IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 18 January 2024, accepted 22 February 2024, date of publication 4 March 2024, date of current version 18 April 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3372397

RESEARCH ARTICLE

Enhancing App Usage Prediction Accuracy With GCN-Transformer Model and Meta-Path Context

XI FANG, HUI YANG, DING DING, WENBIN GAO, LEI ZHANG, YILONG WANG, AND LIU SHI
Beijing Electronic Science and Technology Institute, Beijing 100070, China

Corresponding author: Xi Fang (xfang@besti.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61701008, in part by the Fundamental Research Funds for the Central Universities under Grant 3282023054, in part by the Sophisticated Academic Discipline Construction Project for Capital University under Grant 20210093Z0401 and Grant 20210036Z0401, and in part by the Funding Project for Excellent Master of Beijing Electronics Science and Technology Institute under Grant 328202255 and Grant 328202266.

ABSTRACT In this paper, we introduce MP-GT, a novel Graph Neural Network model that leverages meta-path-guided optimization within the GCN-Transformer framework to enhance application (App) usage prediction accuracy. Our approach addresses issues such as suspended animation and over-smoothing by extracting both local subgraph structures and global graph structures using a combination of GCN and Transformer method. Furthermore, we enhance the capture of semantic information and App usage patterns by incorporating a meta path-guided objective function. Extensive experiments demonstrate that MP-GT outperforms the widely adopted baseline of semantic-aware representation learning via Graph Convolutional Network (SA-GCN) by 13.33% in terms of Accuracy@1. Additionally, MP-GT surpasses the popular baseline of context-aware App usage prediction with heterogeneous graph embedding (CAP) by 74.02% in the same metric. Moreover, MP-GT reduces training time by 79.47% compared to SA-GCN. These findings validate that our approach not only achieves higher prediction accuracy but also converges faster than the baseline models. Therefore, MP-GT proves to be an effective and superior solution for the App usage prediction task.

INDEX TERMS Heterogeneous graph embedding, GCN-transformer, meta-path optimization, application usage modeling.

I. INTRODUCTION

Smartphones have become an indispensable part of people's lives, and understanding App usage behaviors has significant implications for both users and App developers [1]. Predicting the next App that a user is likely to use at a given time is a crucial task in comprehending these behaviors. There exist three types of classic prediction methods.

Firstly, similarity-based methods, such as collaborative filtering [2], rely on user or item similarities to make predictions and recommendations. Secondly, probabilistic graphical models, including Markov models [3], [4], [5] and Bayesian models [6], [7], [8], [9], provide effective tools for modeling relationships between users and items. Thirdly, deep learning-based approaches, such as multi-task learning [10], multimodal embedding methods [11], and transfer learning methods [2], utilize deep neural networks to enhance prediction accuracy.

A notable contribution in this field is the utilization of meta-paths to construct a likelihood function, as demonstrated in [12]. This approach effectively captures contextual information within spatiotemporal App usage patterns. Currently, the model proposed in this paper, SA-GCN, stands out as an algorithm with superior performance. It is a graph representation model that employs Graph Convolutional Networks (GCN) to learn an objective function based on

The associate editor coordinating the review of this manuscript and approving it for publication was Giuseppe Desolda.

meta-paths. SA-GCN, by introducing a meta-path-guided learning strategy, showcases its potential in preserving the similarity among multi-modal units.

However, effectively leveraging context information in App usage prediction remains challenging due to the heterogeneity and complexity of App usage data. Furthermore, challenges like the suspended animation problem and over-smoothing issues within GCN result in extended training times and, particularly when dealing with large datasets, suboptimal convergence effects.

In this study, we propose a novel approach to address these challenges, aiming to enhance convergence speed and capture long-range dependencies within GCN. Our method, named MP-GT, introduces a Transformer submodule behind the GCN layer while retaining the meta-path-based objective function from SA-GCN. The Transformer subnetwork utilizes the attention mechanism to compute interactions among all nodes, enabling the capture of long-distance dependencies and mitigating over-smoothing and over-squeezing issues. This combination of GCN and Transformer makes our model more expressive and comprehensive in understanding the App usage graph. Importantly, our approach seamlessly integrates into existing GCN frameworks.

We evaluate MP-GT on a large real-world datasets of App usage. Experiments show that MP-GT converges faster than SA-GCN. Furthermore, we observe significant improvements in accuracy metrics, namely Mean Reciprocal Rank (MRR) and Accuracy@k. The contributions of this paper can be summarized as follows:

- Proposal of MP-GT: We develop MP-GT, a novel model that combines meta-path-based optimization with GCN-Transformer learning, which captures both short-range structural information and long-range reasoning abilities, and can facilitate a wide range of downstream applications.
- Enhanced convergence speed: Our results show that MP-GT exhibits faster convergence compared to SA-GCN. Notably, we accomplished a remarkable 79.47% reduction in training time in comparison to SA-GCN, showcasing the efficiency and effectiveness of our proposed MP-GT method.
- Improved accuracy: We demonstrate notable advancements in accuracy measures such as MRR and Accuracy@k. More specifically, we achieve the performance gain of 13.33%, 8.15% and 13.57% compared with widely used model SA-GCN and 74.02%, 79.06% and 97.02% compared with the model CAP in terms of the Accuracy@1, Accuracy@10 and MRR.

The structure of this paper is organized as follows: Section II introduces the background, problem description and dataset. Section III delves into our proposed methodology, detailing the architecture and design of MP-GT. Section IV focuses on the evaluation of our approach using a substantial real-world dataset, highlighting the achieved performance improvements. Section V is dedicated to the discussion of related work in the field. Finally, we conclude this work in Section VI.

II. PRELIMINARIES

A. BACKGROUND

It is worth noting that by combining multiple methods and utilizing advanced techniques, such as context modeling based on meta paths or graph neural networks, the prediction performance of App usage prediction can be enhanced. Our work draws inspiration from advanced techniques in App usage prediction, particularly the SA-GCN model proposed by Yu et al. in [12].

SA-GCN is a graph neural network model for predicting App usage, but we have identified that it lacks sufficient information fusion and struggles to effectively capture the contextual information of nodes. This is primarily due to the fact that GCN only considers local neighbor information and overlooks the global perspective of nodes. As a result, it suffers from slow convergence, extended training times, and diminished performance in downstream tasks. Drawing from the popular Transformer model [13], our goal is to harness its capabilities in capturing the global information of nodes, resulting in improved representations of App usage. To take advantage of both GCN and Transformer, we integrate these two architectures to learn node representations. However, as this approach only considers binary co-occurrence relationships, we continue to employ a meta-path guided optimization function to capture observed co-occurrence relationships.

This enables the model to acquire richer and more meaningful contextual relationships, thereby improving prediction performance. Additionally, the model's rapid convergence speed reduces the number of iterations required, effectively addressing the issue of extended training times due to slow convergence when using GCN on our large datasets, resulting from an excessive number of iterations.

B. PROBLEM DEFINITION

We aim to complete heterogeneous graph embedding and predict which App will be used by the user in the next time slot given his historical records. To better describe this problem, we formally give the relevant definitions as follows.

We denote $\mathcal{A}$, $\mathcal{U}$ as the set of all Apps and users, respectively. Let $r_{u_i} = (t_i, l_i, a_i)_u$ be a tuple data belonging to user u, which is the i-th App usage record of user u, where i is the index of the record, a_i denotes the App used in time t_i and location l_i. For each user u, let $\mathcal{R}_u = \{r_{u_1}, r_{u_2}, \cdots, r_{u_N}\}$. Here, N is the number of all his/her App usage records.

1) PROBLEM 1

User Embedding Generation Problem:

Given $\mathcal{R}_u$, where $u \in \mathcal{U}$. We intend to generate user embedding vectors in a low-dimensional latent space considering their App usage history trajectories containing App, time and location, respectively. Here, $E_u = map(\mathcal{R}_u)$.

E_u, *map* are the embedding of user u and the mapping function, respectively. In this paper, we utilize the MP-GT model as the *map*.

2) PROBLEM 2

App Usage Prediction Problem:

predicting the App a_t to be used in the time t, Given a target user u, next time t, and the user's historical App usage records $\mathcal{R}_u$, respectively. To predict the App a_t to be used in the time t. In other words, estimating the probability of using each App $a \in \mathcal{A}$ based on the App usage sequence $\mathcal{R}_u$, i.e., $P(a|\mathcal{R}_u)$. Thus, $a_t = \arg\max_{a \in \mathcal{A}} P(a|\mathcal{R}_u)$. The prediction is based on the history mobile App usage of all users. The algorithm outputs most possible mobile Apps user u will use.

C. DATASET

1) DATASET COLLECTION

We make use of a substantial real-world mobility dataset that has been made available through a recent study [14]. This dataset was collected by one of the largest Internet service providers (ISPs) in Shanghai, China, during the period spanning from April 20th to 26th, 2016. It encompasses the access logs of more than ten thousand mobile users to the cellular network. Through an in-depth examination of network data, each access record is characterized by an anonymized user ID, timestamp, the cellular base station with GPS coordinates, and metadata related to network communication. For App information, we have identified over 2000 unique Apps from the networking metadata using the SAMPLES method [15]. For location information, we utilize cellular base stations as the fundamental units of location and employ widely recognized Voronoi diagrams [16] to partition the city based on the coordinates of these stations. As a result, each location can be linked to a distinct cellular base station, and the corresponding IDs are used to represent the location of App usage. For time-related information, we discretize each day into 24 units and employ these units to record the timing of App usage. This approach helps maintain a low similarity score during plagiarism checks.

2) DATA PREPROCESSING AND CLEANING

Due to the ease with which the most frequently used applications can occur hundreds of millions of times, they offer less meaningful information compared to rare applications. To address the imbalance between rare and common applications, we employed a straightforward subsampling method [17]: for each record with App $a \in \mathcal{A}$ in the dataset, it is discarded with probability of$P(a) = max(1 - \sqrt{f_{th}/f_a}, 0)$, where f_a is the frequency of App a, $f_{th}$is the chosen threshold, which is set as $2|\mathcal{A}|$ by default.This Approach sub-samples Apps whose frequency is greater than f_{th} while preserving the ranking of the frequencies. After subsampling, we filtered out users with fewer than 10 records, applications with fewer than 5 records, and locations with fewer than 5 records, as they lacked a sufficient number of records to reveal meaningful temporal or spatial patterns. Following preprocessing, the dataset used for subsequent experiments comprises 11,170 users, 1,792 applications, and 9,330 locations.

3) ETHICS

Our research on the ISP dataset prioritizes user privacy by excluding personal details in App records and anonymizing user IDs as bit strings. All data is securely stored on provider servers, and analysis is confined to aggregated results. The study has received approval from both the ISP and the local review board, ensuring compliance with China's Data Protection Regulation. Researchers strictly adhere to confidentiality agreements, and access to the dataset is restricted to authorized personnel. These measures underscore our commitment to privacy and ethical standards throughout the research process.

III. METHOD

Figure 1 illustrates the overall framework of our method. We construct an App usage heterogeneous graph based on

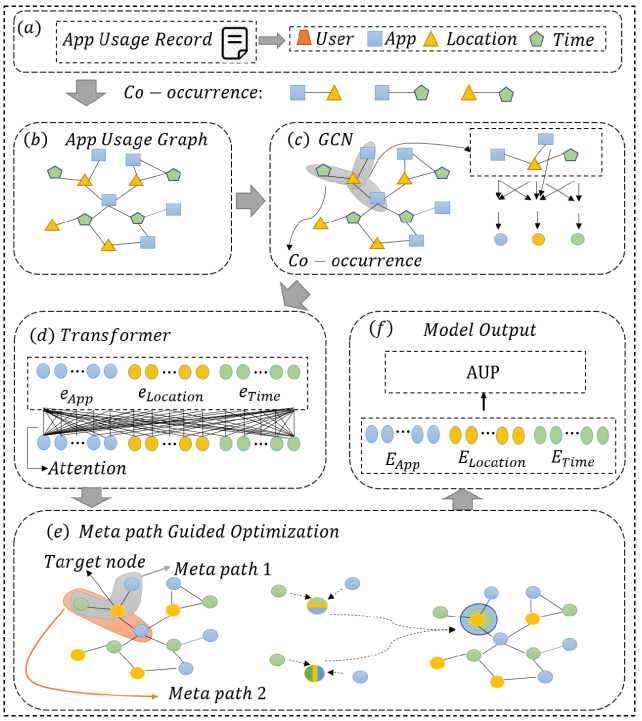

FIGURE 1. The overall architecture of MP-GT. The feature and adjacency matrix of graph are input into the representation learning model MP-GT. Then we utilize meta path-based context to learn representations and generate embeddings for different units through GT. (a) is the App usage recoed composed by an user, App, location and time units; (b) is the App usage graph constructed by co-occurrence relationship and the input to our model; (c)-(e) is the MP-GT learning process, each node's embedding in (c) is updated by the weighted sum of itself and its neighbors, each node's embedding in (d) is updated by the weighted sum of global nodes via attention mechanism; (e) is the meta path guided optimization, each node's loss function is updated by the nodes existed in meta path App-Location-Time; (f) is the output embeddings of App, location and time, and those have been used for App usage prediction.

App usage records. By doing so, we capture the features of units and edges with the different types. Then the semantic-aware App usage embedding obtained by MP-GT guided meta-path objective function. This can significantly facilitate the downstream task such as App usage prediction.

A. APP USAGE GRAPH CONSTRUCTION

SA-GCN [12] and GinApp [18] reveal the importance of effectively representing App usage graph. In order to construct a heterogeneous graph for App usage records, we first introduce the problem setting. Formally, we define the App usage graph as $\mathcal{G} = (V, E, \mathcal{W})$, where V, E stand for the set of nodes and edges, respectively. W represents the set of edges' weight, in other words, the adjacency matrix.

1) NODE CONSTRUCTION

For $u \in \mathcal{U}$, we denote a_i, t_i, l_i with different value as nodes, respectively. Here a_i, t_i, $l_i \in r_{u_i} \in \mathcal{R}_u$. $\mathcal{A}$, $\mathcal{T}$, $\mathcal{L}$ are defined as sets of App, time, and location, respectively. In that way, $V = \{v_1, v_2, \cdots\} = \{a_1, a_2, \cdots, a_{|\mathcal{A}|}, t_1, t_2, \cdots, t_{|\mathcal{T}|}, l_1, l_2, \cdots, l_{|\mathcal{L}|}\}$. The total number of V is $|V| = |\mathcal{A}| + |\mathcal{T}| + |\mathcal{L}|$. In the graph $\mathcal{G}$, there are only three types of nodes, namely App, time, and location nodes with type of 'App', 'time', 'location'.

2) EDGE CONSTRUCTION

For each $u \in \mathcal{U}$, let (a_i, t_i), (t_i, l_i), and (a_i, l_i) be edges, respectively. That means (a_i, t_i), (t_i, l_i), and (a_i, l_i) are regarded as edges when they appear in the same record (a_i, t_i, l_i). Therefore, $E = \{(v_i, v_j) \mid (v_i, v_j) \in (a_i, t_i), (t_i, l_i), (a_i, l_i)\}$. Here node v_i, $v_j \in \mathcal{V}$, and a_i, t_i, $l_i \in r_{u_i} \in \mathcal{R}_u$. There are three types of edges in the graph, namely App-time, time-location and App-location, written as (a, t), (t, l) and (a, l), respectively. These edges preserve the co-occurrence relationship among App, time, and location units in the same record.

3) EDGE WEIGHT CONSTRUCTION

For node v_i and v_j, let w_{ij} be the frequency of edge (v_i, v_j) appearing in the App usage records. Thus, we can define w_{ij} as:

$$w_{ij} = \begin{cases} w_{ij} & v_i,\ v_j \textit{ are the both ends of edge } (v_i, v_j), \\ 0 & \textit{otherwise}. \end{cases} \tag{1}$$

$\mathcal{W} = (w_{ij})_{|\mathcal{V}|\times|\mathcal{V}|}$ is the unnormalized adjacency matrix, while w_{ij} is used to indicate the strength of the interrelationship between node v_i and v_j at both ends of the edge, effectively modeling the strength of the co-occurrence between units.

In the MP-GT Learning module, we adopts the alias sampling method to sample the App-time-location meta-path by using probability function. Hence, It is necessary to normalize $\mathcal{W}$. Here, we employ the min-max normalization method. Let w^m_{at}, w^m_{tl}, w^m_{al} be the maximum weight for (a, t), (t, l), (a, l) edge, respectively.

The normalized edge weight $\widehat{w_{ij}}$ is given as:

$$\widehat{w_{ij}} = \begin{cases} \dfrac{w_{ij}}{w^m_{at}} & (v_i, v_j) \in (a, t), \\ \dfrac{w_{ij}}{w^m_{tl}} & (v_i, v_j) \in (t, l), \\ \dfrac{w_{ij}}{w^m_{al}} & (v_i, v_j) \in (a, l), \\ 1 & v_i = v_j, \\ 0 & \textit{otherwise}. \end{cases} \tag{2}$$

Based on 2, the normalized adjacency matrix is given as: $\widehat{\mathcal{W}} = (\widehat{w_{ij}})_{|\mathcal{V}|\times|\mathcal{V}|}$.

4) FEATURE EXTRACTION

To capture the semantic-rich attribute information, we assign a feature vector to each node, following the design in reference [12]. The feature vector consists of three types of vectors: App feature vector, location feature vector, and time feature vector. For Apps, feature information is determined by the type of the App, as Apps of the same type typically exhibit similar usage patterns. For locations, each location encompasses the base station and its surroundings. The distribution of Points of Interest within this area reflects its socioeconomic function, thus encapsulating the attributes of the region. For time, we categorize time slots into working days and non-working days due to significant variations in people's lifestyles between these two types of days.

B. MP-GT LEARNING

1) PROPAGATION VIA GT

Our model GT is composed of two main modules: a GCN subnetwork and a Transformer subnetwork. In the following sections, we will provide a more detailed explanation of each module.

a: GCN MODULE

Graph Convolutional Network (GCN) [19] is a kind of convolutional neural network that can directly act on graph and utilize its structural information. In our MP-GT model, we adopt a similar configuration as described in [12], where the number of convolutional layers is determined. Specifically, we set k = 2, indicating that our GCN module consists of two convolutional layers. Let $e = (e_1, e_2, \ldots, e_{|V|})$ be the matrix of all node embedding vectors propagated by the GCN module. The propagation function used to calculate these node embeddings is given by the equation:

$$e = \hat{A}\sigma\left(\hat{A}XW^{(1)}\right)W^{(2)}. \tag{3}$$

In this equation, σ represents the non-linear transformation, and X is the feature matrix. $\hat{A}$ is the normalized adjacency matrix defined as $\hat{A} = \tilde{D}^{-(1/2)}\tilde{A}\tilde{D}^{-(1/2)}$, where $\tilde{A} = A + I$. Here A is the original adjacency matrix, and I denotes the identity matrix. $\tilde{D}$ is a diagonal matrix with $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. The parameters $W^{(1)}$ and $W^{(2)}$ are trainable matrices. $W^{(1)}$

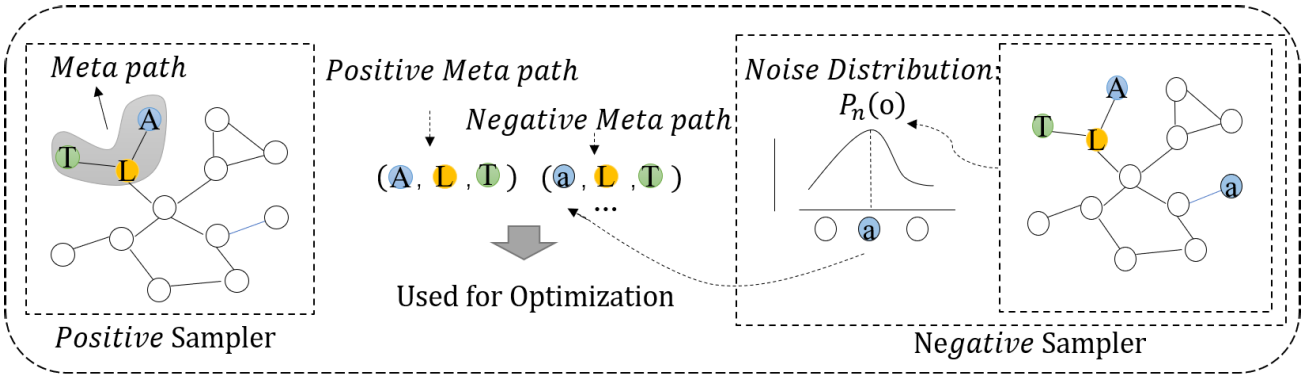

FIGURE 2. Negative sampling for enhanced computational efficiency in meta-path guided optimization.

has dimensions $R^{D_i \times D_g}$, where D_i is the dimension of input feature vector, and D_g is the dimension of GCN's output embeddings. $W^{(2)}$ has dimensions $R^{D_g \times D_g}$. In line with the design in reference [12], we adopt the same adjacency matrix A and the feature matrix X as used in that work.

b: TRANSFORMER MODULE

Because the structural information of the graph is already encoded by the GCN module, the Transformer subnetwork in our model does not include positional embeddings. The encoder in the Transformer operates as follows:

$$\begin{aligned} H^l &= Transformer(H^{(l-1)}) \\ &= softmax\left(\frac{QK^T}{\sqrt{D_o}}\right)V, \end{aligned}$$

where

$$\begin{cases} Q = H^{(l-1)}W_Q^l, \\ K = H^{(l-1)}W_K^l, \\ V = H^{(l-1)}W_V^l. \end{cases} \tag{4}$$

In the above equations, W_Q^l, W_K^l, $W_V^l \in \mathbb{R}^{D_g \times D_o}$ are parameter matrices that can be learned. Q, K and V represent the query, key, and value vector sequences, respectively, which are obtained by multiplying the previous layer's embeddings H^{l-1} with their corresponding parameter matrices. D_g is the dimension of the output embeddings from the GCN module, and D_o is the dimension of the output embeddings from the Transformer.

The GCN embeddings e are input into Transformer module. Each encoder layer in the Transformer extracts features from the input embeddings e. To capture a larger global view and reduce parameters, we design a two-layer encoder. The propagation process of the Transformer can be represented as follows:

$$\begin{cases} H^1 = Transformer(e), \\ H^2 = Transformer(H^1). \end{cases} \tag{5}$$

Therefore, the final embeddings of the nodes in the graph $\mathcal{G}$ are given by: $E = (E_1, E_2, \ldots, E_{|V|}) = (H_1^2, H_2^2, \ldots, H_{|V|}^2)$.

2) OBJECTIVE FUNCTION

For our problem, we are not dealing with a typical supervised task. Instead, we adopt the learning method described in [12], which focuses on capturing the co-occurrence relationship for units using a "App-time-location" meta path-guided learning strategy.

In this approach, for each meta-path (e.g., App a, time t, location l), if it occurs in the record $r = (a, t, l)$, we aim to maximize its likelihood. Let's denote a node $e \in r$ and its context in the record: $r_{-e} = \{o | o \in r, o \neq e\}$. The occurrence probability of each node e given its context is defined as:

$$p(e|r_{-e}) = \frac{exp(s(e, r_{-e}))}{\sum_{o \in \mathcal{X}_e} exp(s(o, r_{-e}))}. \tag{6}$$

where $\mathcal{X}_e$ is the set of nodes with the same type of e. Specifically,

$$\mathcal{X}_e = \begin{cases} \mathcal{A}, & e \in \mathcal{A}(App\ set), \\ \mathcal{T}, & e \in \mathcal{T}(Time\ set), \\ \mathcal{L}, & e \in \mathcal{L}(Location\ set), \end{cases}$$

The similarity score $s(\cdot)$ between the unit e and its context is defined as:

$$s(e, r_{-e}) = \mathbf{E_e} \cdot \left(\frac{1}{|r_{-e}|} \sum_{o \in r_{-e}} \mathbf{E_o}\right). \tag{7}$$

where $\mathbf{E_e}$ is the D_o-dimensional embedding of node e.

The objective function aims to maximize the log-likelihood of Equation 6 with respect to all units and all records. It can be written as:

$$O = -\sum_{u \in \mathcal{U}} \sum_{r \in \mathcal{R}_u} \sum_{e \in r} logP(e|r_{-e}). \tag{8}$$

Here u is a user, $\mathcal{U}$ and $\mathcal{R}_u$ are the user set and the record set of user u, respectively.

However the summation between r_e and $\mathcal{X}_o$ results in the high computational complexity. To enhance computational efficiency, we employ a technique called negative sampling [17] to increase the positive sample probability and decrease the negative sample probability. Specifically, for a given node e as positive sample, we randomly select K negative samples

from $\mathcal{X}_e$, that are not present in the record r (see Fig.2). Consequently, the loss function can be formulated as follows:

$$L = -log\sigma(s(e, r_{-e})) - \sum_{i=1}^{K} \mathbb{E}_{o_i \sim P_n(o)} \left[log\sigma(-s(o_i, r_{-e})) \right]. \tag{9}$$

Here, $\sigma(\cdot)$ is the sigmoid function, and o_i is the i-th negative sample from $\mathcal{X}_e$, $P_n(o)$ refers to the noise distribution. To optimize equation 9, we employ stochastic gradient descent (SGD) as the optimization algorithm.

C. COMPLEXITY ANALYSIS

In the MP-GT model, the MP-GT learning is the main operation. In this algorithm, three dimensions are essential: D_i, D_g and D_o are the dimension of the input feature, output embedding of GCN and output embedding of Transformer, respectively.

First, the propagation of GCN adopted two layers. So the computational complexity of the sparse matrix multiplication in (3) is $O(|V|D_iD_g)$ and $O(|V|D_gD_g)$ for two layers respectively. Second, the propagation of Transformer in (4) has computational complexity as $O(|V|^2D_o)$. Third, the computational complexity of meta path-guided optimization in (9) is $O((K+1)D_o)$. Where K represents the number of negative samples. In summary, the overall computational complexity of the MP-GT algorithm can be expressed as $O(|V|D_iD_g + |V|D_gD_g + |V|^2D_o + (K+1)D_o)$.

D. APPLICATION

To complete App usage prediction task, we follow a two-step process. Firstly, we generate a user embedding to capture their dynamic preferences. Secondly, we compare this user embedding with the App embedding to make predictions.

1) USER EMBEDDING GENERATION

To capture the spatio-temporal properties of App usage for each user, we adopt the method proposed in [12] to generate a new embedding, referred to as a user profile. For a given user $u \in \mathcal{U}$ and a specific time point τ, we begin by extracting the user's App usage records from $\mathcal{R}_u$ that occurred before τ. These records can be defined as: $\mathcal{R}_u^{t<\tau}$. Here $\mathcal{R}_u^{t<\tau} \subset \mathcal{R}_u$, $(a_i, t_i, l_i) \in R_u^{t<\tau}$ represents the i-th record of user u. Subsequently, we define the user profile for user u at time τ as follow:

$$\mathbf{u}_\tau = \sum_{(a_i,t_i,l_i)\in R_u^{t<\tau}} \left[\beta e^{-(\tau-t_i)/T} \mathbf{E}(l_i) + (1-\beta) e^{-(\tau-t_i)/T} \mathbf{E}(a_i) \right]. \tag{10}$$

Here $E(l_i)$ and $E(a_i)$ are the GT embedding of location and App. The term $e^{-(\tau-t_i)}$ is a time decay factor, which reduces the influence of older records on the user profile. $T = 24$ indicates the total hours of a day. The parameter $\beta \in [0, 1]$ can be treated as a balance coefficient between App usage history and trajectory history.

2) APP USAGE PREDICTION

After generating the user embedding, we obtain the user profile $\mathbf{u}_\tau$, where $u \in \mathcal{U}$. Subsequently, we compute the score of different Apps a_j by calculating the similarity between user profile $\mathbf{u}_\tau$ and the App embedding $\mathbf{E}(\mathbf{a_j})$ using the following approach:

$$s_u(a_j) = \frac{\mathbf{u}_\tau \cdot \mathbf{E}(\mathbf{a_j})}{\|\mathbf{u}_\tau\| \cdot \|\mathbf{E}(\mathbf{a_j})\|}. \tag{11}$$

Here $\mathbf{E}(\mathbf{a_j})$ is the embedding of App a_j. The scores mentioned above not only take into account users' historical trajectory but also consider their App usage preferences by factoring in the balance coefficient between App and location, as well as the time decay effect. The App with the highest score, determined through the ranking of scores among different Apps, will be considered as the prediction for users at a later time τ.

3) PREDICTION CASE

As illustrated in Figure 3, based on this records, we want to predict the Apps the user will use at 14:00. We obtained embedded vectors for nodes through MP-GT representation learning. The predictive model takes the embedded vectors of nodes from the user's usage records as input, generating a user profile using Equation 10. Finally, by using Equation 11, the user profile is compared in normalized similarity with embedded vectors of all Apps, and the App with the highest similarity is selected as the predicted App usage for the user at 14:00.

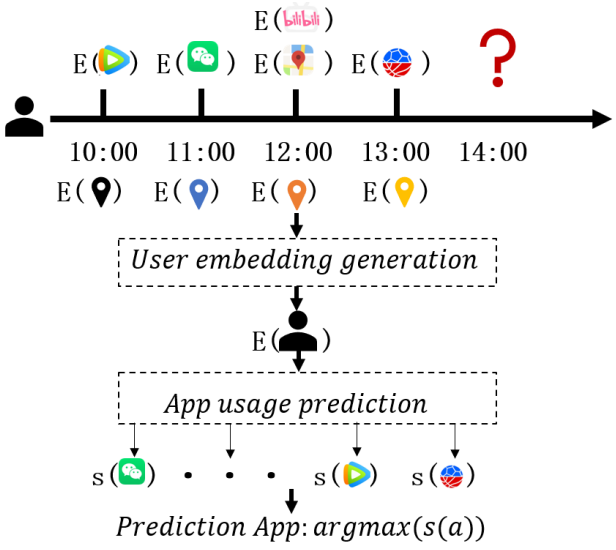

FIGURE 3. An illustration of the prediction case: Forecasting App Usage at 14:00 for a user from Five Consecutive App Records.

IV. EXPERIMENTS

A. EXPERIMENT SETUP

1) BASELINE AND CLASSIC BENCHMARK METHODS

In order to evaluate our proposed MP-GT model, we compare it against a selection of advanced representation learning

models and classic benchmark approaches for App prediction tasks, as discussed in [12]. The comparison aimed to assess the effectiveness of our model in capturing spatio-temporal App usage patterns. These methods include:

a: REPRESENTATION LEARNING-BASED BASELINES

We have compared our MP-GT with a set of contemporary representation learning models, where Grarep, Node2vec, Metapath2vec, ReconEmbed, GCN and SA-GCN operate on the same graph as our model. On the other hand, GraphEmbed and CAP construct their own graphs.

- **Grarep** [20]: This matrix factorization-based method integrates global structural information into the learning process on weighted graphs.
- **Node2vec** [21]: It employs random walk sampling from the graph, treating random walks as sentences, and applies skip-gram for node embedding.
- **Metapath2vec** [22]: This method samples meta paths as random walks and uses a heterogeneous skip-gram model for node embedding, with a focus on the type information of nodes.
- **ReconEmbed** [23]: It samples random walks from the graph and employs gradient descent to learn embedding vectors directly.
- **GraphEmbed** [23]: This approach enhances the co-occurrence graph by adding spatial and temporal neighborhood edges and preserves first and second-order proximities in the graph for representation learning.
- **CAP** [24]. Stand for context-aware App usage prediction. It considers the contextual information (location & time). CAP devises three bipartite graphs, each based on a different edge type (App-location, location-time, and App-time). These graphs are then embedded into a common latent space, enabling personalized mobile App predictions.
- **GCN** [25]: This method is a variant of our approach that applies GCN to graphs without node attributes, utilizing one-hot vectors for encoding node information.
- **SA-GCN** [12]. Short for semantic-aware representation learning via graph convolutional network, is the widely used method for App usage prediction. SA-GCN explores the novel graph by incorporating node properties, node types, and various edge. Then it learns the representation of nodes in the graph via GCN for App usage prediction directly.

Notably, Grarep and Node2vec do not account for node types. While Metapath2vec samples meta paths, it may not fully capture spatio-temporal characteristics. In comparison to ReconEmbed, GraphEmbed, and CAP, which also address co-occurrence, MP-GT excels in modeling complex relationships among diverse entities through deep node attribute propagation in Graph Convolution layers.

In addition to these representation learning models, we implemented six classic methods as benchmarks for App prediction tasks:

b: CLASSIC BENCHMARK METHODS

- **MRU** [6] (Most Recently Used): Predicts App usage based on the most recently used Apps in the last timestamp, assuming continuity in App usage.
- **MFU** [6] (Most Frequently Used): Identifies the most frequently used Apps based on historical user App usage, without considering time and location context.
- **Falcon** [7]: Utilizes user location and temporal access patterns as contextual information to predict App launches.
- **APPM** [26]: Similar to Falcon, this method incorporates spatio-temporal information as contextual features and adapts to usage dynamics for personalized App prediction.
- **MFApp** [9] (Most Frequent App): Incorporates robust similarity estimation between users and uses a simple voting scheme for prediction.
- **Bayesian** [27]: This approach designs a Bayesian network using contextual information, including time, location, user profile, and the latest used App, for predicting mobile App usage.

2) IMPLEMENTATION DETAILS AND PARAMETER SETTINGS

To learn the representations of App, location, and time units, our first step is to construct an App usage graph. To assess the performance of App usage prediction without post-learning process, we reserve some data for testing. Specifically, we sort the data chronologically and use the initial 80% for graph construction on both weekdays and weekends for each user. Subsequently, the last 20% of the data serves as the testing set for predicting App usage. Our experiments were implemented in PyTorch with GPU acceleration. Here are more details regarding the model configuration:

· Adam Optimizer Parameters:

Weight Decay: Set to 0.0001 to control model complexity and prevent overfitting.

Learning Rate: Set to 0.01 after experimental tuning to balance convergence speed and stability.

· Training Epochs, Iterations and Mini-Batch Size:

Epochs: 5 training epochs were performed, determined empirically to allow the model to learn the features of the data adequately.

Iterations: Each epoch comprised 512 iterations, a setting that balances training speed and model performance.

Mini-Batch Size: Set to 1024, the number of samples used for updating model weights in each iteration. A larger mini-batch size can expedite training.

· Output Embedding Dimension and Negative Samples:

Output Embedding Dimension (D_o): Set to 64, possibly to control the number of model parameters and avoid excessive complexity. Negative Samples:

Negative Samples (K): Set to 5, representing the number of negative samples used. Appropriate negative sample numbers assist the model in better learning to differentiate positive samples.

We set the output embedding dimension to $D_o = 64$, and the number of negative samples was fixed at 5 (setting $K = 5$) for the purpose of comparison across different models. The selection of these configurations is often determined through iterative experimentation and adjustments to ensure optimal performance on the given task and dataset.

B. EMBEDDING VISUALIZATION

To assess the quality of the embedding vectors obtained by the MP-GT learning model, we employed the t-SNE (t-Distributed Stochastic Neighbor Embedding) method to map D_o-dimension embedding vectors to a lower-dimensional space, specifically 2-dimensional vectors. We conducted an analysis of the embedding quality based on the following aspects:

1) TIME NODES VISUALIZATION

In Figure 4, we observe 48 points, which represent the overall topology of the visualization for a 24-time unit span on weekends and weekdays. The MP-GT method successfully distinguishes between weekdays and weekends, as well as day and night time units. This finding indicates that App usage exhibits diverse characteristics and regular patterns during different time ranges. Furthermore, there is a noticeable separation between time nodes on weekends and weekdays, indicating a significant difference in App usage patterns when users are awake. For instance, users may be more inclined to use recreational Apps on weekends and work-related Apps on weekdays. On the other hand, the gap between time nodes for nights on weekends and weekdays is relatively close together, suggesting that App usage is minimal during nighttime. Overall, the time embedding generated by our MP-GT model effectively represents the temporal characteristics of App usage.

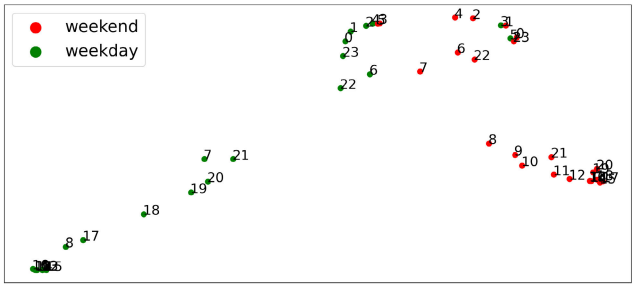

FIGURE 4. Visualization of time node embeddings across weekends and weekdays for a 24-Time unit span.

2) APP NODES VISUALIZATION

In our dataset, we have a total of 1792 different Apps. To analyze the App embeddings in a simplified manner, we utilize App categories. By examining the top subgraph in Figure 5, we find that the App categories include finance, references, photo, and infant & mom. In contrast, the lower subgraph comprises games, education, music, and sports. These categories are well separated in the embedding space. This separation is primarily attributed to the MP-GT model's ability to preserve attribute information, including App categories. Overall, the MP-GT model successfully captures and preserves the attribute information of different App categories, resulting in distinct separations in the App embedding space as observed in Figure 5.

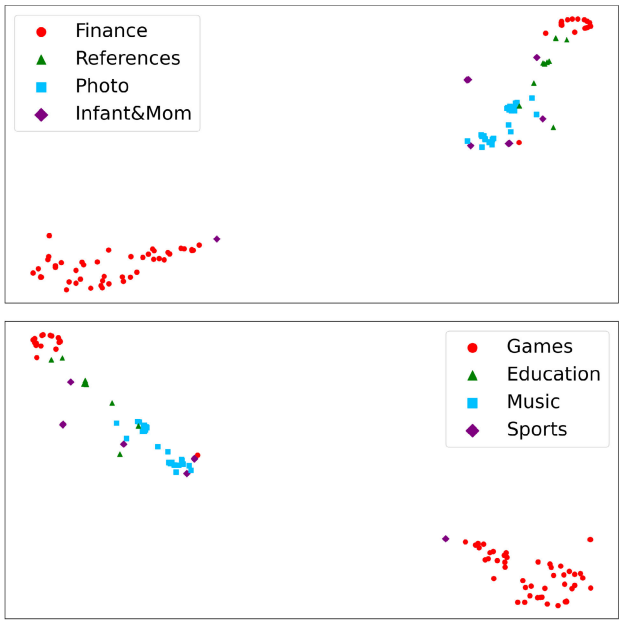

FIGURE 5. App node embeddings Visualization: Distinct separation of categories in the embedding space by MP-GT model.

3) ALL NODES VISUALIZATION

In Figure 6, we consider the App, time, and location nodes simultaneously and visualize them. We also highlight several edges with high weights to depict the co-occurrence relationships in the graph $\mathcal{G}$. Thick edges indicate high edge weights. From the visualization, it is apparent that the App nodes are mostly located at the top of the figure. Additionally, there is a clear separation between the App, time, and location nodes. These findings reinforce the fact that our MP-GT model effectively captures co-occurrence

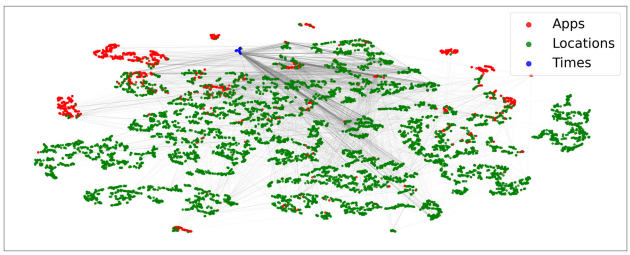

FIGURE 6. Comprehensive visualization of node Embeddings: Simultaneous integration of app, time, and location nodes, emphasizing co-occurrence relationships with highlighted edge weights in graph $\mathcal{G}$.

relationships when provided with spatio-temporal context information. The visualization in Figure 6 demonstrates the capability of our MP-GT model to capture and represent the co-occurrence relationships among App, time, and location nodes, highlighting the model's effectiveness in incorporating spatio-temporal context information.

C. RESULT ANALYSIS

1) PREDICTION PERFORMANCE

a: PERFORMANCE METRIC

We critically selected two evaluation metrics: MRR and Accuracy@k, to critically assess the performance of the App usage prediction.

MRR is called mean reciprocal rank, which is an international common metric to evaluate search algorithms. Formally, it can be defined as

$$MRR = \frac{1}{|R_{test}|} \sum_{i=1}^{|R_{test}|} \frac{1}{rank(a_i)}. \tag{12}$$

Here R_{test} is the number of test set for App usage dataset, a_i is the App to be tested, $rank(a_i)$ represents the position of the App a_i in the ranking list. Specifically, if the ground truth App is ranked n-th in the ranking list, the value of $rank(a_i)$ is equal to n.

Accuracy@k is defined as the possibility of correctly predicting the next top-k Apps regardless of order, which can be represented as:

$$Accuracy@k = \frac{|hit@k|}{|R_{test}|} = \sum_{i=1}^{|R_{test}|} \frac{|hit@k(a_i)|}{|R_{test}|}. \tag{13}$$

MRR is calculated based on the concept of hit@k, where hit@k represents the number of App to be tested in the test set that appear in the top-k position in the ranking list. In detail, the value of hit@k for each App equals 1 if the ground truth App appears in the top-k position, otherwise, equals 0.

b: PERFORMANCE COMPARISON

Table 1 presents the performance and relative gain of the evaluation metrics when our model and the baseline approach are applied to the App usage prediction task. Drawing upon the aforementioned results, we can derive the following conclusions:

- Representation learning-based methods, such as Metapath2vec, Node2vec, and GraphEmbed, consistently demonstrate significant improvements in terms of accuracy. This underscores the crucial significance of acquiring representation learning in various applications.
- Our proposed MP-GT model significantly outperforms all baseline models. Compared to MFU, i.e., the best counting-based baseline, it provides relative performance gain of 50.34%, 23.10% in terms of Accuracy@1, and MRR, respectively. Besides, it also significantly outperforms the representation learning method, GraphEmbed, by 22.78% in Accuracy@1. When compared to the widely used baseline SA-GCN, our model achieves a relative performance gain of 13.33% in terms of Accuracy@1, 8.15% in terms of Accuracy@10, and 13.57% in terms of MRR. Furthermore, when compared with the popular baseline CAP, our model demonstrates significant improvements across all three metrics. Specifically, our model showcases growth in Accuracy@1, Accuracy@10, and MRR, achieving 0.7402 times, 0.7906 times, and 0.9702 times growth, respectively. These results highlight the consistent superiority of our MP-GT model over the advanced baseline model, highlighting its exceptional performance in the App usage prediction task.
- When comparing the performance of the MP-GT model with App prediction algorithms, it becomes evident that, while many non-graph prediction methods excel beyond the fundamental counting-based methods like MRU and MFU, these non-graph methods are primarily tailored for context-aware App prediction. However, the performance improvements they offer are modest at best. This suggests that these approaches are inadequate for modeling intricate spatial-temporal relationships in App usage scenarios. On the contrary, our MP-GT model outperforms them. This further underscores the superiority of our representation-learning approach in effectively modeling relationships among various entities. Additionally, during the prediction phase, certain non-graph prediction methods necessitate the computation of similarity between different users and features, which can be time-consuming. In contrast, our approach enables the direct prediction of App usage from our pre-trained embeddings, avoiding the need for additional computations when compared to other baseline methods.
- In the case of our MP-GT model and SA-GCN model, the objective function guided by meta-paths is employed for model optimization. On the other hand, the CAP model is optimized using the objective function guided by edges. These comparisons clearly demonstrate the effectiveness of integrating meta-paths in enhancing the model's capability to extract heterogeneous information and reconstruct structure-rich graphs. This integration plays a crucial role in improving the accuracy of App prediction.
- Through a comparison between our MP-GT model and SA-GCN, it is evident that the embeddings obtained through the GT method outperform those obtained through GCN when used for App usage prediction task. This comparison highlights the superior feature extraction capability of the GT method, enabling it to effectively extract features from different nodes. This contributes to the advancement of App usage prediction.

2) LOSS

Our MP-GT model and the baseline methods utilize distinct loss functions, resulting in significant variations in loss data.

TABLE 1. Comparison of performance between our model and baselines on the app usage prediction task.

Metrics	Accuracy@1		Accuracy@10		MRR	
Value / Method	Result	Relative Gain Δ	Result	Relative Gain Δ	Result	Relative Gain Δ
MRU	0.110	100.90%	0.539	20.59%	0.211	69.19%
MFU	0.147	50.34%	0.590	10.17%	0.290	23.10%
Falcon	0.171	29.24%	0.566	14.84%	0.281	27.05%
APPM	0.179	23.46%	0.597	8.88%	0.298	19.80%
MFApp	0.175	26.29%	0.594	9.43%	0.299	19.40%
Bayesian	0.161	37.27%	0.523	24.28%	0.266	34.21%
Grarep	0.112	97.32%	0.274	137.23%	0.167	113.77%
Node2vec	0.153	44.44%	0.505	28.71%	0.261	36.78%
Metapath2vec	0.164	34.76%	0.518	25.48%	0.275	29.82%
ReconEmbed	0.125	76.80%	0.354	83.62%	0.173	106.36%
GraphEmbed	0.180	22.78%	0.545	19.27%	0.272	31.25%
CAP	0.127	74.02%	0.363	79.06%	0.181	97.02%
GCN	0.168	22.78%	0.555	17.12%	0.286	24.83%
SA-GCN	0.195	13.33%	0.601	8.15%	0.314	13.57%
MP-GT	0.221	0	0.650	0	0.357	0

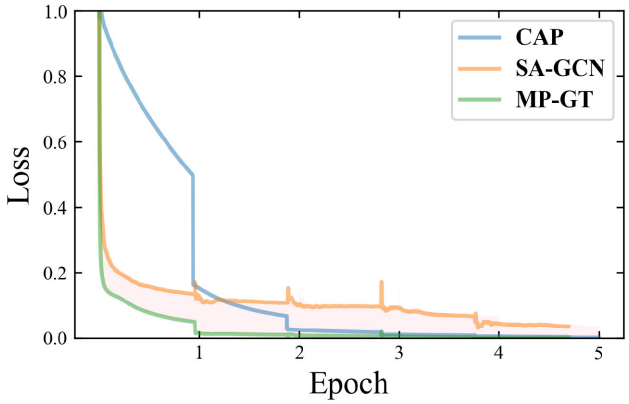

FIGURE 7. Comparison of loss convergence performance- MP-GT vs SA-GCN and CAP: Achieving faster convergence in a shorter epoch.

To facilitate a meaningful comparison of their convergence rates, we transform their respective loss values to a range of 0 to 1 using the min-max normalization method.

Figure 7 illustrates this results. For the convergence property of the loss function, we expect that the convergence (learning) speed should be faster when the loss is larger. Upon examining the loss during the training process, we observed that MP-GT exhibits a faster drop in loss values compared to SA-GCN and CAP, ultimately achieving superior results in fewer epochs. These findings indicate that MP-GT demonstrates better convergence performance.

3) TRAINING TIME

The relationship between epoch and step is described as follows: $step = [(Number\ examples) * epoch]/batsize$. Here step, i.e., iteration. It indicates that the learning is performed once per iteration for each batch size samples.

By observing steps per secend, examples per second and average time per epoch, CAP has the best performance in these indexes, followed by SA-GCN and MP-GT. We conclude that MP-GT is the most complex model for a limited time.

TABLE 2. Training time comparison between MP-GT and Baselines, including SA-GCN and CAP.

Method	CAP	SA-GCN	MP-GT
Steps per secend	2.37	0.57	0.28
Examples per second	2428	589	287
Average time per epoch	0.056h	0.247h	0.508h
Total epoch	10	100	10
Num steps	4180	51200	5120
Total training time	1.13h	24.75h	5.08h

However, the comparison presented in Table 2 illustrates that our MP-GT model demands around 5.08 hours for training, resulting in a substantial reduction in execution time, nearly 79.47%, when compared to the SA-GCN baseline. Additionally, it is evident from Table 1 that MP-GT achieves a considerably higher prediction accuracy on the App usage prediction task compared to CAP.

In summary, our MP-GT model exhibits a higher computational complexity, but it achieves a shorter training time due to its fast convergence rate. The underlying reason for this can be attributed to the following factors: When leveraging the embeddings with local information obtained from GCN, Transformer effectively captures global features through its attention mechanism. Subsequently, MP-GT further optimizes these features using a meta path-guided objective function.

4) GENERALIZATION ABILITY

In addition, as testing set is constructed from time periods and user behaviors unseen by the model, its performance evaluation reflects the model's generalization ability to new Apps. Here is a detailed discussion on the generalization performance of the MP-GT model:

- Effectiveness of Time-Sorted Testing. We utilized the first 80% of data from each user's weekdays and weekends to construct the graph, reserving the final 20% for predictive testing. This approach allows us to assess the

model's performance on unseen time periods and user behaviors. Such testing methodology ensures thorough validation of our model's generalization performance on new data, thereby enhancing the reliability of our experimental results.

- Synergistic Impact of GCN-Transformer Structure. The fusion of GCN and Transformer in the MP-GT model created a synergistic effect, enabling the model to capture long-distance dependencies and local subgraph structures more effectively. This structural design forms a solid foundation for the model's generalization performance, enhancing its understanding of application usage graphs.
- Optimization Strategy with Meta-Path Guidance. Following the embedding of local information obtained from GCN, the Transformer efficiently captures global features using an attention mechanism. Subsequently, the MP-GT model further optimizes these features by introducing a meta-path-guided objective function. This meta-path-guided optimization strategy enhances the model's adaptability to diverse datasets and tasks, contributing to improved generalization performance.
- Flexibility of Learning Methods. The adoption of a specific learning method, focusing on the ''App time-location'' meta-path-guided learning strategy, demonstrates the model's flexibility. The optimization through meta-path guidance ensures the model's adaptability to various co-occurrence patterns, positively impacting generalization performance.
- Consistency in Experimental Evidence. The experimental results demonstrate that our model, MP-GT, outperforms others, achieving optimal predictive performance and demonstrating superior generalization capabilities.

In conclusion, the MP-GT model has achieved inspiring results in terms of generalization performance. The combination of experimental findings and discussions confirms the model's adaptability to diverse applications and datasets, providing robust support for its practical utility.

5) SCALABILITY AND EFFICIENCY

Experimental results show that MP-GT demonstrated improved predictive performance and shorter training time compared to SA-GCN, with good generalization performance. Therefore, the MP-GT model possess better scalability and efficiency when dealing with large-scale datasets. The dissussions on the model's scalability and efficiency are given as follow.

Distributed computing: Large datasets often require processing across multiple computing nodes. By utilizing PyTorch's distributed training functionality, data and computations can be distributed across multiple nodes, a key factor in improving scalability and efficiency. This optimally utilizes computing resources and reduces training time.

Model architecture: The model consists of two main subnetworks, GCN and Transformer. Adaptation to new nodes and edges can be achieved through incremental learning, maintaining good scalability when dealing with continuously changing graph structures.

Hardware optimization: Leveraging high-performance GPU hardware significantly improves the model's training and inference speed on large-scale datasets.

Heterogeneous graph sampling: Using alias sampling to sample the ''App-time-location'' meta-path helps improve training efficiency. Selecting meta-paths through probability functions effectively captures relational connections in the graph without considering all possible paths, enhancing sampling efficiency and, consequently, model scalability.

Negative sampling technique: Adopting negative sampling enhances computational efficiency. By reducing the calculation probability of negative samples, it minimizes computational waste during the training process, accelerating learning.

V. RELATED WORK AND DISCUSSION

A. RELATED WORK

The prediction of App usage behavior is of great importance within social computing systems. Accurate prediction of App usage behavior is vital for enabling personalized recommendations, supporting user behavior analysis, empowering targeted advertising, and improving user experience and satisfaction. Here are some methods involved in the App usage prediction task.

- Traditional methods. Predicting App usage behavior through traditional methods relies on modeling user-specific characteristics, static application properties, and temporal features. Common approaches to predict App usage behavior include sequence pattern mining, similarity-based methods, and probabilistic graph methods. However, these methods have some limitations. Limitations of sequence pattern mining include long-term dependency and excessive focus on historical sequences. Similarity-based methods may not effectively capture complex patterns in user behavior. Probabilistic graph methods may encounter computational complexity challenges when dealing with large-scale data. To overcome these challenges, researchers have combined different techniques to predict App usage. For example, Yu et al. proposed a transfer learning method based on collaborative filtering in [2]. This method predicts App usage by considering the user's points of interest in specific locations.
- Context modeling methods. Context modeling enhances the accuracy of predicting user behavior by modeling factors like location, social network connections, time, and environment. Traditional approaches often overlook or inadequately consider crucial contextual details such as the user's environment, social relationships, and overall context. This leads to limited predictive performance in traditional methods. Researchers have explored context modeling techniques to overcome this limitation. Xia et al. [10] developed a multimodal embedding

method that utilizes spatio-temporal context to learn semantic embeddings of Apps. DeepApp model in [11] and [28] employ deep learning for spatio-temporal context-aware predictions. Fan et al. [29] conducted a thorough analysis of the long-term dynamic changes in App usage context. In recent years, meta-paths have gained recognition for capturing diverse relationships in heterogeneous networks. Meta-paths represent specific interactions (e.g., user-user, App-App, and user-App connections) in the user-App interaction network. Considering various meta-paths allows researchers to capture different information and improve App usage prediction accuracy. The integration of meta-paths into App usage prediction tasks demonstrates the potential for capturing relationship information and enhancing predictive accuracy.

- Graph neural networks (GNNs) models: GNNs are widely used in the modeling and analysis of graph data. They have shown excellent performance in predicting App usage behavior. These networks effectively capture relationships between nodes in a graph and conduct information propagation and representation learning. In the context of App usage prediction, relationships between Apps are modeled as a graph structure. GNNs, such as Graph Convolutional Networks (GCN), GraphSAGE [30], and Graph Attention Networks (GAT) [31], are employed to learn App representations, improving the accuracy of predictions.

B. DISCUSSION

1) BROADER IMPLICATIONS

- Potential Applications in Mobile App Development: Our research opens up potential avenues for mobile App development. By utilizing the MP-GT model, developers can gain a better understanding of user behavior, leading to improvements in App design and functionality. This allows them to cater to user needs, enhancing the overall appeal and utility of the App.
- Enhancement of User Experience: The successful application of MP-GT can significantly enhance the user experience. By providing personalized App recommendations, users are more likely to discover and use Apps that align with their interests, thereby increasing their engagement and satisfaction within the digital ecosystem.
- Impact on Future Research: This study may have far-reaching implications for future research in the field. The successful application of MP-GT could inspire research into more advanced and complex models, aiming to further enhance the accuracy and applicability of App usage predictions. Additionally, the attention to user privacy and ethical considerations may stimulate further research on how to balance personalized App recommendations with user privacy protection.

In summary, this research not only has practical applications in specific domains but also has the potential to instigate profound discussions and further research within the broader scientific community, particularly in areas related to user experience, model development, and ethical considerations.

2) FUTURE WORK

In our paper, we introduced MP-GT as a novel model for enhancing App usage prediction using meta-path-guided optimization within the GCN-Transformer framework. However, as with any research, there are several open research questions and directions for future exploration:

(a)Dynamic App Usage Prediction: Our model primarily focuses on static App usage prediction. Future research could explore adapting our approach to dynamic scenarios where App usage patterns change over time, incorporating temporal information and evolving user preferences.

(b)User Privacy and Ethical Considerations: As App usage prediction becomes more accurate, addressing privacy and ethical concerns is essential. Previous work, such as in [4], tackled privacy issues in mobile App recommendations based on knowledge graph models. Future research should explore privacy-preserving variants of our model and examine the ethical implications of personalized App recommendations.

(c)Interpretability and Explainability: Enhancing the interpretability of the model's predictions is crucial. Future research could focus on developing techniques to make MP-GT's reasoning more understandable to end-users, fostering trust.

3) PRACTICAL APPLICATIONS

Our research primarily concentrated on improving App usage prediction, but it's crucial to discuss the practical managerial significance and potential applications of our proposed MP-GT method, namely adapting or extending the MP-GT model to other applications. Here, we provide some insights into its practical applications:

(a). Adaptable Model: The MP-GT model exhibits adaptability across various domains, allowing for its extension and application beyond App usage prediction.

- Social Networks: Adapt MP-GT to predict events or interactions in social networks, leveraging social network structure and dynamics.
- Healthcare: Explore MP-GT's applicability in healthcare prediction analytics, considering patient demographics, medical history, and lifestyle factors.
- Finance: Investigate using MP-GT in financial forecasting, integrating financial indicators and market data for improved predictions.
- Fraud Detection: Apply MP-GT in fraud detection across domains, identifying abnormal patterns in user behavior.
- Smart Cities: Utilize MP-GT in smart city environments, predicting urban phenomena such as traffic patterns and energy consumption.

- Education: Explore using MP-GT to predict user interactions and preferences in educational platforms.

(b). Multimodal Recommendations: Expand MP-GT to handle multimodal recommendation tasks, including various content types such as images, text, and audio.

- Personalized App Recommendations: MP-GT can be employed to provide highly personalized App recommendations to users, improving the overall user experience on smartphones. App developers and App stores can leverage our model to increase user engagement and satisfaction.
- Content Curation: Content providers, such as news Apps or streaming services, can use MP-GT to tailor content recommendations to individual users, increasing content consumption and user retention.

(c). Managerial Significance.

- Targeted Advertising: Advertisers can use MP-GT to identify the most appropriate moments to display advertisements based on predicted user App usage. This can lead to more effective and less intrusive advertising campaigns.
- Resource Allocation: Mobile network providers can optimize network resources based on App usage predictions, ensuring a better quality of service for users during peak usage times.

These suggestions demonstrate the versatility of the MP-GT model and provide guidance for future research on its application and impact. By exploring its applicability across diverse domains, we can better understand its potential and challenges, fostering innovation in the field.

VI. CONCLUSION

This paper introduces a novel embedding representation learning model, MP-GT, for App usage prediction. The experimental results verify that MP-GT evidently outperforms other popular baselines. It is particularly noteworthy that our model achieves higher prediction accuracy than the CAP and SA-GCN in terms of performance metric thanks to the superior feature extraction and the promotion of the ability to capture semantic relationships. Moreover, MP-GT reduces 79.47% training time than SA-GCN, which benefit from fast convergence speed.

REFERENCES

[1] H. Cao and M. Lin, "Mining smartphone data for app usage prediction and recommendations: A survey," *Pervas. Mobile Comput.*, vol. 37, pp. 1–22, Jun. 2017.

[2] D. Yu, Y. Li, F. Xu, P. Zhang, and V. Kostakos, "Smartphone app usage prediction using points of interest," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 1, no. 4, pp. 1–21, Jan. 2018.

[3] Z.-X. Liao, S.-C. Li, W.-C. Peng, P. S. Yu, and T.-C. Liu, "On the feature discovery for app usage prediction in smartphones," in *Proc. IEEE 13th Int. Conf. Data Mining*, Dec. 2013, pp. 1127–1132.

[4] N. Natarajan, D. Shin, and I. S. Dhillon, "Which app will you use next? Collaborative filtering with interactional context," in *Proc. 7th ACM Conf. Recommender Syst.* New York, NY, USA: Association for Computing Machinery, Oct. 2013, pp. 201–208.

[5] X. Zou, W. Zhang, S. Li, and G. Pan, "Prophet: What app you wish to use next," in *Proc. ACM Conf. Pervasive Ubiquitous Comput. Adjunct Publication*. New York, NY, USA: Association for Computing Machinery, Sep. 2013, pp. 167–170.

[6] C. Shin, J.-H. Hong, and A. K. Dey, "Understanding and prediction of mobile application usage for smart phones," in *Proc. ACM Conf. Ubiquitous Comput.* New York, NY, USA: Association for Computing Machinery, Sep. 2012, pp. 173–182.

[7] T. Yan, D. Chu, D. Ganesan, A. Kansal, and J. Liu, "Fast app launching for mobile devices using predictive user context," in *Proc. 10th Int. Conf. Mobile Syst., Appl.* New York, NY, USA: Association for Computing Machinery, Jun. 2012, pp. 113–126.

[8] T. M. T. Do and D. Gatica-Perez, "Where and what: Using smartphones to predict next locations and applications in daily life," *Pervas. Mobile Comput.*, vol. 12, pp. 79–91, Jun. 2014.

[9] Y. Xu, M. Lin, H. Lu, G. Cardone, N. Lane, Z. Chen, A. Campbell, and T. Choudhury, "Preference, context and communities: A multi-faceted approach to predicting smartphone app usage patterns," in *Proc. Int. Symp. Wearable Comput.* New York, NY, USA: Association for Computing Machinery, Sep. 2013, pp. 69–76.

[10] T. Xia, Y. Li, J. Feng, D. Jin, Q. Zhang, H. Luo, and Q. Liao, "DeepApp: Predicting personalized smartphone app usage via context-aware multi-task learning," *ACM Trans. Intell. Syst. Technol.*, vol. 11, no. 6, pp. 1–12, Dec. 2020.

[11] H. Wang, Y. Li, M. Du, Z. Li, and D. Jin, "App2Vec: Context-aware application usage prediction," *ACM Trans. Knowl. Discovery from Data*, vol. 15, no. 6, pp. 1–21, Jun. 2021.

[12] Y. Yu, T. Xia, H. Wang, J. Feng, and Y. Li, "Semantic-aware spatio-temporal app usage representation via graph convolutional network," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 4, no. 3, pp. 1–24, Sep. 2020.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NIPS)*. Red Hook, NY, USA: Curran Associates, 2017, pp. 6000–6010.

[14] Z. Tu, R. Li, Y. Li, G. Wang, D. Wu, P. Hui, L. Su, and D. Jin, "Your apps give you away: Distinguishing mobile users by their app usage fingerprints," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 2, no. 3, pp. 1–23, Sep. 2018.

[15] H. Yao, G. Ranjan, A. Tongaonkar, Y. Liao, and Z. M. Mao, "SAMPLES: Self adaptive mining of persistent LExical snippets for classifying mobile application traffic," in *Proc. 21st Annu. Int. Conf. Mobile Comput. Netw.*, Sep. 2015, pp. 439–451.

[16] F. Aurenhammer, "Voronoi diagrams—A survey of a fundamental geometric data structure," *ACM Comput. Surv.*, vol. 23, no. 3, pp. 345–405, Sep. 1991.

[17] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[18] Z. Shen, X. Zhao, and J. Zou, "GinApp: An inductive graph learning based framework for mobile application usage prediction," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2023, pp. 1–10.

[19] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Long Beach, CA, USA, 2019, pp. 11305–11312.

[20] S. Cao, W. Lu, and Q. Xu, "Grarep: Learning graph representations with global structural information," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manag. (CIKM)*. New York, NY, USA: ACM, 2015, pp. 891–900.

[21] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: ACM, Aug. 2016, pp. 855–864.

[22] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 135–144.

[23] C. Zhang, K. Zhang, Q. Yuan, H. Peng, Y. Zheng, T. Hanratty, S. Wang, and J. Han, "Regions, periods, activities: Uncovering urban dynamics via cross-modal representation learning," in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 361–370.

[24] X. Chen, Y. Wang, J. He, S. Pan, Y. Li, and P. Zhang, "CAP: Context-aware app usage prediction with heterogeneous graph embedding," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 3, no. 1, pp. 1–25, Mar. 2019.

[25] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017.
[26] A. Parate, M. Böhmer, D. Chu, D. Ganesan, and B. M. Marlin, "Practical prediction and prefetch for faster access to applications on mobile phones," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, Sep. 2013, pp. 275–284.
[27] K. Huang, C. Zhang, X. Ma, and G. Chen, "Predicting mobile application usage using contextual information," in *Proc. ACM Conf. Ubiquitous Comput.*, Sep. 2012, pp. 1059–1065.
[28] Z. Shen, K. Yang, X. Zhao, J. Zou, and W. Du, "DeepAPP: A deep reinforcement learning framework for mobile application usage prediction," *IEEE Trans. Mobile Comput.*, vol. 22, no. 2, pp. 824–840, Feb. 2023.
[29] Y. Fan, Z. Tu, T. Li, H. Cao, T. Xia, Y. Li, X. Chen, and L. Zhang, "Understanding the long-term dynamics of mobile app usage context via graph embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 3, pp. 3180–3194, Mar. 2023.
[30] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NIPS)*. Red Hook, NY, USA: Curran Associates, 2017, pp. 1025–1035.
[31] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lió, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.
[32] D. Dave, A. Sharma, S. M. Abdulhamid, A. Ahmed, A. Akhunzada, and R. Amin, "SAppKG: Mobile app recommendation using knowledge graph and side information—A secure framework," *IEEE Access*, vol. 11, pp. 76751–76767, 2023.

XI FANG received the Ph.D. degree from Peking University. He was a Postdoctoral Researcher with Tsinghua University. He is currently a Vice Professor with Beijing Electronic Science and Technology Institute. He has published more than 100 papers in prestigious academic journals and important conferences. His research interests include optical communication, phase noise estimation, and nonlinear equalization.

HUI YANG received the B.Sc. degree in mathematics from Southwest Jiaotong University, China. He is currently pursuing the master's degree in cybersecurity with Beijing Electronic Science and Technology Institute. His research interests include graph embedding and human–computer interaction.

DING DING received the Ph.D. degree in information and communication engineering from Beijing University of Posts and Telecommunications, in 2018. She is currently with the Department of Electronics and Communication Engineering, Beijing Electronic Science and Technology Institute.

WENBIN GAO received the Ph.D. degree from Beijing Institute of Technology, Beijing, China, in 2018. Since 2018, he has been a Lecturer with Beijing Electronic Science and Technology Institute. His research interests include orthogonal frequency-division multiplexing systems, channel estimation, and SAR signal processing.

LEI ZHANG received the Ph.D. degree in vehicle engineering from China Agricultural University, in 2007. He is currently a Senior Engineer with Beijing Electronic Science and Technology Institute.

YILONG WANG received the B.E. degree in engineering from Shandong University. He is currently pursuing the master's degree in cybersecurity with Beijing Electronic Science and Technology Institute. His research interest includes the nonlinear effects of CO-OFDM systems.

LIU SHI received the B.E. degree in engineering from North China Electric Power University. She is currently pursuing the master's degree in cybersecurity with Beijing Electronic Science and Technology Institute. Her research interest includes the signal processing of CO-OFDM systems.