ODNET: A Novel Personalized Origin-Destination Ranking Network for Flight Recommendation

Jia Xu[1*†], Jin Huang[2*†], Zulong Chen[2], Yang Li[2], Wanjie Tao[2], Chuanfei Xu[3]
[1] College of Computer, Electronics and Information, Guangxi University, Nanning, Guangxi, China
[2] Fliggy, Alibaba Group, Hangzhou, Zhejiang, China
[3] Concordia University, Montreal, Quebec, H3G 1M8, Canada
xujia@gxu.edu.cn, {fengchu.hj, zulong.czl, sushu.ly, wanjie.twj}@alibaba-inc.com, chuanfeixu@126.com

***Abstract*—Origin-Destination recommendation that recommends personalized origin city (O) and destination city (D) of flight itinerary is of great value for both Online Travel Platforms (OTPs) and users. Existing studies on next location recommendation propose to model the sequential regularity of users' check-in location sequences, but cannot well solve two new challenges facing OTPs, namely the necessity of exploring $O\&D$ and learning $O\&D$ as a whole. To this end, we propose a novel personalized Origin-Destination ranking NETwork (ODNET) for flight recommendation. In particular, a heterogeneous spatial graph (HSG) which models historical interactions between users and cities is designed at first. HSG is then deployed in ODNET to identify user preference Os and Ds by exploring the neighborhood information in HSG. To cope with the second challenge, the idea of multi-task learning is employed by ODNET to learn O and D jointly so as to capture their correlations. Moreover, temporal information of Os and Ds are also considered to further improve the accuracy of origin-destination recommendation. An offline experiment on multiple real-world datasets and an online A/B test both show the superiority of ODNET towards the state-of-the-art methods. Further, the implementation and deployment details of the proposed ODNET at Fliggy, one of the most popular OTPs in China, are also described. ODNET has now been successfully applied to provide high-quality flight recommendation service at Fliggy, serving tens of millions of users.**

***Index Terms*—Recommendation System, Flight Recommendation, Origin-Destination Recommendation; Heterogeneous Spatial Graph, Online Travel Platforms**

I. Introduction

Nowadays, Online Travel Platforms (OTPs) have become the most prevalent channels for flight booking [1]–[3] and the flight booking business has become the main income resource of the OTPs. According to the financial report of Ctrip[1] in 2020, the flight booking business accounts for nearly 39% of the company's business income [4]. Similarly, for Alibaba Fliggy[2], another popular OTPs in China, the Gross Merchandise Volume (GMV) from flight booking business contributes most of its total GMV. Therefore, OTPs are very keen to understand users better so as to do well on flight recommendation tasks. The ***Origin-Destination*** **(OD) recommendation problem**, which recommends both of the origin city (O) and the destination city (D) for a user who want to book a flight based on the user's preference, is the core to support and optimize the flight recommendation tasks.

Recently, the closest research domain of the OD recommendation, i.e., next Point of Interest (POI) or location recommendation, has aroused considerable research interests. While its early works explore conventional collaborative filtering or sequential ideas to solve the problem [5]–[9], more up-to-date works [10]–[20] proposed to employ RNN-based neural networks to model the sequential dependencies of users' historical POI visits to learn their preferences, as well as considering spatial and temporal factors in different ways. Although these approaches can be adapted to solve the OD recommendation problem, they are primarily designed for serving the Location-based Social Network (LBSN) domain and hence are not effective in solving the OD recommendation problem raised by the flight recommendation scenario at OTPs. Two major challenges of OD recommendation problem still faced by OTPs are:

- **Exploration of *O*&*D*.** To explain the need of exploring O and D at OTPs, Figure 1 shows the prices of different air routes captured from the Fliggy Mobile App. Based on the information of Figures 1(a) and 1(b), a user in Ningbo may prefer to book the flight from Shanghai to Sanya, since there is no direct flight from Ningbo to Sanya and the flight from the explored nearby city (i.e., Shanghai) costs hundreds of CNY less than that from Ningbo. Similarly, a user's Ds should also be explored, largely due to the similar patterns owned by different Ds. Suppose a user in Shanghai wants to go to a seaside city for vacation. Though the user has visited the seaside city Sanya before and has never been to the seaside city Qingdao yet, Qingdao may need to be recommended in priority, because flights from Shanghai to Qingdao are much cheaper than flights from Shanghai to Sanya (by comparing Figures 1(b) and 1(c)). To sum up, it is the consideration of time or price cost that requires the exploration of Os and Ds in flight recommendation.
- **Unity of *O*&*D*.** It is the correlations between O and D that make them a unity, which indicates O (or D) cannot be separately learned or recommended. For example, there may not be a cost-effective flight from the best O to the best D, where O and D are separately learned. As another

* Equal contribution; † Corresponding author.
[1]www.ctrip.com
[2]www.fliggy.com/

example, only learning O and D in an inseparate way, the urgent demand of users for buying return tickets can be captured.

Coping with the above two challenge properties of O and D is very critical in guaranteeing the quality of OD recommendation. However, most existing next POI recommendation works for LBSN predict destinations only based on users' feedback Ds (known popularly as "exploitation") [10]–[19], which indicates they cannot solve the two challenges due to the absence of the consideration of users' origin information. Although the most recent work [20] argues the importance of considering users' Os and proposes a novel origin-aware next POI recommendation model, it fails to explore O and D and thus only gains sub-optimal performance. [15] proposes to build the interaction graphs of locations (or users) to achieve the exploration only for Ds, but it never takes the advantage of the important heterogeneous interactions between users and locations in its model.

In this paper, we study the origin-destination recommendation problem raised by the flight booking business of OTPs and propose a novel personalized Origin-Destination ranking NETwork (ODNET). To handle the first challenge, we propose the Heterogeneous Spatial Graph (HSG), following the idea of heterogeneous graph attention network (HAN) [21]. HSG has two types of nodes (i.e., user and city), two types of edges (i.e., departure and arrival) indicating historical interactions of users and cities, and spatial information of all city-type nodes. HSG thus enables ODNET to explore of Os and Ds based on heterogeneous interactions between users and cities. Then, high-quality hidden representations of each user or city are derived by exploring and attending its neighbor cities in HSG. For the second challenge, we propose an $O\&D$ joint learning component in ODNET, which is designed based on the idea of multi-task learning [22], to ensure correlations between Os and Ds can be explicitly learned from the data. Moreover, statistics of temporal information of each city are leveraged in ODNET to learn the temporal preferences of users to cities. The contributions of this paper include:

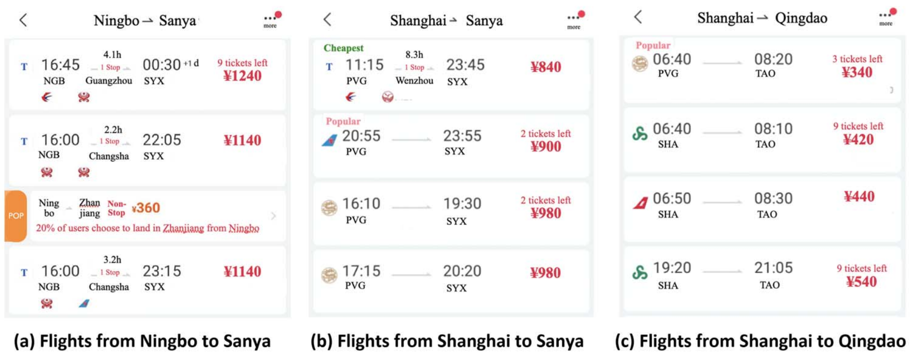
- To the best of our knowledge, this is the first attempt to study the learning of users' origin city (O)-destination city(D) preferences and to define the OD recommendation problem under the scenario of flight recommendation at OTPs.
- We propose a novel OD recommendation model (i.e., **ODNET**), to improve the effectiveness of flight recommendation at OTPs. In specific, ODNET utilizes the proposed heterogeneous spatial graph (HSG) to optimize the exploration of Os and Ds and the idea of joint learning to capture correlations between Os and Ds.
- We verify the effectiveness of the proposed ODNET on Fliggy dataset with 2.6 million users and two other public datasets of next POI recommendation. Experimental results show the superiority of ODNET compared with related methods in tackling two challenges facing OPTs. In particular, ODNET improves the CTR by on average 11.25% compared with the state-of-the-art methods in an online A/B test. ODNET has been successfully deployed at Fliggy, serving most online flight booking traffic.

II. Related Works

A. Next POI Recommendation

The closest line of works to our Origin-Destination (OD) recommendation problem is the next Point-of-Interest (POI) or location recommendation generally discussed for LBSN domain, which does not use users' origin information (Os) and only applies users' sequentially visited locations (Ds) to predict the next locations. Early works build recommendation models by using Markov Chain (MC) [8] or collaborative filtering via Matrix factorization (MF) [6], [7]. More works [9], [23], [24] are then proposed to optimized the MF or MC based approaches. Currently, RNN-based methods have shown their superiority in modeling sequential dependencies in user location sequences. ST-RNN [10] is an early work which shows that spatial and temporal intervals between neighboring POIs can be used in an RNN for next POI recommendation. In [12], a recurrent architecture CARA is proposed which leverages both sequences of user check-ins and sequence contextual information to learn users' dynamic preferences in venue recommendation. [16] presents STGN and its variant STGCN that introduce dedicated time and distance gates in LSTM to learn short and long term user preferences. In [19], an LSTPM that learns user preferences by considering temporal and spatial correlations between user past and current trajectories is proposed. There are still many RNN-based works proposed [11], [13], [14], [17], [18]. These works, however, only learns POI-POI relationships in a local view based on independent user visit sequences, which limits the model's ability to directly learn across users in a global view to recommend semantically trained POIs. In view of this, STP-UDGAT [15] is proposed to learn POI-POI relationships from both local (current user) and global (all users) views based on Spatial, Temporal and Preference (STP) POI-POI graph attention networks (GATs), while allowing users to selectively learn from other users. Note that STP-UDGAT achieves the exploration of Ds users' may prefer based on the STP GATs. However, all the before-mentioned works are not designed to work with the origin information of users (Os). Most recently, authors of [20] argue that the consideration of origin information provides several key benefits in POI recommendation. In [20], a novel STOD-PPA model is thus presented based on spatial-temporal LSTM to learn OO, DD, and OD relationships and achieve origin-aware next destination recommendation.

STP-UDGAT [15] and STOD-PPA [20] are state-of-the-art methods for next POI recommendation. However, they either fail to work with Os and thus cannot treat $O\&D$ as a unity in the learning process [15], or never explore O and D simultaneously [15], [20], proposing new challenges to OD recommendation problem at OTPs.


Ningbo → Sanya

16:45 NGB — 4.1h 1 Stop Guangzhou — 00:30 +1d SYX — 9 tickets left ¥1240

16:00 NGB — 2.2h 1 Stop Changsha — 22:05 SYX — ¥1140

POP Ningbo — Zhanjiang Non-Stop ¥360 — 20% of users choose to land in Zhanjiang from Ningbo

16:00 NGB — 3.2h 1 Stop Changsha — 23:15 SYX — ¥1140

(a) Flights from Ningbo to Sanya

Shanghai → Sanya

Cheapest 11:15 PVG — 8.3h 1 Stop Wenzhou — 23:45 SYX — ¥840

Popular 20:55 PVG — 23:55 SYX — 2 tickets left ¥900

16:10 PVG — 19:30 SYX — 2 tickets left ¥980

17:15 PVG — 20:20 SYX — ¥980

(b) Flights from Shanghai to Sanya

Shanghai → Qingdao

Popular 06:40 PVG — 08:20 TAO — 3 tickets left ¥340

06:40 SHA — 08:10 TAO — 9 tickets left ¥420

06:50 SHA — 08:30 TAO — ¥440

19:20 SHA — 21:05 TAO — 9 tickets left ¥540

(c) Flights from Shanghai to Qingdao

Fig. 1. Price information of different air routes (Screenshots captured on May 25, 2021 at Fliggy Mobile App).

B. Graph Neural Networks

Graph neural networks (GNNs) which extend the deep neutral network to handle arbitrary graph-structured data are introduced in [25]. Recently, convolution operation is generalized for graph-structured data, falling into two types: spectral methods and spatial methods. Spectral methods [26], [27] study the convolution for spectral representations of graphs. GraphSAGE [28], as a representative for spatial methods, defines convolutions directly on the graph, which can generate node embeddings by sampling and aggregating features from a node's local neighborhood in the graph. Motivated by the exciting attention mechanisms [29], [30], Graph Attention Network (GAT) [31] is proposed, which computes hidden representations of each node by attending over its neighbors. Later, GAT is widely applied to the personalized recommendation domain [14], [15], [32], where STP-UDGAT [15] is the first approach that applies GAT to optimize the next POI recommendation tasks. A limitation of GAT which is used by STP-UDGAT is that it only copes with homogeneous nodes and edges, and thus fails to utilize complex interactions among different types of objects. Seeing this, in [21], Wang et al. design the heterogeneous graph attention network (HAN). HAN generates node embeddings by aggregating features from meta-path based neighbors in a hierarchical manner. For the OD recommendation, our ODNET is the first work applying HAN to model heterogeneous interactions between users and cities, so as to facilitate the exploration of users' preferable Os and Ds. Moreover, ODNET is still able to learn Os and Ds in a joint manner, capturing correlations among them.

III. Preliminaries

Problem Definition. Let $U = \{u_1, \ldots, u_{|U|}\}$ represent the set of $|U|$ users and $OD = \{od_1, \ldots, od_{|OD|}\}$ be the set of $|OD|$ "Origin city–Destination city" pairs (abbreviated as OD pairs) corresponding to different flight itineraries. The long-term flight booking behaviors of a user $u_i \in U$ is a time-ordered sequence $L_{u_i} = \{od_{t_1}, \ldots, od_{t_m}\}$, where every OD pair associates with a flight historically booked by the user. Similarly, the short-term flight clicking behaviors of a user $u_i \in U$ is denoted by $S_{u_i} = \{od_{r_1}, \ldots, od_{r_n}\}$, in which each OD pair corresponding to a flight clicked by the user recently (e.g., in the last month). The objective of the Origin-Destination (OD) recommendation problem studied in this paper is to consider both long-term behaviors (L_{u_i}) and short-term behaviors (S_{u_i}) of user u_i to recommend the top-k OD pairs based on their probabilities of being related to the next booking flight of u_i.

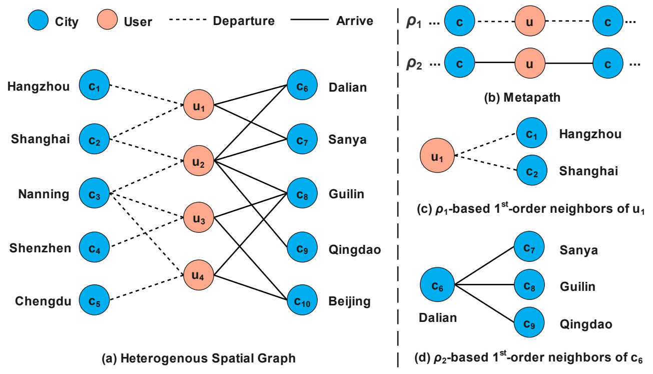

City
User
Departure
Arrive
Hangzhou
Shanghai
Nanning
Shenzhen
Chengdu
Dalian
Sanya
Guilin
Qingdao
Beijing
c_1 c_2 c_3 c_4 c_5 c_6 c_7 c_8 c_9 c_{10}
u_1 u_2 u_3 u_4
(a) Heterogenous Spatial Graph
ρ_1 … c u c …
ρ_2 … c u c …
(b) Metapath
(c) ρ_1-based 1st-order neighbors of u_1
(d) ρ_2-based 1st-order neighbors of c_6

Fig. 2. An example of heterogeneous spatial graph and related concepts derived using data offered by Fliggy.

Definition 1. ***Heterogeneous Spatial Graph (HSG)***. *A heterogeneous spatial graph is defined as $HSG(\mathcal{V}, \mathcal{E}, \mathcal{D})$ with a node type mapping function $\varphi : \mathcal{V} \rightarrow \{user, city\}$, an edge type mapping function $\psi : \mathcal{E} \rightarrow \{departure, arrive\}$, and a distance matrix $\mathcal{D} \in \mathbb{R}^{n \times n}$ where n is the number of city-type nodes in $\mathcal{V}$ and each $d_{ij} \in \mathcal{D}$ is the L_2 norm distance between the i^{th} and the j^{th} city-type nodes computed based on longitude and latitude values of the two cities.*

Example. Figure 2(a) shows an HSG illustrating the departure-type or arrive-type relationships between user-type nodes and city-type nodes, which are derived based on users' historical flight booking behaviors at Fliggy.

Definition 2. ***Metapath***. *Let v_T be a node in the node set $\mathcal{V}$ of HSG with a node type $T \in \{user, city\}$. A metapath ρ in HSG is defined as a path in the form of $\rho = v_T \xrightarrow{R}$*

$v_{T'} \xrightarrow{R} v_T \xrightarrow{R} \ldots$, where two different types of nodes (i.e., v_T and $v_{T'}$) are connected by edges with the same type $R \in \{departure, arrive\}$. The length of metapath ρ is defined as the number of edges in ρ.

Example. As shown in Figure 2(b), two types of metapaths, i.e., ρ_1 and ρ_2, are considered in the HSG. In ρ_1, two types of nodes are linked by departure-type edges, while in ρ_2 two types of nodes are connected by arrive-type edges.

Definition 3. ***Metapath Based Neighbor Cities**. Given a node $v \in \mathcal{V}$ and a metapath ρ (start from v) in an HSG, the metapath-based neighbor cities of v is defined as a set of all visited city-type nodes in $\mathcal{V}$ when v walks along the given metapath ρ. Specifically, we denote the i^{th} step visited neighbor cities of node v along ρ as its i^{th}-order neighbor cities, denoted by $\mathcal{N}_\rho^i(v)$ and with $\mathcal{N}_\rho^0(v) = v$.*

Example. Figs. 2(c) and 2(d) separately display the metapath ρ_1-based 1^{st} order neighbor cities of user u_1, denoted by $\mathcal{N}_{\rho_1}^1(u_1) = \{c_1, c_2\}$, and the metapath ρ_2-based 1^{st} order neighbor cities of city c_6, denoted as $\mathcal{N}_{\rho_2}^1(c_6) = \{c_7, c_8, c_9\}$. A node's neighbor cities conveys rich semantics. For example, $\mathcal{N}_{\rho_1}^1(u_1)$ contains all historical departure cities of user u_1, which will help to identify preferable Os of user u_i. $\mathcal{N}_{\rho_2}^1(c_6)$, as another example, includes all other visited cities of users who have visited c_6, which benefits the exploration of Ds having the same pattern. In this case, c_6 (Dalian) as a seaside city, its 1^{st} order neighbor cities contains c_7 (Sanya) and c_9 (Qingdao), which are other two famous seaside cities in China. Apparently, the metapath-based neighborhood relationships in HSG offer vital support for exploring Os and Ds a user may prefer.

IV. ODNET

In this section, we present the proposed **O**rigin-**D**estination Ranking **Net**work (**ODNET**). Figure 3 shows the overall architecture of ODNET, which consists of three modules: the Heterogeneous Spatial Graph Component (HSGC), Preference Extracting Component (PEC), and Origin&Destination Joint Learning Component (O&D-JLC). In particular, each of HSGC and PEC has two copies in the model, extracting preferences of users related to origin city (Origin-Aware) and destination city (Destination-Aware), separately. After that, O&D-JLC, which is designed based on the popular multi-task learning structure–Multi-gate Mixture-of-Experts (MMoE) [22], is deployed to learn correlations between hidden representations of candidate Os and Ds. Finally, the probability of a user choosing an OD pair to initiate a flight itinerary can be derived. We elaborate these modules in the following subsections.

A. Heterogeneous Spatial Graph Component

The Heterogeneous Spatial Graph Component (HSGC), which is designed based on the proposed Heterogeneous Spatial Graph (HSG) in Section III generates spatial semantic embeddings for users and cities by exploring them based on their neighborhoods in HSG. Inputs of an HSG copy in ODNET include basic information of users (i.e., user id and user current city id), long-term flight booking behaviors of users and short-term flight clicking behaviors of users all in the form of time-ordered city ids, as well as candidate origin city ids (w.r.t. Origin-aware HSGC) or candidate destination city ids (w.r.t. Destination-aware HSGC). Apparently, inputs of HSGC contain two types of elements – user ids and city ids. By employing Algorithm 1, HSGC is capable of generating high-quality spatial semantic embeddings for user ids and city ids, by leveraging the neighborhood information of users and cities in HSG.

Algorithm 1 HSG-based spatial semantic embedding generation.

```
Input:
    HSG($\mathcal{V}, \mathcal{E}, \mathcal{D}$); id-type feature vectors $\{h_{v_i}, \forall v_i \in \mathcal{V}\}$ with
    $h_{v_i} \in \mathbb{R}^{l \times 1}$; transformation matrix $M_T \in \mathbb{R}^{d \times l}$; exploration
    depth $K$; weight matrix $W^k$, $\forall k \in \{1, \ldots, K\}$; city
    weight matrix $A^k = [\alpha_{ij}^k]$, $\forall k \in \{1, \ldots, K\}$, $\forall v_i, v_j \in$
    $\{1, \ldots, |\mathcal{V}|\}$; metapath $\rho$ in HSG; mapping function of
    neighbor cities $\mathcal{N}_\rho : v_i \rightarrow 2^{\mathcal{V}}$;
Output:
    Spatial semantic embedding $e_{v_i}$ for all $v_i \in \mathcal{V}$
1: $e_{v_i}^0 \leftarrow M_T \cdot h_{v_i}$, $\forall v_i \in \mathcal{V}$;
2: for $k \leftarrow 1$ to $K$ do
3:     for each $v_i \in \mathcal{V}$ do
4:         $e_{\mathcal{N}_\rho^1(v_i)}^{k-1} \leftarrow \sum_{\forall v_j \in \mathcal{N}_\rho^1(v_i)} \left( \alpha_{ij}^{k-1} \times e_{v_j}^{k-1} \right)$;
5:         $e_{v_i}^k \leftarrow ReLU \left( W^k \cdot CONCAT(e_{v_i}^{k-1}, e_{\mathcal{N}_\rho^1(v_i)}^{k-1}) \right)$;
6:     end for
7: end for
8: return $\{e_{v_i}\}$, $\forall v_i \in \mathcal{V}$ and $e_{v_i} = e_{v_i}^K$;
```

The intuition behind Algorithm 1 is that for each exploration step, nodes of user-type or city-type in HSG aggregate information from their neighbors, and as this process iterates, nodes incrementally get more and more user spatial preference information. Specifically, in Algorithm 1, l-dimensional feature vectors of user ids or city ids are firstly transformed into a space with a lower dimensionality d by using a $d \times l$ transformation matrix M_T (line 1). Let k be the current exploration step, $e_{v_i}^k$ be the representation of node v_i at step k, and $\mathcal{N}_\rho^1(v_i)$ be the metapath ρ-based 1^{st}-order neighbor cities of node v_i. Note that the HSGC becomes the origin-aware HSGC in Figure 3 when the metapath ρ alternately links user-type nodes and city-type nodes by departure-type edges, and becomes the destination-aware HSGC when the metapath ρ links those two types of nodes by arrive-type edges. Therefore, the origin-aware HSGC and destination-aware HSGC focus on users' spatial preferences to Os and Ds, respectively. Then, for each step in the outer loop of Algorithm 1, each node v_i in the node set $\mathcal{V}$ of HSG aggregates the representations of its 1^{st}-order neighbor cities, i.e., $e_{v_j}^{k-1}, \forall v_j \in \mathcal{N}_\rho^1(v_i)$, by applying weights of node v_j with respect to node v_i at the k^{th} step, i.e., α_{ij}^{k-1} (line 4). Note that the aggregation step relies on the representations of nodes generated at the

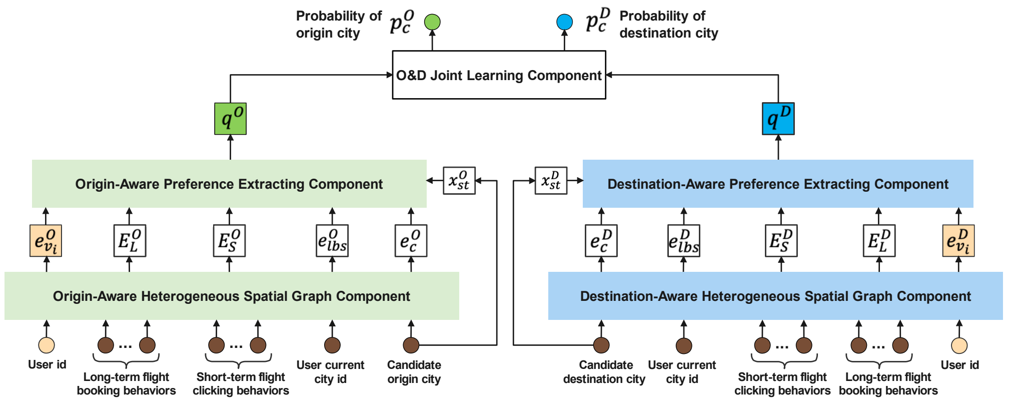

Probability of origin city
p_c^O
p_c^D
Probability of destination city
O&D Joint Learning Component
q^O
q^D
Origin-Aware Preference Extracting Component
x_{st}^O
x_{st}^D
Destination-Aware Preference Extracting Component
$e_{v_i}^O$ E_L^O E_S^O e_{lbs}^O e_c^O
e_c^D e_{lbs}^D E_S^D E_L^D $e_{v_i}^D$
Origin-Aware Heterogeneous Spatial Graph Component
Destination-Aware Heterogeneous Spatial Graph Component
User id
Long-term flight booking behaviors
Short-term flight clicking behaviors
User current city id
Candidate origin city
Candidate destination city
User current city id
Short-term flight clicking behaviors
Long-term flight booking behaviors
User id

Fig. 3. The architecture of the ODNET model.

previous iteration, i.e., $k-1$. After the aggregation phase, the current representation of node v_i, i.e., $e_{v_i}^{k-1}$, is concatenated with its aggregated neighborhood representations, i.e., $e_{\mathcal{N}_\rho^1(v_i)}^{k-1}$. The concatenated feature vector is then fed through a fully connected layer with a nonlinear activation function (line 5), after which the representation of node v_i at the k^{th} step, i.e., $e_{v_i}^k$, is derived. Finally, the spatial semantic embedding of each $v_i \in \mathcal{V}$, i.e., e_{v_i}, is the representation of v_i at the K^{th} step, which is returned by the algorithm (line 8). By iteratively aggregating information delivered by further cities neighbors in HSG, output embeddings of user ids and city ids contain explored user spatial preference information to Os (w.r.t. Origin-aware HSGC) or Ds (w.r.t. Destination-aware HSGC).

The city weight matrix $A^k = [\alpha_{ij}^k]$ used by the k^{th} iteration of Algorithm 1 is determined by:

$$\alpha_{ij}^k = \begin{cases} \dfrac{exp\Big(ReLU(e_{v_i}^k \cdot e_{v_j}^k)\Big)}{\sum_{\forall v_p \in \mathcal{N}_\rho^1(v_i)} exp\Big(ReLU(e_{v_i}^k \cdot e_{v_p}^k)\Big)} & \text{, if } v_i \text{ is a user} \\ \dfrac{exp\Big(ReLU(w_{ij} \cdot e_{v_i}^k \cdot e_{v_j}^k)\Big)}{\sum_{\forall v_p \in \mathcal{N}_\rho^1(v_i)} exp\Big(ReLU(w_{ip} \cdot e_{v_i}^k \cdot e_{v_p}^k)\Big)} & \text{, if } v_i \text{ is a city} \end{cases} \tag{1}$$

where w_{ij} is the spatial weight of city v_j when considers city v_i. Given n as the number of city-type nodes in $\mathcal{V}$ and L_2 norm distance between any two cities, i.e., $d_{ij} \in \mathcal{D}$, w_{ij} is computed by Eq. 2.

$$w_{ij} = \begin{cases} 0, & \text{if } i = j \\ \frac{1/d_{ij}}{\sum_{\forall p \in \{1,\dots,n\}} (1/d_{ip})}, & \text{if } i \neq j \end{cases} \tag{2}$$

By observing Equation 1, we know that the computation of α_{ij}^k follows the dot-product attention mechanism, which encourages the computation of neighborhood representation of node v_i focusing on the spatial semantic of node v_i. Moreover, the spatial weights of two cities, i.e, $\{w_{ij}\}$, are also involved in the computation of α_{ij}^k when v_i represents a city, to assign a spatially nearer neighbor city of node v_i with a larger weight in the neighborhood aggregation. Since inputs of HSGC only contain user ids or city ids, each input can thus be converted to embedding results focusing on user spatial preferences to O (or D) by origin-aware HSGC (or destination-aware HSGC). In Figure 3, notations $e_{v_i}^X$, E_L^X, E_S^X, e_{lbs}^X, and e_c^X represent the X-aware embedding results of user's id, long-term behaviors, short-term behaviors, current city id, and candidate city ids, respectively.

B. Preference Extraction Component

After HSGC learning spatial semantic embeddings of inputs by applying HSG, the follow-up Preference Extraction Component (PEC) further extracts preferences users based on their short and long-term behaviors. The architecture of PEC is shown in Figure 4. Let the embeddings of user long-term flight booking behaviors and user short-term flight clicking behaviors denote by $E_L = [e_L^1, \dots, e_L^t]$ and $E_S = [e_S^1, \dots, e_S^t]$ respectively, where each element in E_L or E_S (i.e., the embedding of a city id) is from $\mathbb{R}^d$. In Figure 4, E_L and E_S are separately fed into an encoding layer. The encoding layer applies the multi-head attention mechanism [30] which enables the PEC to model users' preferences from multiple view of interests [33]. The encoded matrix of E_L or E_S after the multi-head attention, denoted by $\hat{E}_X = [\hat{e}_X^1, \dots, \hat{e}_X^t]$, is calculated as:

$$\hat{E}_X = MultiHead(E_X) = concat(head_1, \dots, head_h)W^O, \\ \text{where } head_i = Attention(E_X W_i^Q, E_X W_i^K, E_X W_i^V), \tag{3}$$

where h represents the amount of heads, $W^O \in \mathbb{R}^{hd_k \times d}$ denotes the weight matrix of output linear transformation with $d_k = \frac{1}{h}d$, $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d \times d_k}$ are learnable projection matrices for the i^{th} head corresponding to query, key, and value respectively.

After the encoding layer, the output matrix of a user's short-term behaviors, i.e., $\hat{E}_S \in \mathbb{R}^{t \times d}$, is processed by an average pooling layer so as to derive a representation vector of E_S, represented as v_S in Figure 4. In the follow-up attention layer, a dot-product attention mechanism is applied following Eq. 4, by setting the vector v_S as query and the matrix $\hat{E}_L$ as both key and value. The purpose of the dot-product attention is to encourage the computation of users' historical preferences

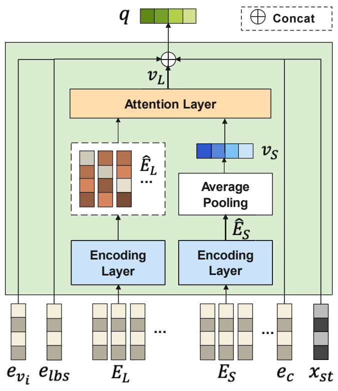


Fig. 4. Architecture of PEC.

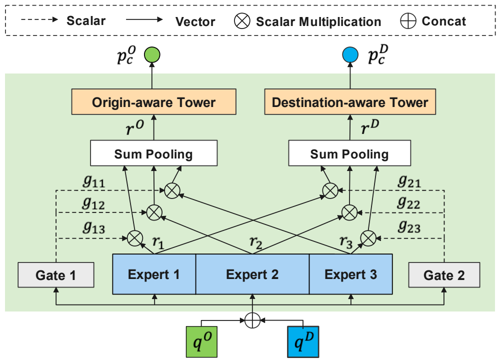


Fig. 5. Architecture of O&D-JLC.

focusing on their latest flight-booking intentions reflected by their short-term flight clicking behaviors.

$$e_i^* = v_s^T W^* \hat{e}_L^i, \tag{4}$$

where $\hat{e}_L^i \in \hat{E}_L$, and $W^* \in \mathbb{R}^{d \times d}$ is a learnable weight matrix.

Then, the output of attention layer, i.e., v_L, is determined by:

$$v_L = \sum_{i=1}^{t} \bar{e}_i^* \hat{e}_L^i, \tag{5}$$

where $\bar{e}_i^* = exp(e_i^*)/\sum_{j=1}^{t} exp(e_j^*)$.

Finally, the PEC-generated user preference representation v_L, the HSGC-generated spatial semantic embeddings of user id (e_{v_i}), user current city id (e_{lbs}), candidate city id (e_c), as well as a vector x_{st} which contains temporal statistics of cities (such as the number of visits to a city in the last month or in the same period of history) are concatenated into a representation for origin city, i.e, q^O, if E_L^O and E_S^O are fed to PEC, or a representation for destination city, i.e., q^D, if E_L^D and E_S^D are fed to PEC. The introduction of x_{st} is very helpful in identifying temporal preferences of users to cities.

C. O&D Joint Learning Component

Considering the unity property of O and D, the Origin&Destination Joint Learning Component (O&D-JLC), which is inspired by the Multi-gate Mixture-of-Experts (MMoE) [22] multi-task learning structure, is proposed to learn correlations between the task of predicting O and the task of predicting D. As shown in Figure 5, O&D-JLC consists of two tower networks, each of which corresponds to a task of predicting O or predicting D, three Multilayer Perceptron (MLP) networks, each of which is called an expert, two gating networks introduced for two tasks, each of which is called a gate. Both MLP networks and gating networks in O&D-JLC take the concatenation of PEC-generated representations of O and D, i.e., $q^{\oplus} = CONCAT(q^O, q^D) \in \mathbb{R}^{d_q}$, as the input. The output of the i^{th} expert, denoted by $r_i \in \mathbb{R}^{d_r}$, is computed by Eq. 6 below.

$$r_i = W^{expert_i} q^{\oplus}, \tag{6}$$

where, $W^{expert_i} \in \mathbb{R}^{d_r \times 2d_q}$ is a weight matrix for the i^{th} expert.

The output of the j^{th} gate in O&D-JLC is a triplet, denoted by $r_{g_j} = \{g_{j1}, g_{j2}, g_{j3}\}$ with $\sum_{k=1}^{3} g_{jk} = 1$, which is derived through multiplying the input by a trainable weight matrix $W^{gate_j} \in \mathbb{R}^{3 \times 2d_q}$ and then applying the Softmax function (see Eq. 7 for details).

$$r_{g_i} = softmax(W^{gate_j} q^{\oplus}) \tag{7}$$

The triplet output by each gate is then fed into a sum pooling layer which treats values in triplet as weights to assemble outputs of three experts in a weighted sum manner, allowing different tasks to use experts differently. The results of assembled experts, denoted as $r^O, r^D \in \mathbb{R}^{d_r}$ in Figure 5, are then passed into the task-specific tower networks. In such way, the gating networks for different tasks can learn different mixture patterns of experts assembling, and thus capture the correlations between the O-prediction task and D-prediction task. Every task-specific tower network is simply nonlinear transformation of the input with a *sigmoid* layer, outputting the probability of a candidate being the next origin city of user (i.e., p_c^O) if it is origin-aware, or the probability of a candidate being the next destination city of user (i.e., p_c^D) if it is destination-aware.

D. Training & Serving

All parts of the modules in ODNET are trained jointly by back-propagation. In the training phase, we use the next booking flights of users as labels and minimize the loss function of ODNET:

$$Loss = \theta L_O + (1 - \theta) L_D, \tag{8}$$

where L_O is the loss of predicting O used by the origin-aware tower, L_D is the loss of predicting D used by the destination-aware tower, and θ is a learnable hyperparameter. Equations for computing L_O and L_D are listed in Eq. 9 and Eq. 10, separately.

$$L_O = -\frac{1}{|S|} \sum_{o^* \in S} \left(I_{o^*}^O \log p_{o^*}^O + (1 - I_{o^*}^O) \log(1 - p_{o^*}^O) \right) \tag{9}$$

$$L_D = -\frac{1}{|S|}\sum_{d^* \in D}\left(I^D_{d^*}\log p^D_{d^*} + (1 - I^D_{d^*})\log(1 - p^D_{d^*})\right) \quad (10)$$

In Equations 9 and 10, $|S|$ is the cardinality of the training dataset S. o^* (or d^*) is an origin city id (or destination city id) in S, and $p^O_{o^*}$ (or $p^D_{d^*}$) represents the probability of o^* (or d^*) being the O (or the D) of user predicted by ODNET. An indicator variable I is used to label each city id. For example, $I^O_{o^*} = 1$ presenting the next origin city id of user is o^*, and 0 otherwise.

In the serving phase, ODNET computes the probability of a given OD pair od_j related to the next booking flight of user u_i, i.e., $p^{od_j}_{u_i}$, by Eq. 11. Flights corresponding to the OD pairs with the top-K highest probabilities will be returned and recommended to user u_i.

$$p^{od_j}_{u_i} = \theta p^O_{od_j.o} + (1 - \theta)p^D_{od_j.d} \quad (11)$$

Here, $od_j.o$ and $od_j.d$ represent the O part and the D part in the OD pair od_j, respectively.

Complexity Analysis. It is not necessary to emphasize on the training cost of ODNET, since the training cost of ODNET is one-time cost and it can be easily alleviated by involving more workers that execute parallel training tasks. The serving (or inferring) cost of ODNET is mainly affected by the multi-head attention mechanism and the MLP layers. Let B_L denote the length of the user's long-term flight booking behavior sequence, D_{HSG} represent the dimension of the embedding vector derived by executing the graph embedding procedure based on HSG, M be the number of MLP layers, N be the amount of hidden nodes in each MLP layer, the inferring time complexity of ODNET is $O(D_{HSG}{B_L}^2) + O(MN^2)$. Since the part $O(MN^2)$ is determined, and it is decided by the network structure of ODNET, the inferring complexity of the ODNET is mainly dominated by the length of user long-term behavior sequence similar as in existing related POI (or location) recommendation models.

V. Experiments

In this section, we conduct plenty of experiments to answer the following research questions:

- **RQ1.** How the hyper-parameters in the proposed ODNET affect its performance?
- **RQ2.** How does ODNET perform compared with baselines and its variants?
- **RQ3.** How efficient are ODNET and comparison methods?
- **RQ4.** How do ODNET and comparison methods perform on the online travel platform?
- **RQ5.** Can ODNET cope with the before-mentioned two challenges facing OTPs in the flight recommendation scenario?

A. Experiment Settings

1) Datasets: We perform experiments on three real-world datasets, namely Fliggy, Foursqure [34], and Gowalla [7]. The Fliggy dataset, with details given in Table I, is generated on March 2021 based on user logs collected at Fliggy. In specific, the origin city (O) and destination city (D) of each flight booked by a user on a day in March 2021 are utilized to produce a positive sample for the user, denoted by (O^+, D^+). Then, given a positive sample (O^+, D^+), two samples for each of its partially negative form, i.e., (O^+, D^-) or (O^-, D^+), are generated by fixing O^+ or D^+ and randomly sampling D^- or O^-, and two samples for its negative form, i.e., (O^-, D^-), are also derived by random sampling. Besides, user long-term flight booking behaviors which are used to predict (O^+, D^+), are collected from the last two years of the day that generates (O^+, D^+), while user short-term flight clicking behaviors are gathered from the last 7 days of the day that generates (O^+, D^+). Foursquare and Gowalla are two publicly large-scale LSBN datasets containing only users' sequential historical check-in locations with no origin information, which are widely used in the evaluation of next POI recommendation models. By pruning inactive users and unpopular locations, the statistics of these two LSBN datasets are listed in Table II.

2) Metrics: We use standard metrics of Area Under Curve (AUC), Hit Ratio at the top-k (HR@k), and Mean Reciprocal Rank at the top-k (MRR@k) for evaluation. The computation of HR@k and MRR@k are defined as follows:

- **HR@*k***: Hit ratio measures if the OD pair $od^{true}_{u_i}$ with respect to the next booking flight of user $u_i \in U$ is within the top-k OD pairs, denoted as OD^{top-k}, returned by the model:

$$HR@k = \frac{1}{|U|}\sum_{\forall u_i \in U}\sum_{\forall od_j \in OD^{top-k}} hit(od^{true}_{u_i}, od_j), \quad (12)$$

  where $|U|$ is the cardinality of user set U, $hit(od^{true}_{u_i}, od_j)$ equals to 1 if the next booking flight of user u_i is relevant to od_j, and equals to 0 otherwise.
- **MRR@*k***: Mean reciprocal rank is used to measure how well the recommendation model ranked the relevant OD pairs against the irrelevant ones:

$$MRR@k = \frac{1}{|U|}\sum_{\forall u_i \in U}\frac{1}{rank(od^{true}_{u_i})}, \quad (13)$$

  where $rank(od_{u_i})$ is the position of the relevant OD pair of user u_i (i.e., $od^{true}_{u_i}$) in the top-k ranked list. Note that MRR@k equals to HR@k when $k = 1$.

3) Baselines: ODNET is compared with baselines below:

- **MostPop**: It ranks cities by their popularities, computed by the number of visits of users. A user's current city is paired up with most popular cities to get recommended flights.
- **GBDT** [35]: It is a scalable tree-based model for recommending and ranking tasks, which is generally used in industry.
- **LSTM** [36]. It is a variant model of RNN for sequential prediction.

TABLE I
STATISTICS OF FLIGGY DATASET.

Properties	Training	Testing
# of samples	21,996,450	5,299,441
# of (O^+, D^+) samples	3,142,350	757,063
# of (O^+, D^-) and (O^-, D^+) samples	12,569,400	3,028,252
# of (O^-, D^-) samples	6,284,700	1,514,126
# of users	2,037,869	587,042
# of origin cities	200	200
# of destination cities	200	200

TABLE II
STATISTICS OF FOURSQURE AND GOWALLA DATASETS.

Dataset	# of users	# of POIs	# of check-in records
FourSqure	243,680	203,219	16,571,651
Gowalla	196,344	381,595	20,365,084

- **STGN** [16]: An LSTM variant for predicting POIs, which learns long and short-term location visit preferences of users by taking both spatial and temporal factors into account.
- **LSTPM** [19]: An LSTM variant that models long-term preferences of users by the usage of a non-local network, and short-term preferences of users via a geo-dilated network. It is widely used in predicting next POIs for users.
- **STOD-PPA** [20]: It is an origin-aware deep neural network which is built on LSTM and can learn OO, DD, and OD relationships to benefit the prediction of the next destination.
- **STP-UDGAT** [15]: It is an explore-exploit model that concurrently exploits user preferences and explores new POIs based on global spatial-temporal-preference POI neighbourhoods in graph attention networks.

STOD-PPA and STP-UDGAT are the state-of-the-art models for next POI recommendation.

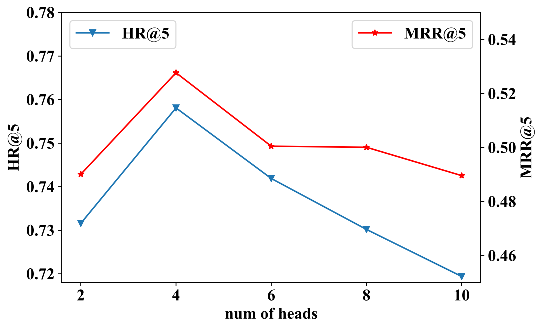
4) Variants of ODNET: To evaluate the effectiveness of each proposed optimization strategy in ODNET, ODNET is also compared with its variants in the experiment.

- **ODNET-G**: A variant of ODNET. It removes the Heterogeneous Spatial Graph Component (HSGC) from ODNET which is used to improve the embeddings of inputs by exploring preferable Os and Ds of users.
- **STL+G**: A variant of ODNET. It employs the HSGC, but learns Os and Ds in a separate manner. Finally, the O and D with the highest predicted scores are concatenated to generate recommended OD pairs.
- **STL-G**: A variant of ODNET. It is is derived by removing the HSGC from STL+G.

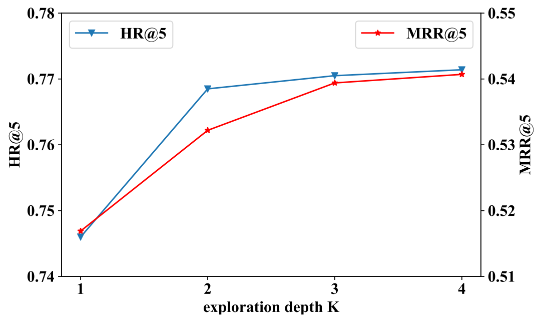
5) Other Experimental Setting:

- **Experimental Environment.** Every comparison model in the experiment is implemented using Alibaba PAI (Platform of Artificial Intelligence)[3], which is a framework developed by Alibaba Group for building large-scale distributed recommendation algorithms. PAI supports distributed training tasks that are written in the Tensorflow[4] style. Considering massive data on training, we implement each model in PAI in a distributed manner to train them in parallel. The parameter server architecture of Tensorflow is used to form a distributed approach for storing parameters, fetching data, and training models. In specific, 5 parameter servers and 50 workers are used in the architecture. Every parameter server owns 6 CPU cores with 32 GB RAM, being responsible for storing part of the parameters. Each worker also has 6 CPU cores and 32 GB memory, which fetches a portion of training samples, computes and delivers the computed results (e.g., gradients of parameters) to parameter servers. During the serving phase, each model is stored and inferred at The Personalization Platform (TPP) of Alibaba Group. TPP provides open and consistent recommendation solutions for all recommendation scenarios, so that many models and technologies can be efficiently docked on the platform.
- **Parameter Setting.** We use Adam with batch size of 128 and run the experiments with 5 epochs, while the learning rate is set to 0.01 for all comparison methods. A Gaussian distribution ($\mu = 0$ and $\sigma = 0.05$) is used to initialize the parameters used by methods built on deep-neural networks. The number of trees used in the GBDT method is set to 300, based on the proposed setting in [35]. Following the suggested setting in [37], the cardinality of a node's neighbors in HSG is restricted to 5, to ensure the computation efficiency in HSG.

[3]https://help.aliyun.com/product/30347.html
[4]https://www.tensorflow.org/



(a) Vary number of heads



(b) Vary exploration depth K

Fig. 6. Parameter Analysis of ODNET.

TABLE III
COMPARISON OF METHODS ON FLIGGY DATASET (WE BOLD BEST VALUES AND UNDERLINE NEXT BEST VALUES).

Methods		AUC-O	AUC-D	HR@1	HR@5	HR@10	MRR@5	MRR@10
Rule-based	MostPop	-	-	0.2197	0.3491	0.4091	0.2357	0.3479
Single-Task Learning (STL)	GBDT	0.8623	0.8571	0.235	0.5322	0.6254	0.3287	0.5255
	LSTM	0.8698	0.8694	0.2351	0.6421	0.6971	0.391	0.5434
	STGN	0.8749	0.8744	0.2721	0.6845	0.7432	0.4265	0.5824
	LSTPM	0.8875	0.8834	0.2794	0.7049	0.7527	0.4404	0.5949
	STOD-PPA	0.9149	0.9038	0.2806	0.7379	0.7851	0.5051	0.6122
	STP-UDGAT	0.9151	0.9074	0.2839	0.7388	0.7879	0.5054	0.6136
	STL-G	0.9084	0.897	0.2789	0.7477	0.7986	0.5017	0.6429
	STL+G	0.9246	0.912	0.3122	0.7593	0.8181	0.5219	0.6669
Multi-Task Learning (MTL)	ODNET-G	0.9201	0.9117	0.2898	0.7583	0.8081	0.5164	0.6553
	ODNET	**0.9432**	**0.931**	**0.3461**	**0.7685**	**0.8264**	**0.5322**	**0.6785**
Improvement (w.r.t. next-best)		+2.01%	+2.08%	+10.86%	+1.21%	+1.01%	+1.97%	+1.74%

TABLE IV
COMPARISON OF METHODS ON FOURSQUARE AND GOWALLA DATASETS (WE BOLD BEST VALUES AND UNDERLINE NEXT BEST VALUES).

Methods	Foursquare Dataset						Gowalla Dataset					
	AUC	HR@1	HR@5	HR@10	MRR@5	MRR@10	AUC	HR@1	HR@5	HR@10	MRR@5	MRR@10
MostPop	-	0.0262	0.0428	0.0573	0.0281	0.0315	-	0.0174	0.0319	0.0441	0.0227	0.028
GBDT	0.7303	0.0967	0.1359	0.1632	0.1004	0.1255	0.7118	0.0814	0.1196	0.1337	0.0942	0.1157
LSTM	0.7389	0.137	0.1577	0.2017	0.1388	0.1835	0.7137	0.1102	0.1389	0.1552	0.1201	0.1678
STGN	0.7693	0.1418	0.1824	0.2984	0.1504	0.2116	0.7354	0.1178	0.1601	0.2321	0.1383	0.1922
LSTPM	0.7913	0.1653	0.2341	0.3549	0.176	0.2349	0.7556	0.1315	0.1826	0.296	0.1457	0.2111
STOD-PPA	0.8085	0.1653	0.2984	0.4049	0.1725	0.2419	0.7926	0.1527	0.2805	0.3351	0.1604	0.2243
STP-UDGAT	0.8104	0.1671	0.3001	0.4056	0.1778	0.2437	0.7949	0.1531	0.2814	0.3356	0.1611	0.2258
STL-G	0.7954	0.1664	0.2893	0.391	0.1751	0.2256	0.8097	0.1507	0.2672	0.3283	0.1566	0.2117
STL+G	**0.8418**	**0.1729**	**0.3391**	**0.4175**	**0.1926**	**0.2638**	**0.8226**	**0.161**	**0.3034**	**0.3822**	**0.1791**	**0.2469**
Improvement (w.r.t. next-best)	+3.87%	+3.47%	+13.00%	+2.93%	+8.32%	+8.25%	+1.59%	+5.16%	+7.82%	+13.89%	+11.17%	+9.34%

B. Hyper-parameter Analysis of ODNET (RQ1)

- **Impact of different number of heads.** Figure 6(a) displays the impact of the number of heads set for the multi-head attention mechanism deployed in the Preference Extraction Component (PEC) of ODNET by using the Fliggy dataset. As shown in the figure, when the number of heads equals to 4, ODNET achieves its best performance in terms of both HR@5 and MRR@5, while introducing more heads beyond 4 reduces its performance. Hence, we set number of heads used in ODNET and its variants to 4 in the experiment.
- **Impact of different exploration depth K.** Figure 6(b) shows the impact of exploration depth K which is the depth when exploring neighborhood of each node in Algorithm 1, on the performance of OD recommendation. Obviously, setting K to a larger value improves both HR@5 and MRR@5, since more neighbor nodes in HSG are taken into account in computing the embedding vectors of users and cities. Experiments are also conducted to evaluate the impact of exploration depth K on the training time of ODNET. When K is set to 1, 2, 3, and 4, separately, the corresponding training time of ODNET is 55, 73, 94, and 135 minutes, respectively. There are more neighbors in the HSG needed to be explored with the growth of K, and thus the training time of ODNET increases rapidly with the growth of K. Since $K = 2$ provides a relatively significant boost in accuracy compared with the setting $K = 1$ and increases K beyond 2 giving no marked marginal returns while rising the training time of ODNET, K is set to 2 in ODNET and its variants with respect to all experiments.

C. Effectiveness of ODNET (RQ2)

We show the comparison results of ODNET, our proposed variants, and baselines in Table III (on Fliggy dataset) and Table IV (on Foursquare and Gowalla datasets). Since ODNET-G and ODNET adopt the idea of multi-task learning, they cannot be evaluated by the Foursquare and Gowalla datasets. This is because these two datasets only support single-task learning models that merely learn next POIs (i.e., Ds) of users without taking origin locations of users (i.e., Os) into account.

- Results from Tables III and IV show that models deployed our proposed HSGC (i.e., ODNET and STL+G) consistently surpass other models on three datasets, which indicates the effectiveness of exploring preferable Os and Ds of users based on the heterogeneous interactions of users and cities encoded in HSG.
- By observing Table III, as a multi-task learning model, ODNET apparently outperforms its variant (i.e., STL+G) that uses two single tasks to predict Os and Ds separately. Hence, we conclude that the proposed O&D joint learning component is effective in learning the correlations between Os and Ds, which is very helpful in improving the recommended of OD pairs.
- In Table III, by comparing the two variants of ODNET, i.e., STL+G and ODNET-G, we observe that the exploration of Os and Ds (w.r.t. STL+G) plays relatively a larger part than the learning of correlations between Os and Ds (w.r.t. ODNET-G) in the optimization of OD recommendation, since STL+G gains on average 1.8% improvement for all metrics compared with ODNET-G.
- Tables III and IV also show that STL-G, a variant of

ODNET, either gets comparable or better performance compared with the RNN based models (i.e, LSTM, LSTPM, STGN, and STOD-PPA). This is because unlike those RNN based models which can only cope with location sequences of limited length, STL-G can utilize the multi-head attention mechanism which can learn information from longer sequences so as to improve the modeling of user preference from multiple view of interests.

- According to Table III, ODNET remarkably increases the scores of AUC, HR@k, and MRR@k by average 2.8%, 10.3%, and 7.9% , respectively, compared with the best baseline STP-UDGAT. This is for ODNET successfully copes with the two challenges facing OTPs by introducing HSG in embedding computation and proposing to learn Os and Ds in a unity manner.
- Tables III and IV also show us, apart from ODNET and its variants, STP-UDGAT beats all other methods, due to the exploration of Ds based on local/global homogeneous POI-POI relationships provided by spatial-temporal-preference user dimensional graph attention networks. This, again, confirms the importance of exploring Ds in OD recommendation. STOD-PPA, as another state-of-the-art method for next POI recommendation, gains appreciable increase compared with LSTPM (e.g., highest of 14.7% for MRR@5 on Fliggy) and STGN (e.g., highest of 18.4% for MRR@5 on Fliggy), which indicate the necessity of considering origin information of users in OD (or next POI) recommendation. However, STOD-PPA fails to beat STP-UDGAT, since it only exploits users' feedback Os and Ds rather than explore them.
- MostPop gets the worst results on all datasets (see Tables III and IV), which indicates the power of deep neural networks in improving the performance of OD (or next POI) recommendation tasks.

D. Efficiency Analysis (RQ3)

In this section, we conduct experiments to verify the efficiency of both ODNET and its comparison methods by using the Fliggy dataset. Note that the MostPop method does not need to execute model training, and thus we do not compare it in this experiment. Experimental results are displayed in Table V, with each value is the average of 5 repeated tests. We observe that RNN based methods (i.e, LSTM, LSTPM, STGN, and STOD-PPA) consistently get longer training time compared with other methods, since these sequential methods update the current state based on previous states and thus are not friendly to achieve high degree of parallel training. On the other hand, ODNET and its variants (i.e., STL-G, STL+G, and ODNET-G) do not have such limitation and thus the values of their training time are all within 75 minutes. We are delighted to see that the training time of ODNET is 22.3% and 11.0% smaller than the training time of the two state-of-the-art methods STOD-PPA and STP-UDGAT, respectively. Another observation is that the training time of ODNET is larger than that of its single task variant STL+G, for more parameters are involved in the joint learning process of two tasks in ODNET. As for the model inferring time, all methods of single task learning (i.e., STL-G and STL+G) obtain longer inferring time compared with methods of multi-task learning (i.e., ODNET-G and ODNET), since they need to execute two inferences to derive Os and Ds separately. Table V shows that ODNET achieves the third best inferring time, which satisfies the requirements of online serving in industrial applications.

TABLE V
COMPARISON OF EFFICIENCY ON FLIGGY DATASET

Methods	Training Time (min)	Inferring Time (ms)
GBDT	30	8.1
LSTM	85	19.4
STGN	93	22.8
LSTPM	90	23.3
STOD-PPA	94	25.7
STP-UDGAT	82	22.5
STL-G	59	21.9
STL+G	64	23.4
ODNET-G	68	14.2
ODNET	73	16.3

E. Online A/B Test (RQ4)

Previous offline experiment results have demonstrated the superiority of the proposed ODNET method. In this subsection, we conduct online A/B test by deploying ODNET and seven competitive methods to cope with real traffic generated by $400,000$ users at Fliggy for one week in April 2021. To ensure the fairness in comparison, the scheduling engine of Fliggy is revised to approximately assign $\frac{1}{7}$ daily traffic of personalized interfaces to each method. Click Through Rate (CTR), a widely-used industrial metric, is employed to evaluate different recommendation methods for serving online traffic. The computation of CTR follows Equation 14 below, which taking the total number of times flight itineraries are clicked on and dividing it by the measured flight itinerary impressions.

$$CTR = \frac{number\ of\ clicks}{number\ of\ impressions} \tag{14}$$

Experimental results in Figure 7 show that ODNET consistently performs better than other methods, which again demonstrates the effectiveness of exploring Os and Ds from HSG in computing embeddings of inputs and the advantage of jointly learning Os and Ds in OD recommendation tasks. In specific, the CTR of ODNET is on average 11.25% higher than the two state-of-the-art methods, i.e., STP-UDGAT and STOD-PPA, and gains 17.3% improvement compared with MostPop. Further, by comparing ODNET-G and STL+G with STP-UDGAT, we can conclude that the marginals brought by employing the HSG and the joint learning component of O and D are both remarkable, which indicates the necessity of addressing the two challenges facing OTPs, i.e., the exploration of $O\&D$ and the unity of $O\&D$, in improving the performance of OD recommendation tasks.

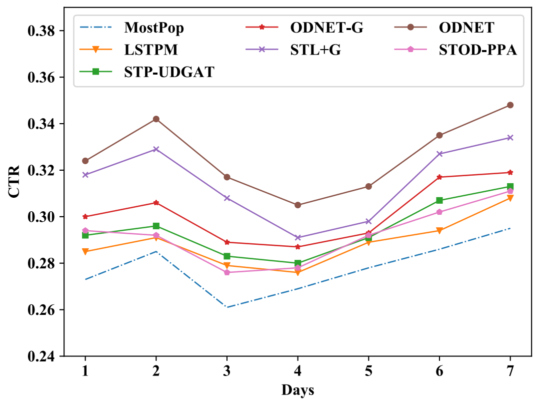


Fig. 7. Online CTRs of different methods at Fliggy from April 18, 2021 to April 24, 2021.

F. Case Study (RQ5)

To better explain the effectiveness of the proposed ODNET model in coping with the before-mentioned two challenges facing OTPs in flight recommendation, representative cases of ODNET recommended flight lists for two real-world users (named as user A and user B) captured at Fliggy Mobile App are illustrated in Figure 8(a) and Figure 8 (b), respectively.

Case 1: As shown in Figure 8(a), for user A who recently stays in the city Hangzhou of China, ODNET recommends the flight $Hangzhou \rightarrow Xi'an$ to user A in priority, since ODNET learns that flights from Hangzhou to Xi'an have been searched by the user in the last few days. Due to the same reason, the flight $Hangzhou \rightarrow Chengdu$ is also pushed to user A by ODNET. Meanwhile, by exploring the origin city (O) of the air line $Hangzhou \rightarrow Xi'an$ based on HSG and learning popular OD pairs with $Destination$=$Xi'an$ by the $O\&D$ joint learning mechanism, the nearby city of Hangzhou, i.e., Shanghai, is fetch by ODNET to generate another recommended flight to Xi'an, i.e., $Shanghai \rightarrow Xi'an$, due to its cheaper price. Similar reason for recommending the flight $Ningbo \rightarrow Chengdu$. Finally, by learning from historical buying logs of user A, ODNET finds that this user flies to the seaside city Sanya for vacation in October of each year. As a result, in addition to recommend the flight $Hangzhou \rightarrow Sanya$, more cities (i.e., Weihai and Xiamen) that have the same pattern as Sanya (i.e., near the sea), are retrieved to generate more recommended flights having cheaper prices, i.e., $Hangzhou \rightarrow Weihai$ and $Hangzhou \rightarrow Xiamen$, based on the exploration of destination cities (Ds) in HSG.

Case 2: Figure 8(b) displays the list of recommended flights for a user B who now stays in the city Beijing of China. If Os and Ds are learned in a separate manner, Beijing will never become the optimal destination city of his next itinerary. Benefit from the $O\&D$ joint learning mechanism in ODNET, ODNET learns users have strong demand in purchasing return tickets, and thus recommends the return flight $Chengdu \rightarrow Beijing$ to user B in priority, when it gets to know the user has already bought the ticket of the flight $Beijing \rightarrow Chengdu$ setting out in a certain day in the future. Meanwhile, ODNET learns from user B's recent searching logs and discovers that the user has launched searches with the keywords Qingdao, Dali, and Nanning. Therefore, flights from user B's living city Beijing to those cities that have been search by the user are recommended. By exploring HSG based on the origin city (i.e., Beijing) and deriving popular air routes to Nanning through $O\&D$ joint learning, ODNET also delivers the flight $Shijiazhuang \rightarrow Nanning$ to the user, since Shijiazhuang is not far from the origin city and the flight $Shijiazhuang \rightarrow Nanning$ costs 130 CNY cheaper than the flight $Beijing \rightarrow Nanning$. Finally, we also delighted to see from the figure that: 1) by exploring the destination city Qingdao which is a famous seaside city in China, more seaside cities which are not far from Qingdao, i.e., Yantai and Dalian, have been involved to render the recommended flights $Beijing \rightarrow Yantai$ and $Beijing \rightarrow Qingdao$ with cheaper prices; 2) by exploring the destination city Dali, which is a famous tourist city in Yunnan province, Kuming, another important tourist city in Yunnan province is employed to derive the recommended flight $Beijing \rightarrow Kunming$, which is 210 CNY cheaper than the flight $Beijing \rightarrow Dali$.

VI. System Implementation and Deployment

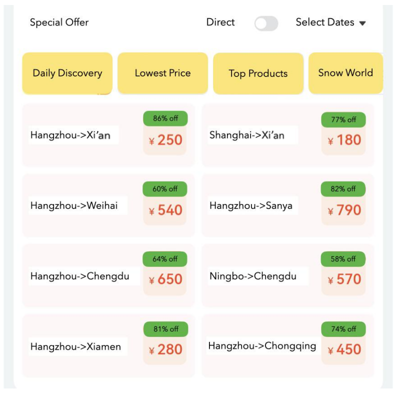
This section describes the implementation and deployment of the proposed ODNET model at Fliggy, which provides a demonstration of applying ODNET at OTPs to optimize the recommended flights. The offline training and online serving deployment of ODNET at Fliggy is illustrated in Figure 9.

A. Offline Training

As shown in Figure 9, in the offline training phase, we gather user logs and store them on the self-developed MaxCompute platform of Alibaba Group. By processing the stored logs, we construct both of the training and testing datasets. Then, we build and train the ODNET model using Alibaba Platform of Artificial Intelligence (PAI) in a distributed manner. The trained ODNET model is then saved in the Ranking Service System (RSS) of Fliggy, as shown in Figure 9, providing support for online serving.

B. Online Serving

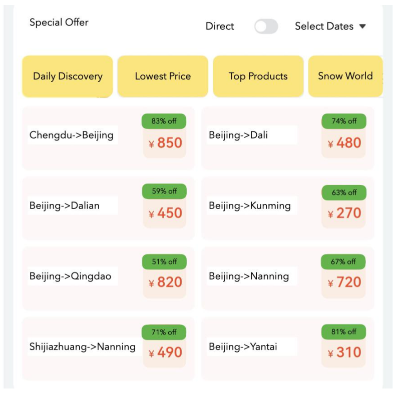
As illustrated in Figure 9, online serving of ODNET is conducted at The Personalization Platform (TPP) of Alibaba Group. When a user launches personalized flight recommendation interface in Fliggy Mobile App, TPP receives the user's request and generates a query containing user id. Then, the query is imposed to Real-Time Features Service (RTFS) system of Fliggy to fetch abundant user information (e.g., basic information, historical purchase behaviors, and real-time clicking behaviors) based on the user's id. Applying the retrieved user information, candidate OD pairs are retrieved from RTFS by using multiple recall strategies. For example, the user's current city, adjacent cities, resident cities, as well


Special Offer Direct Select Dates

Daily Discovery | Lowest Price | Top Products | Snow World

Hangzhou->Xi'an 86% off ¥250 | Shanghai->Xi'an 77% off ¥180
Hangzhou->Weihai 60% off ¥540 | Hangzhou->Sanya 82% off ¥790
Hangzhou->Chengdu 64% off ¥650 | Ningbo->Chengdu 58% off ¥570
Hangzhou->Xiamen 81% off ¥280 | Hangzhou->Chongqing 74% off ¥450

(a) Recommended Flights for User A


Special Offer Direct Select Dates

Daily Discovery | Lowest Price | Top Products | Snow World

Chengdu->Beijing 83% off ¥850 | Beijing->Dali 74% off ¥480
Beijing->Dalian 59% off ¥450 | Beijing->Kunming 63% off ¥270
Beijing->Qingdao 51% off ¥820 | Beijing->Nanning 67% off ¥720
Shijiazhuang->Nanning 71% off ¥490 | Beijing->Yantai 81% off ¥310

(b) Recommended Flights for User B

Fig. 8. Cases of Recommended Flights by ODNET to Two Users (Screenshots captured on October 20, 2021 at Fliggy Mobile App).

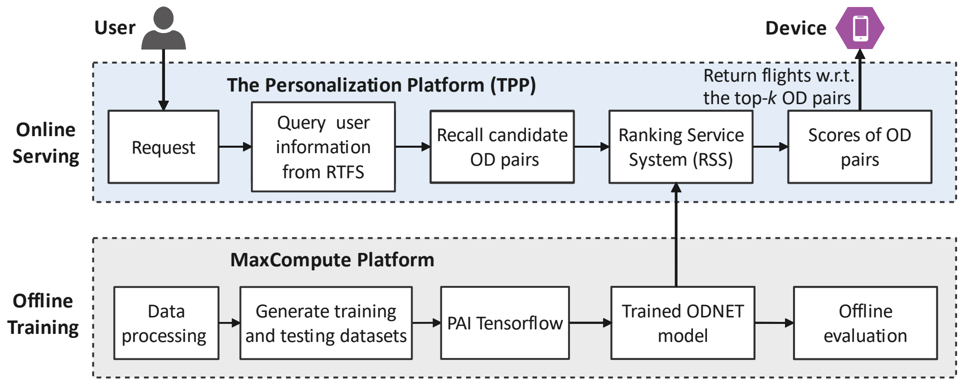

User
Device
The Personalization Platform (TPP)
Return flights w.r.t. the top-k OD pairs
Online Serving
Request
Query user information from RTFS
Recall candidate OD pairs
Ranking Service System (RSS)
Scores of OD pairs
MaxCompute Platform
Offline Training
Data processing
Generate training and testing datasets
PAI Tensorflow
Trained ODNET model
Offline evaluation

Fig. 9. Offline and Online Deployment of ODNET.

as origin cities of historical booking flights can be selected as the candidate origin cities (i.e., Os) of the user. On the other hand, candidate destination cities (i.e., Ds) of the user can be generated based on user's destination cities of historical booking flights, destination cities corresponding to popular air lines, destination cities of flights clicked by the user, and etc. After that, candidate Os and Ds are assembled to get candidate OD pairs, which are then delivered to the RSS. RSS computes the scores (or probabilities) of every candidate OD pair by using the trained ODNET model. Then, the Top-k OD pairs with the highest scores are used to generate an flight recommendation list for the user, which is finally returned to the user's Fliggy Mobile APP.

VII. Conclusion

Summary. To the best of our knowledge, this paper makes the first attempt to define and solve the Origin-Destination (OD) recommendation problem under the scenario of flight recommendation at Online Travel Platforms (OTPs). A novel ODNET model is proposed to address the OD recommendation problem. Extensive results demonstrate the effectiveness of ODNET in tackling two major challenges of the OD recommendation problem facing OTPs in recommending flights. In future, we will consider to take travel intentions of users into account, to further improve the quality of flight recommendation.

Generalization. Though ODNET is discussed under the context of flight recommendation at OTPs, it can be easily generalized to improve the next POI (or location) recommendation tasks in LSBN domain, since the consideration of origin information of users is very important according to the recent study [20]. Moreover, ODNET can also be directly applied to achieve high-quality train recommendation at OTPs.

Related Resources. To better promote the study of OD recommendation which is a prototype problem supporting many practical applications, the source code of the proposed ODNET model and the Fliggy dataset used in the experiment are all released and can be assessed though the link below: https://github.com/github-fc/OD-net

Acknowledgment

This work is supported by the National Natural Science Foundation of China (No. 62067001), and the Special funds for Guangxi BaGui Scholars. This work is partially supported by the Guangxi Natural Science Foundation (No. 2019JJA170045).

References

[1] H. Lu, J. Cao, Y. Tan, and Q. Xiao, "Auxiliary service recommendation for online flight booking," in *International Conference on Web Information Systems Engineering (WISE)*, vol. 10570, 2017, pp. 450–457.

[2] J. Cao, Y. Xu, H. Ou, Y. Tan, and Q. Xiao, "PFS: A personalized flight recommendation service via cross-domain triadic factorization," in *IEEE International Conference on Web Services (ICWS), San Francisco, CA, USA, July 2-7, 2018*, 2018, pp. 249–256.

[3] J. Huang, Y. Li, S. Sun, B. Zhang, and J. Huang, "Personalized flight itinerary ranking at fliggy," in *ACM International Conference on Information and Knowledge Management (CIKM), Virtual Event, Ireland, October 19-23, 2020*, 2020, pp. 2541–2548.

[4] T. Group, "Annual reports 2020," [EB/OL], https://investors.trip.com/financial-information/annual-reports/ Accessed Nov. 6, 2021.

[5] M. Ye, P. Yin, W. Lee, and D. L. Lee, "Exploiting geographical influence for collaborative point-of-interest recommendation," in *ACM Conference on Research & Development in Information Retrieval (SIGIR), Beijing, China, July 25-29, 2011*, 2011, pp. 325–334.

[6] C. Cheng, H. Yang, I. King, and M. R. Lyu, "Fused matrix factorization with geographical and social influence in location-based social networks," in *International Conference on Artificial Intelligence (AAAI), July 22-26, 2012, Toronto, Ontario, Canada*, 2012.

[7] Y. Liu, W. Wei, A. Sun, and C. Miao, "Exploiting geographical neighborhood characteristics for location recommendation," in *ACM International Conference on Information and Knowledge Management (CIKM), Shanghai, China, November 3-7, 2014*. ACM, 2014, pp. 739–748.

[8] C. Cheng, H. Yang, M. R. Lyu, and I. King, "Where you like to go next: Successive point-of-interest recommendation," in *International Joint Conference on Artificial Intelligence, Beijing (IJCAI), China, August 3-9, 2013*, 2013, pp. 2605–2611.

[9] J. He, X. Li, L. Liao, D. Song, and W. K. Cheung, "Inferring a personalized next point-of-interest recommendation model with latent behavior patterns," in *International Conference on Artificial Intelligence (AAAI)*, 2016, p. 137–143.

[10] Q. Liu, S. Wu, L. Wang, and T. Tan, "Predicting the next location: A recurrent model with spatial and temporal contexts," in *International Conference on Artificial Intelligence (AAAI), February 12-17, 2016, Phoenix, Arizona, USA*, 2016, pp. 194–200.

[11] R. Li, Y. Shen, and Y. Zhu, "Next point-of-interest recommendation with temporal and multi-level context attention," in *IEEE International Conference on Data Mining (ICDM), Singapore, November 17-20, 2018*, 2018, pp. 1110–1115.

[12] J. Manotumruksa, C. Macdonald, and I. Ounis, "A contextual attention recurrent architecture for context-aware venue recommendation," in *ACM Conference on Research & Development in Information Retrieval (SIGIR), Ann Arbor, MI, USA, July 08-12, 2018*, 2018, pp. 555–564.

[13] J. Feng, Y. Li, C. Zhang, F. Sun, F. Meng, A. Guo, and D. Jin, "Deepmove: Predicting human mobility with attentional recurrent networks," in *The World Wide Web Conference (WWW), Lyon, France, April 23-27, 2018*, 2018, pp. 1459–1468.

[14] A. Dadoun, R. Troncy, O. Ratier, and R. Petitti, "Location embeddings for next trip recommendation," in *The World Wide Web Conference (WWW)*, 2019, pp. 896–903.

[15] N. Lim, B. Hooi, S.-K. Ng, X. Wang, Y. L. Goh, R. Weng, and J. Varadarajan, "Stp-udgat: Spatial-temporal-preference user dimensional graph attention network for next poi recommendation," in *ACM International Conference on Information and Knowledge Management (CIKM)*, 2020, p. 845–854.

[16] P. Zhao, H. Zhu, Y. Liu, J. Xu, Z. Li, F. Zhuang, V. S. Sheng, and X. Zhou, "Where to go next: A spatio-temporal gated network for next POI recommendation," in *International Conference on Artificial Intelligence (AAAI), Honolulu, Hawaii, USA, January 27 - February 1, 2019*, 2019, pp. 5877–5884.

[17] F. Yu, L. Cui, W. Guo, X. Lu, Q. Li, and H. Lu, "A category-aware deep model for successive POI recommendation on sparse check-in data," in *The World Wide Web Conference (WWW), Taipei, Taiwan, April 20-24, 2020*, 2020, pp. 1264–1274.

[18] Q. Guo, Z. Sun, J. Zhang, and Y.-L. Theng, "An attentional recurrent neural network for personalized next location recommendation," in *International Conference on Artificial Intelligence (AAAI)*, vol. 34, no. 01, 2020, pp. 83–90.

[19] K. Sun, T. Qian, T. Chen, Y. Liang, Q. V. H. Nguyen, and H. Yin, "Where to go next: Modeling long-and short-term user preferences for point-of-interest recommendation," in *International Conference on Artificial Intelligence (AAAI)*, vol. 34, no. 01, 2020, pp. 214–221.

[20] N. Lim, B. Hooi, S. Ng, X. Wang, Y. L. Goh, R. Weng, and R. Tan, "Origin-aware next destination recommendation with personalized preference attention," in *ACM International Conference on Web Search and Data Mining (WSDM), Virtual Event, Israel, March 8-12, 2021*, 2021, pp. 382–390.

[21] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *The World Wide Web Conference (WWW), San Francisco, CA, USA, May 13-17, 2019*. ACM, 2019, pp. 2022–2032.

[22] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, and E. H. Chi, "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *ACM International Conference on Knowledge Discovery & Data Mining (SIGKDD)*, 2018, pp. 1930–1939.

[23] S. Feng, X. Li, Y. Zeng, G. Cong, Y. M. Chee, and Q. Yuan, "Personalized ranking metric embedding for next new POI recommendation," in *International Joint Conference on Artificial Intelligence (IJCAI), Buenos Aires, Argentina, July 25-31, 2015*, 2015, pp. 2069–2075.

[24] B. Chang, Y. Park, D. Park, S. Kim, and J. Kang, "Content-aware hierarchical point-of-interest embedding model for successive poi recommendation," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018, p. 3301–3307.

[25] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2005, pp. 729–734 vol. 2.

[26] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[27] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *International Conference on Neural Information Processing Systems (NIPS)*, 2016, p. 3844–3852.

[28] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *International Conference on Neural Information Processing Systems (NIPS), December 4-9, 2017, Long Beach, CA, USA*, 2017, pp. 1024–1034.

[29] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations (ICLR), San Diego, CA, USA, May 7-9, 2015*, 2015, pp. 1–15.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *International Conference on Neural Information Processing Systems (NIPS), December 4-9, 2017, Long Beach, CA, USA*, 2017, pp. 5998–6008.

[31] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[32] X. Wang, X. He, Y. Cao, M. Liu, and T. Chua, "KGAT: knowledge graph attention network for recommendation," in *ACM International Conference on Knowledge Discovery & Data Mining (SIGKDD), Anchorage, AK, USA, August 4-8, 2019*, 2019, pp. 950–958.

[33] C. Li, Z. Liu, M. Wu, Y. Xu, H. Zhao, P. Huang, G. Kang, Q. Chen, W. Li, and D. L. Lee, "Multi-interest network with dynamic routing for recommendation at tmall," in *ACM International Conference on Information and Knowledge Management (CIKM), Beijing, China, November 3-7, 2019*, 2019, pp. 2615–2623.

[34] D. Yang, D. Zhang, and B. Qu, "Participatory cultural mapping based on collective behavior data in location-based social networks," *ACM Trans. Intell. Syst. Technol.*, vol. 7, no. 3, pp. 30:1–30:23, 2016.

[35] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[36] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[37] S. Fan, J. Zhu, X. Han, C. Shi, L. Hu, B. Ma, and Y. Li, "Metapath-guided heterogeneous graph neural network for intent recommendation," in *ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2019, pp. 2478–2486.