jiawei Han. 2016. Catchtartan: Representing and summarizing dynamic multicontextual behaviors. In *Proceedings of the 22nd ACM SIGKDD*. 945–954.
[16] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv:1609.02907* (2016).
[17] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. 2019. Self-Attention Graph Pooling. *arXiv:1904.08082* (2019).
[18] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *CIKM*. 1419–1428.
[19] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv:1707.01926* (2017).
[20] Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *IJCAI*. 4068–4074.
[21] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.
[22] Franco Manessi, Alessandro Rozza, and Mario Manzo. 2017. Dynamic graph convolutional networks. *arXiv:1704.06199* (2017).
[23] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *ICML*. 807–814.
[24] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. 2016. Learning convolutional neural networks for graphs. In *ICML*. 2014–2023.
[25] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681.
[26] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. 2018. Structured sequence modeling with graph convolutional recurrent networks. In *ICNIP*. 362–373.
[27] Martin Simonovsky and Nikos Komodakis. 2017. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *CVPR*. 3693–3702.
[28] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *Workshop on DLRS*. 17–22.
[29] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv:1710.10903* (2017).
[30] Daheng Wang, Meng Jiang, Qingkai Zeng, Zachary Eberhart, and Nitesh V Chawla. 2018. Multi-type itemset embedding for learning behavior success. In *KDD*. ACM, 2397–2406.
[31] Daheng Wang, Tianwen Jiang, Nitesh V Chawla, and Meng Jiang. 2019. TUBE: Embedding Behavior Outcomes for Predicting Success. In *Proceedings of the 25th ACM SIGKDD*. 1682–1690.
[32] Boris Weisfeiler and Andrei A Lehman. 1968. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia* 2, 9 (1968), 12–16.
[33] Chen Wu and Ming Yan. 2017. Session-aware information embedding for e-commerce product recommendation. In *CIKM*. 2379–2382.
[34] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *AAAI*, Vol. 33. 346–353.
[35] Xian Wu, Baoxu Shi, Yuxiao Dong, Chao Huang, Louis Faust, and Nitesh V Chawla. 2018. Restful: Resolution-aware forecasting of behavioral time series data. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1073–1082.
[36] Zhang Xinyi and Lihui Chen. 2019. Capsule graph neural network. In *ICLR*.
[37] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *KDD*. 974–983.
[38] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. 2018. Hierarchical graph representation learning with differentiable pooling. In *NeurIPS*. 4800–4810.
[39] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv:1709.04875* (2017).
[40] Wenhao Yu, Mengxia Yu, Tong Zhao, and Meng Jiang. 2020. Identifying referential intention with heterogeneous contexts. In *Proceedings of The Web Conference 2020*. 962–972.
[41] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. In *NeurIPS*. 5165–5175.
[42] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. 2018. An end-to-end deep learning architecture for graph classification. In *AAAI*.